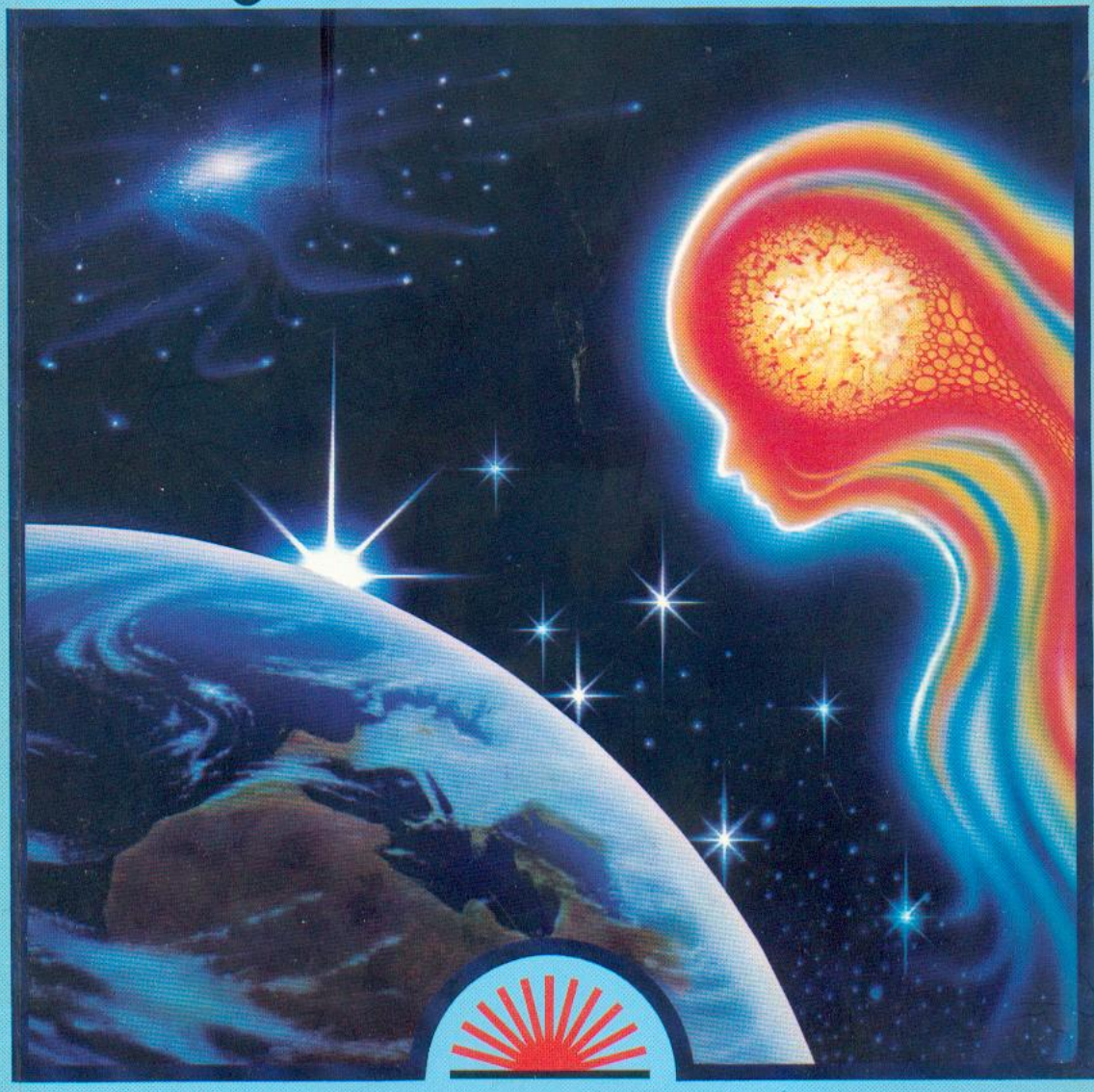
ZX spectrum astronomy

discover the heavens on your computer

maurice gavin



Preface

Astronomy, the study of the heavens, and your Spectrum are just made for each other! The excellent graphic potential of the Spectrum means that dusty subjects like Kepler's or Bode's Laws can be brought to life.

This book emphasises the visual side of computing and astronomy: in doing so it may displease some purists who only wish to see astrocomputing as number crunching to the nth decimal place. However computers are in the business of communication. Just as BASIC is an acceptable computer language, so a graphic display, in addition to pure numbers, is an important aid to the rapid assimilation of facts and concepts.

This book is not specifically directed at astronomers but at Spectrum owners who wish to expand their computing interests into other fields. I've kept explanations of the mathematics or trigonometry used in some of the programs to an absolute minimum. You don't need to know about such subjects to RUN the programs — simply key in and RUN. I've also explained the general working of each program, in full, where appropriate and included relevant information about astronomy.

There is a commonly held impression that everything which happens in science is happening today. In many ways this is a good thing as it means that so many people are interested in what goes on. But astronomy has a long history and standards, formulae and computing devised centuries ago, even before the telescope was invented, are still in common usage. (A classic example would be Hipparchus' stellar magnitude scale — still in use after 2110 years.) Not only is astronomy an ancient science, computing is older than you might think!

We are all prisoners on this beautiful island Earth — a tiny speck sailing through the cosmos. It is hoped that some of the programs included in this book will entice the user to flights of fancy to other worlds and so widen an experience of the mind.

Acknowledgements

Grateful thanks are due to Dr Peter Duffett-Smith for his published algorithms used in the two planetary ephemeris programs and to James Weightman who wrote the bulk of the Jovian satellites program for this book, based on an algorithm by Jean Meeus.

Guidance Notes

The following notes are intended to provide some guidance to the layout of this book, and its relevance to Spectrum users.

16K and 48K Spectrum compatibility

Although the programs in this book were written on my 48K Spectrum, they should RUN on the 16K Spectrum without modification (apart from the Starmaps program in Chapter 8, which will fit on the 48K only). The total memory capacity necessary to RUN the other programs should not exceed about 7K — well within the 16K Spectrum's 9K of usable memory. No machine code routines are included and the few POKEs and PEEKs that are used are compatible with both versions of the Spectrum.

The programs

Most of the LLISTings in this book were prepared via an Epson printer and an RS232 interface to the Spectrum. Only the ASCII character set can normally be committed to a printer (other than the ZX printer) and so the Spectrum's chunky graphics and UDG set CHR$ 128 to CHR$ 164 inclusive are named as CHR$ CODEs in the LLISTings when they are included in programs. This doesn't affect the RUNning of programs and is preferable in many ways — leaving no doubt as to which character is intended.

NB. The Epson printer does not include the hash sign (#), which is represented in the LLISTings by the pound (£) sign, or the copyright sign (©), which is represented by the @ sign.

As far as possible, I have tried to ensure that the LLISTings are identical to the screen LISTings you will see as you key in the programs.

Although it is unlikely that one programmer will completely understand the technique of another, you may find it helpful if I clarify some of the points behind the layout of the program LISTing.

1. The Spectrum screen and the LISTings are limited to a maximum of 32 characters per line, including the line number.

2. Multi-statement lines do not aid legibility.

3. Only 19 characters (within quotation marks) can be used in a single-line PRINT statement, before it overflows on to the next line.

4. REM statements are an important aid in breaking up the program.

I don't subscribe to the commonly-held belief that the quality of a program should be judged by its length. A program should do the most for the least amount of effort, especially if it has to be keyed in. This means that my PRINT commands are sometimes rather curt, and only information which is necessary to RUN the program is contained in the screen display.

Error-trapping

Generally, error-trapping of INPUTs is kept to a minimum: I've included some useful hints in Chapter 10, which you can add if you wish. If a program crashes with a nonsensical INPUT, for example, you usually lose little (although it may be a blow to your pride) and the program can be restarted with a GOTO command (any special conditions are included in the program notes)

ZX printer COPYs

All the programs have been prepared to give a good standard of screen display in both monochrome and colour, so that text and graphics stand out clearly and you can send legible COPYs to the ZX printer (remembering that the printer will only give you black and white COPYs, with no intermediate tones). Also, although I have used an inverse presentation (BORDER 0: PAPER 0: INK 9 – black screen with white or light colours for the display, conjuring up an impression of the night sky) the results when committed to the printer will always come out as black on white, and will look like a negative image of the screen display.

If you do not have a ZX printer, or your printer is not connected to the Spectrum, the program will just skip the COPY, LLIST, LPRINT commands wherever they occur and go on to the next line. If you don't know how to remove the printer commands from the program, it is safer to leave them in.

Amendments

None of the programs in this book is sacrosanct. You should make your own amendments as required, and personalise the finished product. For instance, I have not used BEEP instruction very often. This stems purely from my own association of astronomy with nights enjoyed in blissful silence under a starry sky. You may feel that BEEPs are appropriate to some programs (in addition to those few where I have included them) and you should feel free to add as many BEEPs as you like.

Running Speed

A lot is written in the computing press about the time taken to execute set routines as a test of the running speed of the particular computer. If a program is searching or sorting information, then this can be important. However, none of the programs in this book are of this kind, so the question does not arise. The operating speed of the Spectrum for most tasks in amateur astronomy is more than adequate. As an example, the Planetary Ephemeris program in Chapter 6 takes about 20 seconds to compute and PRINT to the screen all the planetary positions – Mercury through to Pluto – within, generally, a few minutes of arc accuracy for any chosen date. A skilled mathematician would take about 15 to 20 minutes per planet to do this 'by hand'.

From Spectrum to night sky

Some of the programs have an observational bias, and I certainly hope that you will feel encouraged to leave your Spectrum sometimes, and find the stars and planets referred to in the programs, using a pair of binoculars or a telescope. This would make these programs much more rewarding.

You can get a telescope which is quite small and still very effective. The important thing to look out for is not the magnification (which is just a by-product of the optical arrangement) but the size of the aperture of the main collecting lens – or mirror, if it is a reflecting telescope. Anything which improves on the human eye's miserly aperture (7 mm diameter at best, when adapted to the dark) will produce dramatic results. For example, a 60 mm aperture refractor (lens) telescope collects at least 70 times — ie $(60/7)^2$ — more light than the eye, and so makes many faint night-time objects clearly visible.

If you end up hooked on astronomy, the appendix lists some organisations which will be pleased to help you.

Warning! On no account should any telescope or pair of binoculars be pointed at the Sun! Instant and permanent blindness will result. (This applies even if the instrument is fitted with so-called sun-filters, which are not safe – they may shatter under the sun's heat or pass harmful radiation.)

Contents in Detail

- **Kepler's Orbits, demonstrates the first two of Kepler's Laws**
- **Orbit Foci, plotting the second focus**
- **Comet Orbit, eccentric orbits**
- **Halley's Comet, depicts one complete orbit, 1948-2023**
- **Pluto's Orbit, plots the relative positions of Pluto and Neptune from 1880-2128, one complete orbit for Pluto**
- **Solar Apex, corkscrew motion of a planet towards the Solar Apex**.

CHAPTER 6

The Planets

- **Solar System Trek, view the solar system from the skies of any planet on any date**
- **Planetary Ephemeris, trace the planets in the sky for any date**
- **The Moons of Mars, animated presentation of Mars and its moons**
- **Jupiter's Satellites, identify and name the positions of Jupiter's four moons on any date**
- **The Rings of Saturn, simulation of Saturn and its rings**
- **Saturn's Rings, brief outline only**
- **Saturn Draw, a Computer Aided Design**
- **Planets through a Telescope, relative sizes of the planets**
- **Globe-pixel, plots a globe divided at 10° intervals of latitude and longitude**
- **Globe Projection, as before, using lines**.

CHAPTER 7

Star Systems

- **Tri-star Orbits, places two companion stars in orbit around a central star**
- **Binary-star Orbit, a more normal situation, one star orbiting another**
- **Spirals, various options**
- **Galaxy, simulate the Milky Way**.

CHAPTER 8

Starcharts

- **Starmaps, display a simple starmap**
- **Constellation Plot, entering star patterns with CHR$ CODEs**
- **Stellar Magnitudes, plotting differing star brightnesses**
- **Star Graphics, constructing complex shapes**
- **Flashing Stars, two ways to make stars flash**
- **Startrax, watch the changing shapes of the constellations over hundreds of thousands of years**
- **Stellar Magnitude, magnitude ranges of stars.**

CHAPTER 9

Further Programs

- **The Messier List, check out the fake comets**
- **Telescope, the facts about telescopes, binoculars, cameras and astronomy**
- **Star Tester, a quiz**
- **Ellipses, various ellipses**
- **Spectrum World Map, a map of the world in CHR$ CODE.**

CHAPTER 10

Spectrum Hints and Tips

- **Error-trapping of entries**
- **Line renumbering**
- **The ZX printer, Looking after your LLISTings**.

Introduction

About 7 pm on Sunday 25th September 1983, seeking a well earned break from the typewriter and this book, I wondered if the planet Jupiter was visible in the clear dusk skies. The planet was already low in the southwest when I had last seen it some weeks previously, and the situation would not improve. Speed was of the essence. A quick run through of programs (as contained in this book) was called for.

First the tape of the Planetary Ephemeris program was RUN with the INPUT date of ”1983:9:25″. Barely 20 seconds later the computation was complete and a COPY made to the ZX printer of all the planetary positions.

Next the Startracker program was RUN with Jupiter’s location at RA 16h 15m and Dec – 20.8° entered, together with my latitude (+ 51.2° N). The program was stopped during the PLOTting at 18.00hr GMT (equal to 7 pm BST) and another COPY made. This COPY indicated that Jupiter would be at an azimuth bearing of 211 ° (south-southwest direction) and at an altitude of +12° above the horizon. There was still a chance of seeing Jupiter but only in a bright sky before the stars became visible!

Finally the Jupiter’s Satellites program was RUN and the date entered for PLOTting at two-hourly intervals. A COPY was made of this screen display, too. There followed a frantic dash to the bottom of the garden where the telescope was permanently mounted. Luck was on my side! There, nestling in a gap in the rooftops was the planet Jupiter with four attendant moons in a ragged line — all clearly visible through the telescope. Reference to the printout for the Jovian satellites identified them as Europa, Io, Callisto, Jupiter itself and then, on the other side of the planet, a solitary Ganymede. All were in precisely the locations as predicted for that date and hour.

Some 20 minutes later at 18.20 GMT, Jupiter set behind chimney pots, but not before some of the Gavin family were roped in for a spy through the telescope. This shows that, even under pressure, the programs can be quickly chained together to find out the relevant information which makes astronomy more enjoyable. I hope that they add to *your* enjoyment.

Chapter 1 Time (Measuring time, different calendars)

Time and the calendar

Time is not just the ticking of a clock. If time did not exist, then neither would the universe. Time is fundamentally a measure of *change* in the position of an object (ie movement) — be it an electron in orbit about the atomic nucleus or the separation of galaxies since the Big Bang.

Fortunately, time seems so natural it's just like breathing. The ticking clock is a nice homely reminder of passing events measured in seconds, minutes and hours. And the calendar marks some order in events beyond a 24-hour period.

Astronomers use all the familiar systems to measure time, but variations are needed.

The Earth's timekeeping

The Earth, once thought to be a perfect timekeeper in its 24-hour rotation period, has been found to have small 'glitches' that can be detected with modern super-accurate clocks. Not only do we have leap years to correct the calendar (because there are 365.2422 days in a year), but 'leap seconds' to correct the grinding rotation of Earth. Leap seconds are only applied at very infrequent intervals, of perhaps a year or more, and their necessity cannot be predicted with any certainty.

The Earth can be compared to a spinning ice skater. Extending the arms will slow a skater down — the spin is converted into 'angular momentum'. In the case of the Earth, it appears that climatic changes (like a series of North Atlantic depressions over several months) can contribute to slight changes in the Earth's angular movements, resulting in the need for leap seconds. It has not been proved conclusively that this happens, but it is worth remembering that a single summer thunderstorm over Britain releases more energy than would a single H-bomb — such is the power of weather.

The calendar

The 365 days in the year (366 in a leap year) are allocated in seven days to a week, between 28 and 31 days to a month, and twelve months to the year. The word month is probably a corruption of 'moonth' for this is the approximate length of time which the Moon takes to orbit the Earth The four quarters of the Moon, ie between New Moon and First Quarter and so on to Full Moon, Last Quarter and New Moon again, occupy about seven days each.

The Gregorian or Reform Calendar

Julius Caesar is recognised for the introduction of the calendar which allowed for leap years every fourth year, so losing the quarter of a day (about 6 hours) gained each year. This assumed that the year was 365.25 days long whereas it proved to be 0.0078 days (11.23 minutes) less than this. This may seem trivial in a year, but over the centuries it can amount to an error of several days to the start of the year. In the sixteenth century the matter was considered serious.

In October 1582, Pope Gregory introduced the calendar which bears his name, designed to improve on the Julian Calendar. The days of October 5th to October 14th, inclusive, of that year were deleted completely, so that October 4th was immediately followed by October 15th. This brought the calendar back into line with that started by Julius Caesar.

The second measure was to ensure that such a discrepancy did not occur again for several thousand years. If the full century was divisible by 400 then it would be a leap year: otherwise it would not. This means that 1700 1800, and 1900 (leap years in the Julian Calendar) would not be, whereas 1600 and 2000 would remain leap years. Simple but effective. The error of 11.23 minutes is now reduced to about 26 seconds between the civil year (as accepted by governments) and the tropical year, or true year This 26 seconds error can be corrected in the calendar by omitting a leap year every 3334 years but we can let the future generations worry about that! You can see that Pope Gregory (and his calendar advisor Clavius) were far-sighted men.

Because the Gregorian Calendar was instigated by the Vatican, it did not meet with approval from Protestant countries like England. England and the North American colonies did not adopt this Calendar until September 1752. This is why reference is rarely made to the events of October 1582 in American (computing) books. One of the last states to adopt the Gregorian Calendar was the USSR in 1917.

The message here is important for the historian whether of political or astronomical subjects. The calendar used by a witness must be carefully researched before it is possible to know to which date he refers.

With this background information we can now proceed with some programs relevant to the calendar and time.

Chapter 1 – Time (Spectrum Calendar)

Spectrum Calendar

This program PRINTs to the screen a colourful calendar, four months at a time, for any selected year after 1582 – the year the Gregorian or modern Reform Calendar was introduced.

The program

The essential part of the program is Line 10 where the Julian Day number as it is called, is calculated via the variable j(m) for the first day of each month, using the first FOR/NEXT m loop. This is later compared (in Line 130) with the Julian Day number for each day using the variable jx against the Julian Day number for the first day of the next month:

```
130 IF jx = j(m+1) THEN GOTO 190
```

ie start a new month. Thus each month has the correct number of days and full account is taken of leap years (February 29th) at 4, 100 and 400 year intervals.

The brevity of the Julian Day number routine as a simple one-liner is achieved with Bolean Logic in that the portions in brackets are only calculated if true, returning a nil value if false. Line 140 identifies the day of the week, eg Sunday, Monday.

Line 220 is a conditional GOTO, for displaying the remaining two pages of the calendar starting with May and September. It is important that the rather odd Lines 311 and 313 are not amended, otherwise the program will not proceed beyond the first page covering January to April. The value of the second FOR/NEXT m loop (Line 60) is upgraded via the variable a in Line 311. The conditional GOTO in Line 220 repeats this m loop until m=12+1 (rather strangely it is always + 1 more than the current value of the loop) for December so the program will GOTO 313 (300+13) and is reRUN.

The REM statements identify the general structure of the program. Most of them are concerned with the complex screen display and formatting Each Julian Day number is shown briefly on the bottom line of the screen and, as the Julian Day begins at noon (this will be explained in the Julian Date program later in this chapter) a value of 0.5 appears in each number See Figure 1.1 for a sample page from the Spectrum calendar.

The program contains a couple of useful tips. Firstly, the use of the n$ at the beginning of the program to serve the dual purpose of titling the LLISTing and the screen display. Secondly, reserving space via DIM a$ (30) for use in Line 40:

```
40 PRINT… d$ + a$ + d$….
```

instead of the more usual

PRINT… d$ + ″ 30 blank spaces ″ + d$

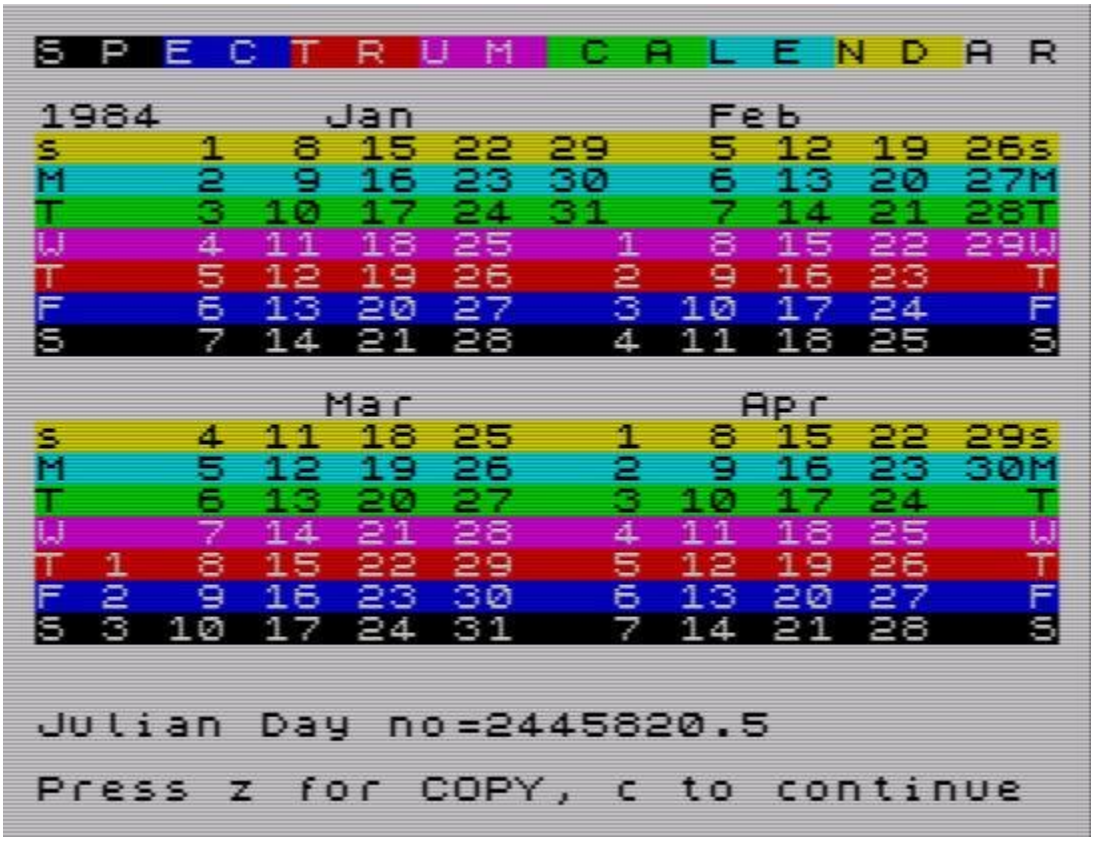
where the d$ contains the initial letters of the days of the week from character 37 onwards. (See Line 6 for the complete contents of the d$.) The rest of the d$ holds the month names.

Line 210 contains an option to COPY the screen display one page at a time via the INKEY$ command.

```
3 LET n$="SPECTRUMCALENDAR"
5 DIM a$(30): DIM j(12): DIM e$(12,3): LET a=1
6 LET d$="JanFebMarAprMayJunJulAugSepOctNovDecsMTWTFS"
7 INPUT "Year=";y: IF y<1583 THEN GO TO 7
8 LET yy=y/100: LET y1=INT yy: LET yt=y/400: LET y4=INT yt
10 FOR m=1 TO 12: LET j(m)=INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))-
y1+y4+1720996.5+1+(1 AND yy=y1 AND yty4 AND m<3)
12 LET e$(m)=d$(m*3-2 TO m*3)
15 NEXT m
20 CLS : LET z=1: LET zz=0
30 FOR n=1 TO 16: PRINT PAPER n/2-1; INK 9;(" " AND n=9)+n$(n)+" ";: NEXT n
40 PRINT 'y: FOR f=0 TO 1: FOR n=1 TO 7: PRINT PAPER 7-n; INK
9;d$(36+n)+a$+d$(36+n): NEXT n: PRINT ": NEXT f
50 PRINT AT 2,9;e$(a);AT 2,21;e$(a+1);AT 11,9;e$(a+2);AT 11,22;e$(a+3)
60 FOR m=a TO a+3: LET xx=1
80 IF m=3 OR m=4 OR m=7 OR m=8 OR m=11 OR m=12 THEN LET xx=10: LET zz=0
90 IF m=3 OR m=7 OR m=11 THEN LET z=1
100 FOR d=1 TO 31
110 LET jx=j(m)+d-1
120 IF m=12 THEN GO TO 140
130 IF jx=j(m+1) THEN GO TO 190
140 LET q=INT (jx-5)-7*INT ((jx-5)/7)+1
150 LET x=q+1: IF q=1 THEN LET x=x-q+1: LET z=z+3
160 PRINT AT x+xx,z+zz; PAPER 7-q;(" " AND m/2=INT (m/2))+(" " AND d<10); INK 9;d
170 PRINT AT 21,0;"Julian Day no=";jx
180 NEXT d
190 NEXT m
200 PRINT #0;"Press z for COPY, c to continue": PAUSE 0
210 IF INKEY$="z" THEN COPY
220 GO TO 300+m
310 LET a=a+4: GO TO 20
313 RUN
```

Figure 1.1

Sample page from calendar.



Julian Date

Astronomers are naturally interested in events far removed from Earth which sometimes have a periodicity quite unrelated to our calendar. Timing these events by quoting the date when they occur would be tiresome, so astronomers resort to a very simple counting system called the Julian Day number or JD for short.

The reference date is noon (GMT) January 1st 4713BC. Every date has a separate JD beginning at noon – currently this is a seven-figure number in excess of 2,400,000. That particular JD was passed on November 17th 1858 and is sometimes used as a reference date called the Modified Julian Day number (MJD).

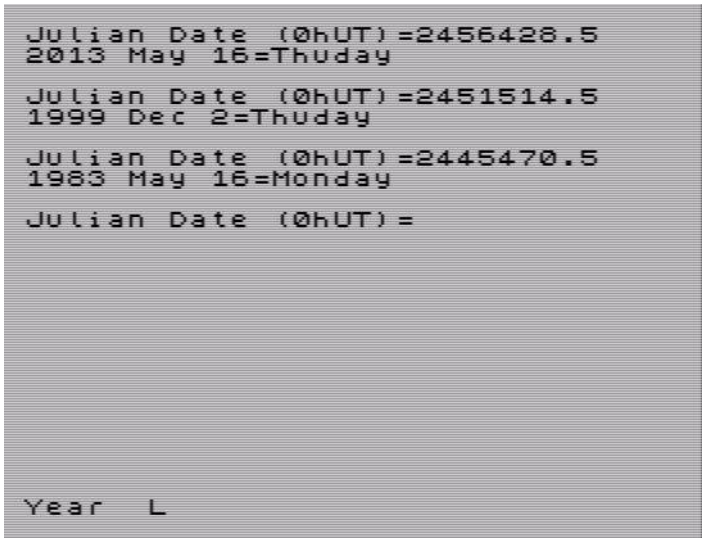
The following routine calculates the JD for any INPUT date after October 1582 – the month the current (Gregorian) calendar was introduced. Decimal days can be included in the JD so that an event timed even to a fraction of a second on a particular day can be written as a single, if long, number. For example:

JD no 2445470.501 = 1983 May 16 (Monday) 0h 01m 26.4s (GMT).
Eg 1 minute and 26.4 seconds after midnight on Monday 16th May 1983 (GMT).

It can be seen that, for computing purposes, the JD is much more convenient than handling dates from the regular calendar.

```
10 PRINT "Julian Date (0hUT)=";
30 LET d$="SatSunMonTueWedThuFriJanFebMarAprMayJunJulAugSepOctNovDec"
40 INPUT "Year  ";y'"Month ";m'"Day   ";d
50 LET m$=d$(m*3+19 TO m*3+21)
100 GO SUB 1000: LET j=jj+d
160 LET q=j-7*INT (j/7)+1
170 LET e$=d$(3*q-2 TO 3*q)
200 PRINT j+1720996.5
300 PRINT y;" ";m$;" ";d;"=";e$;"day": PRINT : GO TO 1
1010 LET yy=y/100: LET y1=INT yy: LET yt=y/400: LET y4=INT yt
1030 LET jj=INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))-y1+y4
1040 IF yy=y1 AND yt<>y4 AND m<3 THEN LET jj=jj+1
1050 RETURN
9900 REM ************************************************
9990 SAVE "Jcal"
```

Image taken from the above program showing Julian dates for 16/05/2013, 02/12/1999 and 16/05/1983

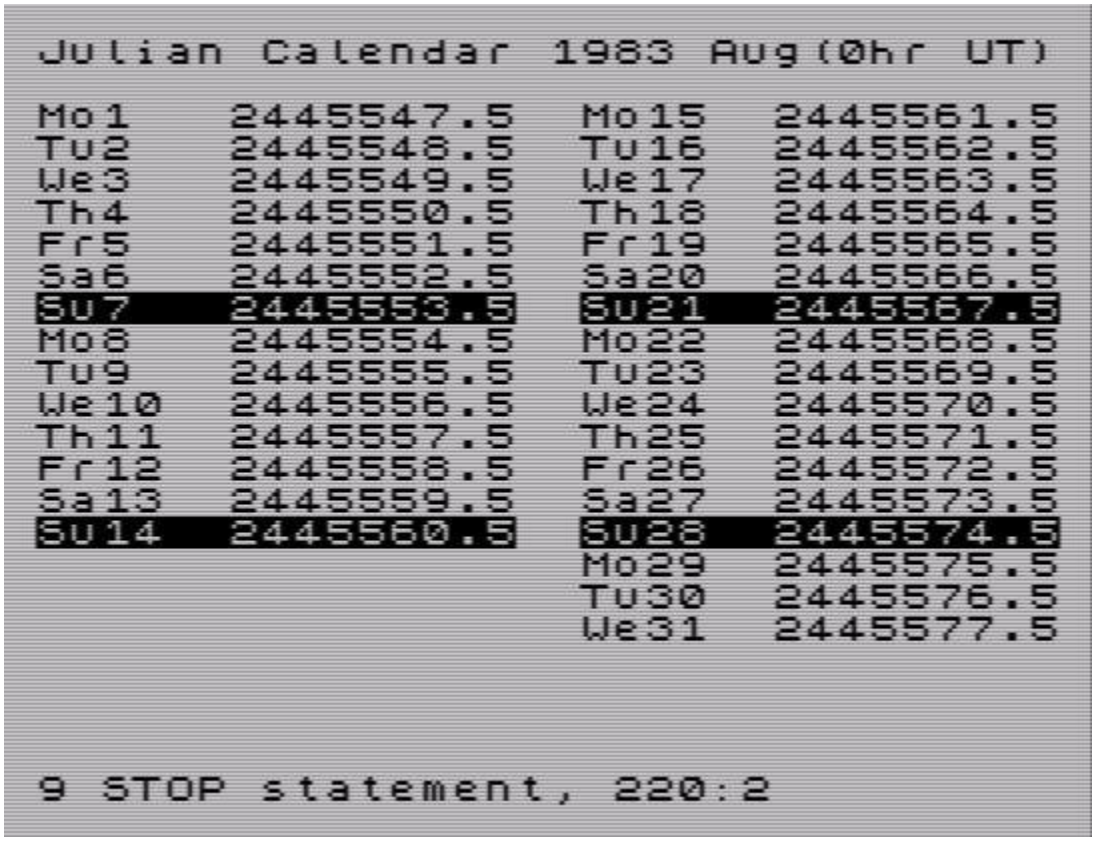


Julian Calendar

This program is a variation of the Julian Date program and PRINTs to the screen a complete Julian Day Calendar for any month of any year after October 1582 It also identifies the day of the week – highlighting the Sundays in INVERSE. See Figure 1.2.

Figure 1.2

Julian Day Calendar, with Julian Day number for each day.



The program accounts for the actual number of days in the selected month. The JD is PRINTed in full and is timed at OhrUT (GMT) ie from midnight. As the JD begins at noon (of the previous day) a value of 0.5 is included in the answer. This represents the 12 hours between noon and midnight – the latter marking the start of a civil day.

```
9 REM ************************
10 CLS : PRINT "Julian Calendar ";
11 REM ************************
30 LET d$="TuWeThFrSaSuMoJanFebMarAprMayJunJulAugSepOctNovDec"
40 INPUT "Year ";y,"Month ";m
50 LET m$=d$(m*3+12 TO m*3+14)
60 PRINT y;" ";m$;"(0hr UT)"
100 GO SUB 1000
```

```
110 LET x=0: LET a=0
115 FOR d=1 TO 31: LET j=jj+d
120 IF d>14 THEN LET x=14: LET a=3
150 IF j=j1+1 THEN STOP
160 LET q=j-7*INT (j/7)+1
170 LET e$=d$(2*q-1 TO 2*q)
180 INVERSE 0
190 IF e$="Su" THEN INVERSE 1
200 PRINT AT 1+d-x,x+a;e$;d;
210 PRINT TAB x+6+a;j+.5
220 NEXT d: STOP
1000 LET ed=1720996: LET m1=m+1
1010 LET yy=y/100: LET y1=INT yy: LET yt=y/400: LET y4=INT yt
1020 IF yy=y1 AND yt<>y4 AND m<3 THEN LET ed=ed+1
1030 LET jj=ed+INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))-y1+y4
1040 IF yy=y1 AND yt<>y4 AND m1=3 THEN LET ed=ed-1
1050 LET j1=ed+INT (365.25*(y-(m1<3)))+INT (30.6001*(m1+1+12*(m1<3)))-y1+y4
1060 RETURN
9900 REM ***********************
9990 SAVE "Jcal"
```

Day of the Week

On what day of the week were you born?

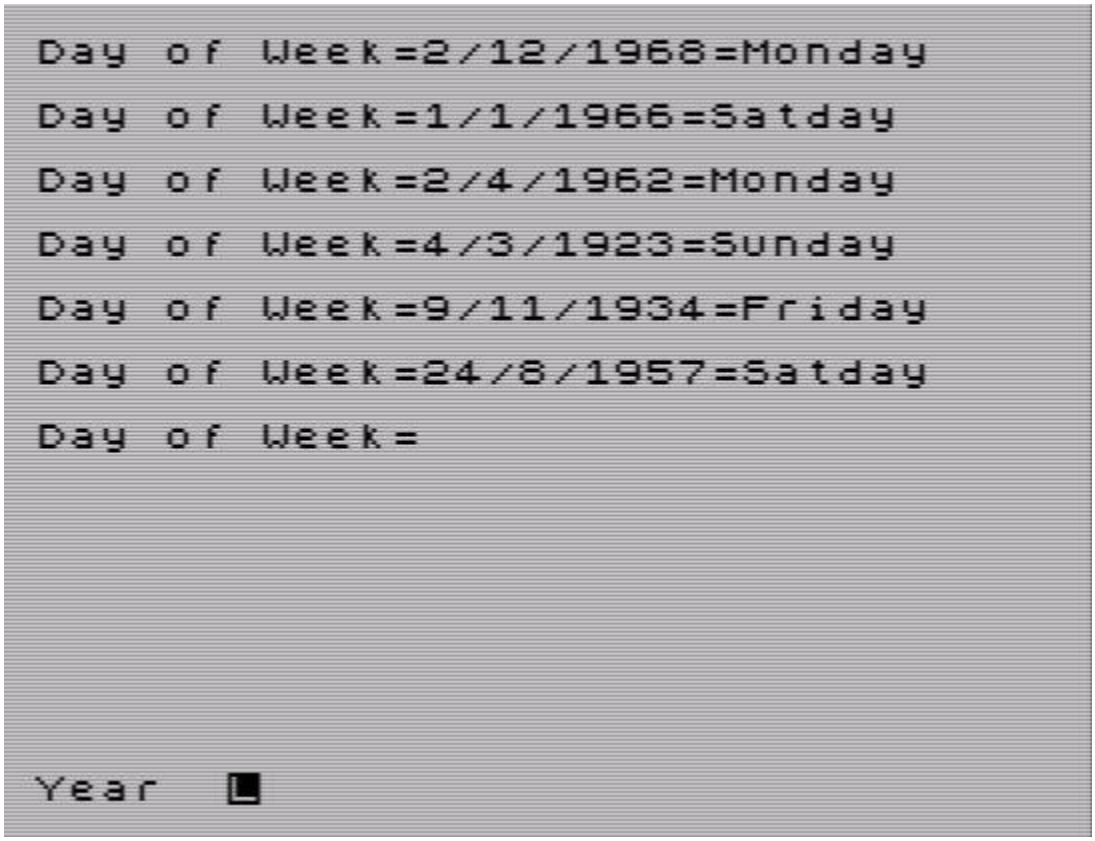
When will Christmas Day next fall on a Thursday?

This simple program answers such problems by identifying any INPUT date against the day of the week – Sunday, Monday, etc. It is; ideal for incorporating into a longer program as a subroutine where the date is included.

```
9 REM ***********************
10 PRINT "Day of Week=";
11 REM ***********************
30 LET d$="SatSunMonTueWedThuFri"
40 INPUT "Year ";y'"Month ";m'"Day ";d: GO SUB 1000
160 LET q=j-7*INT (j/7)+1
170 LET e$=d$(3*q-2 TO 3*q)
300 PRINT d;"/";m;"/";y;"=";e$;"day"": GO TO 1
```

```
1010 LET yy=y/100: LET y1=INT yy: LET yt=y/400: LET y4=INT yt
1030 LET j=d+INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))-y1+y4+(yy=y1 AND
yt<>y4 AND m<3): RETURN
9900 REM ***********************
9990 SAVE "DoW"
```

Sample output from the Day of the Week program



Interval Days

This program computes the precise interval in days, hours and minutes between two INPUT dates which may be between a few minutes to centuries apart. The important part of the program is from Line 10, where an MJD is worked out for each INPUT date. Each date has a unique JD and full account is taken of leap years (February 29) at 4, 100 and 400 year intervals. If an invalid date is entered this is corrected to the true date, for the purposes of computing the interval.

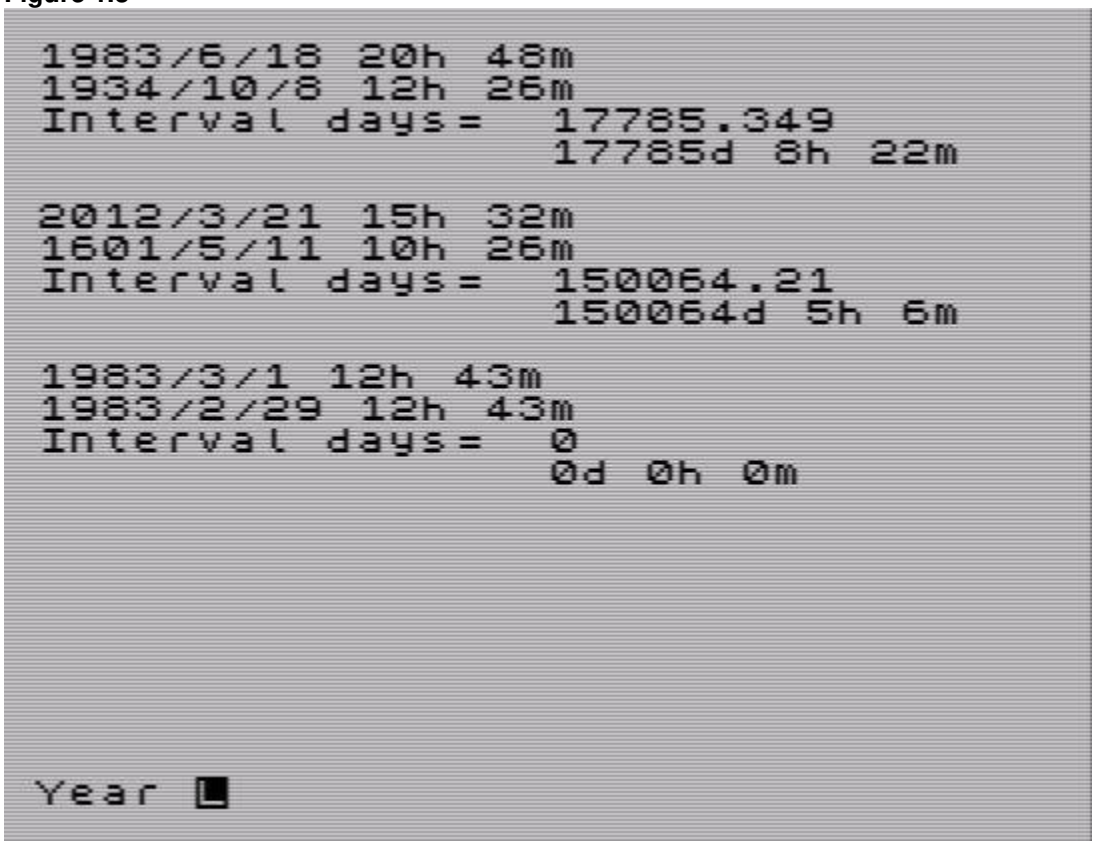
Example

a) 1983 Feb 33 (invalid) = 1983 Mar 5 (valid)

b) 1984 Feb 33 (invalid) = 1984 Mar 4 (valid)

The two INPUT dates may be entered in either order. Line 70 ensures that the result is always positive. The month should be entered numerically – January = 1, February – 2, etc. Figure 1.3 shows typical results.

Figure 1.3




```
9 REM ***********************
10 REM *****Interval Days*****
11 REM ***********************
20 DIM j(2): FOR x=1 TO 2
40 INPUT "Year ";y;" Month ";m;" Day ";d,"Hour (0-23) ";h;" min ";mn
50 PRINT y;"/";m;"/";d;" ";h;"h ";mn;"m": LET d=d+h/24+mn/1400
60 GO SUB 1000: NEXT x
70 LET day=ABS (j(1)-j(2))
80 PRINT "Interval days=",day
90 LET hour=24*(day-INT day): LET min=60*(hour-INT hour)
100 PRINT ,INT day;"d ";INT hour;"h ";INT min;"m"": GO TO 1
1010 LET yy=y/100: LET y1=INT yy: LET yt=y/400: LET y4=INT yt
```

```
1030 LET j(x)=d+INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))-y1+y4+(yy=y1
AND yty4 AND m<3): RETURN
9900 REM ************************
9990 SAVE "intdays"
```

Local Sidereal Time

Our household clocks, which use a 24-hour period (or two 12-hour periods), reflect the daily passage of the Sun across the sky. The Sun is approximately due south at noon (approximately because the Earth's orbit about the Sun is not a perfect circle but slightly elliptical). This means that the Earth's orbital velocity will vary with the seasons whereas the Earth's rotation on its axis is almost constant. The net result is the Sun gets out of synchronisation by plus or minus about 15 minutes from its noonday passage of the southern meridian.

The stars, in contrast, keep virtually perfect time. Because they are so remote, the Earth's varying orbital velocity is of no consequence. The stars return to the same point in the sky (for any fixed location) four minutes earlier each day whether daylight blots them out or not. Sidereal time or star time is based, therefore, on a clock that runs four minutes fast on the household clock.

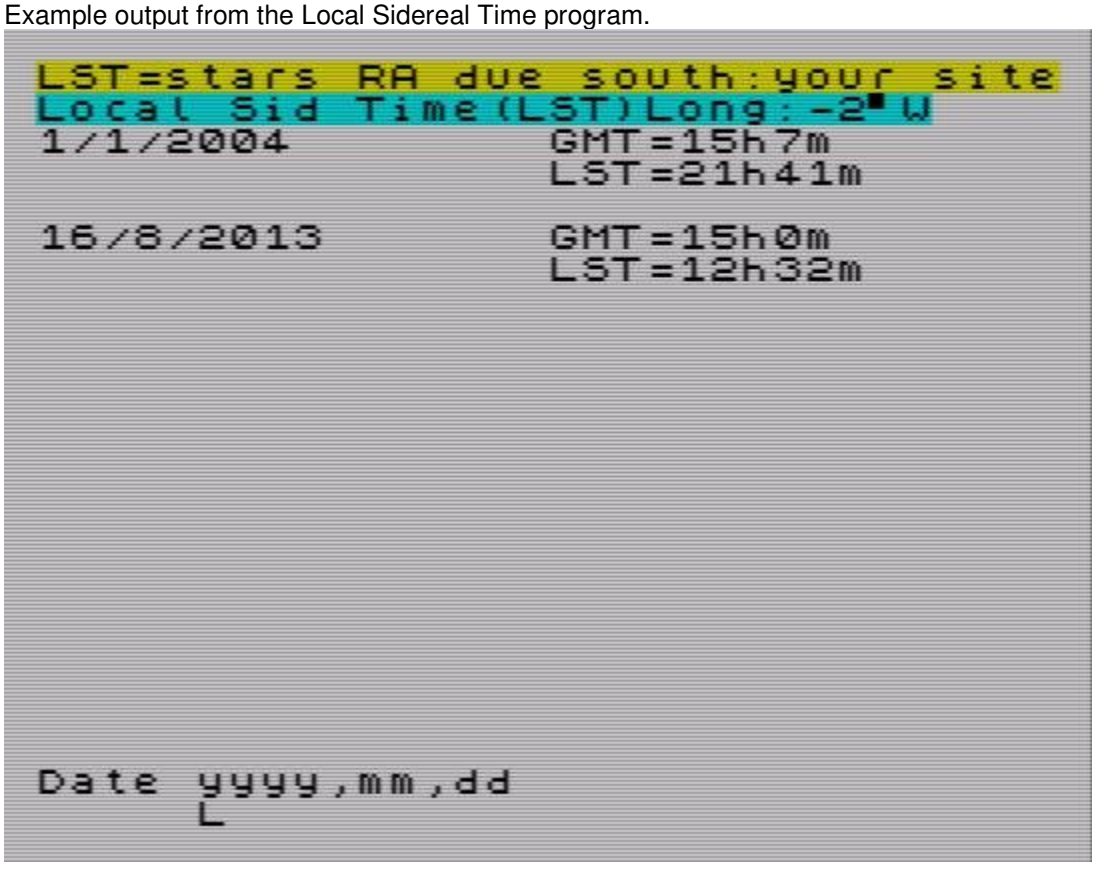
Knowing the sidereal time for any date and hour is important for astronomers so that they can plan ahead for their viewing sessions (weather permitting). The best viewing occurs when the region of interest of the sky is to the south of the observer in the northern hemisphere (to the north in the southern hemisphere) and at the greatest altitude. (Twinkling stars caused by a disturbed atmosphere are less prevalent away from the horizon.)

This short program calculates the (local) sidereal time against the prompted INPUTs. For example, if the sidereal time is conputed as 6hr 44m then Sirius – the brightest star in the sky – will be due south as this time coincides with the Right Ascension (explained in the next chapter) for Sirius.

```
9 REM **********************
10 REM Local Sidereal * Time
11 REM **********************
15 PRINT PAPER 6;"LST=stars RA due south:your site"
20 INPUT "Your longitude lll.l:Greenwich=0-(west)+(east) "; LINE l$: LET l=VAL l$
30 IF l$(1)"-" AND l$(1)"+" OR ABS l>180 THEN GO TO 20
40 PRINT PAPER 5;"Local Sid Time(LST)Long:";l$;CHR$ 130;
50 PRINT PAPER 5;("W" AND l$(1)="-")+("E" AND l$(1)="+")
60 INPUT "Date yyyy,mm,dd";TAB 5;y;TAB 10;mm;TAB 13;d
70 IF mm>12 OR d31 THEN GO TO 60
75 PRINT d;"/";mm;"/";y,
80 INPUT "GMT/UT:hh.mm";TAB 7; LINE e$: LET e=VAL e$
```

```
85 IF e>24 THEN GO TO 80
90 PRINT "GMT=";: LET t=INT e+((e-INT e)/60*100)
100 LET m=mm: IF m>2 THEN LET m=m+1: GO TO 120
110 LET y=y-1: LET m=m+13
120 LET j=INT (365.25*y)+INT (30.6001*m)+d+1720982
130 LET g=6.63627+6.570982e-2*(j-2443144)
140 LET ts=g-INT (g/24)*24
150 LET s=l/15+t+ts+t/1436*4
160 IF s>24 THEN LET s=s-24
170 IF s<0 THEN LET s=s+24
180 LET st=INT (s*100)/100
200 LET h=t: GO SUB 300: PRINT ',"LST=";: LET h=st: GO SUB 300
210 PRINT : PRINT : GO TO 60
300 PRINT INT h;"h";INT ((h-INT h)*60+.5);"m ",: RETURN
9900 REM ************************
9990 SAVE "LSTime"
```

Example output from the Local Sidereal Time program.



Reaction Timer

It is appropriate to include this program under the section dealing with time although it very much relates to the rather imprecise impression our brain has of short intervals of time. Because the human eye is readily available as a superbly sensitive detector at the telescope (and photo-detectors are expensive, complex and generally unavailable) the human eye continues to hold sway.

Occultations – timing the disappearance or reappearance of a star trom behind the Moon or an asteroid – can yield accurate results of scientific value about the motion or diameter of that body. This work is almost exclusively done by amateur astronomers using small telescopes. It is, however, important to know what is called the 'personal equation' of each observer (the observer's accuracy) to adjust his results – hopefully to 0.1 second accuracy.

The program

This program uses the Spectrum's 50 Hertz 'clock' – started with POKE 23672,0 in Line 45 — to test your reaction times to a series of mock occultations of both disappearances and reappearances. Each result is revealed audibly, numerically and visually (via a coloured bar-graph). Each event is started with a random PAUSE in Line 30 to avoid complete anticipation. If you cheat – by pressing key 'p' prematurely – or are so slow as to take longer than 0.44 seconds, you get a warning, and the result is disregarded. The Spectrum ROM will also try to cheat by skipping lines if the PAUSE 1 in Line 21 is omitted, but this is not added to your result.

The essential part of the program is in Lines 45 and 100 — which, respectively POKE the timer to start and PEEK the answer. By limiting the instructions between these two to an INKEY$ loop, <0.02 seconds (to process the data) is added to your reaction time. Your results may be 10 to 20 times slower than this delay. Figure 1.4 shows a typical result.

Figure 1.4

Typical results for one individual's reaction times. The brain invariably proves more sluggish on reappearances (0.316 seconds on average) in comparison with disappearances (0.336 seconds average), as this sample demonstrates.

```
2 REM **********************
3 PRINT "Reaction Timer"
4 PRINT '"Personal Equation :10 simulatedoccultations."
5 FOR i=1 TO 7: PAUSE 20: PRINT AT 9,0; INK i;" 5 reappearances";AT 11,0; INK 8-
i;"then 5 disappearances": NEXT i: PAUSE 200
6 PRINT ""If your time >.44sec or =0sec the result is discounted."
7 PRINT #0; FLASH 1;" Press any key to start…": PAUSE 0
8 BORDER 0: PAPER 0: INK 7: CLS : LET tt=0
10 PRINT "Press 'p' as soon as the 'star' appears/disappears"
12 PAUSE 200: PRINT AT 9,0; FLASH 1;"seconds"
15 LET d=0: LET x=0: FOR z=0 TO 1: LET tt=0: FOR f=1 TO 5
20 PRINT AT 7,14;"_ _"
21 PAUSE 1
25 IF x=0 THEN GO TO 30
26 PLOT 127,112
30 PAUSE 200+RND*300
40 PLOT OVER d;127,112
45 POKE 23672,0
50 IF INKEY$="p" THEN GO TO 100
60 GO TO 50
100 LET t=(PEEK 23672)/50
110 PRINT AT 10+x+f,0;f;" ";
120 IF t=0 OR t>.44 THEN PRINT "Too fast/slow": LET q=3+t*20: GO SUB 260: PRINT AT
10+f+x,0;"Ready ": GO TO 20
130 PRINT t
140 FOR n=9 TO t*50+9
150 PRINT AT f+x+10,n; INK t*20-3;CHR$ 140: NEXT n: LET q=20*t: GO SUB 260
160 INPUT FLASH 1;"Press ENTER", LINE z$
170 LET tt=tt+t: NEXT f
180 PRINT "Your PE ="; FLASH 1;tt/5; FLASH 0;("(re" AND x=0)+("(dis" AND
x0)+"appearance)"
190 IF x=0 THEN LET q=3: GO SUB 260: INPUT FLASH 1;"Now disappearances-press
ENTER"; LINE z$
200 LET d=1: LET x=6: NEXT z
210 PRINT "Z for printout : R to RUN again"
220 LET q=1: GO SUB 260
230 IF INKEY$="" THEN GO TO 230
240 IF INKEY$="z" THEN COPY
250 RUN
```

```
260 FOR v=-30 TO 40 STEP q: BEEP .01*q,v: NEXT v: RETURN
9900 REM ************************
9990 SAVE "speed"
```

Chapter 2 – Spheres Within Spheres (The Celestial Sphere and Coordinate Positions)

The celestial sphere and coordinate positions

Just as your Spectrum uses a two-dimensional coordinate system to PRINT and PLOT on the screen, so astronomers use a virtually identical system to represent the skies above our heads. But, whereas the Spectrum TV display is effectively small and flat, the skies have a complete 'wraparound' through 360°, both horizontally and vertically. One hemisphere is above our heads and one below our feet making a complete sphere.

The computation of positions on this sphere is relatively complex and requires almost constant use of trigonometry. It is beyond the scope of this book to explain the mathematics involved: in fact, this is deliberately excluded, so that you can simply key in the programs and get results without any knowledge of mathematics or trigonometry.

There are a number of coordinate systems which are used by astronomers. One we can all relate to concerns the relationship between our local horizon and the stars as they pass across the sky. This involves the conversion of, say, a star position from one system to another. A couple of experimental programs follow (experimental in their means of presentation) to explain this relationship. If you would prefer a written explanation first, go to the section headed '**The celestial sphere explained**' and return to the programs afterwards.

The Celestial Sphere – 3D Images

The celestial sphere, a device beloved by astronomers, is difficult to depict on a flat page so two separate attempts will be made to reconstruct a three-dimensional image via the Spectrum. Do not be too disappointed if the illusion does not work for you — it is only an experiment. I've tested both of these systems with reasonable success but some factors may be beyond your control.

The first system involves DRAWing two separate images on the TV screen and viewing the result through coloured filters. A colour TV must be used for the display and it may be necessary to adjust the contrast and colour controls for the best effect.

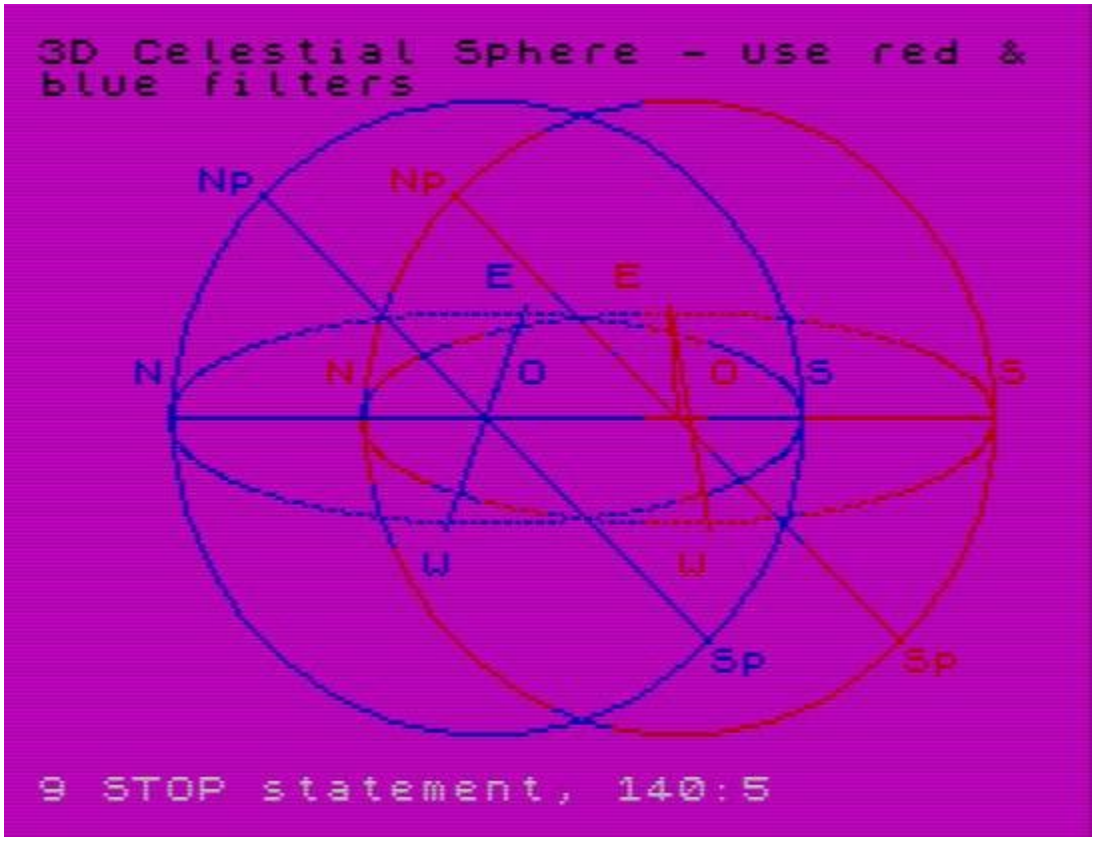
Figure 2.1 shows a COPY from the screen after RUNning this program. The COPY is obviously in monochrome and what it does not reveal is that the screen display uses two INK colours on a single overall PAPER colour.

The two overlapping images are PLOTted in separate INK colours The right hand image is in red (INK 2) and the left hand image in blue (INK 1). It will be noted that, with the exception of the east/west line and the letters E and

W, both images are identical but displaced one from the other horizontally. The background colour is magenta (PAPER 3). These three colours have been chosen with considerable care and should not be amended until the experiment is completed as programmed.

Figure 2.1

This display appears in 3D, when viewed through red and blue filters on a colour TV The program introduces the necessary colour into the two images.



Viewing the results

You will need two deeply-coloured filters – one red and one blue – with which to view the TV display. It is important to stress that these should be *deeply* coloured. Pale tinted filters are useless in this application. In the test, the filter used should render its own coloured image on the TV screen virtually invisible, whilst the contrasting coloured image appears boldly: ie the blue filter should show up the red TV image and vice versa. The filters can be made of clear coloured cellophane or acetate sheet – in layers if need be to get the right depth of colour. A square 5 x 5cm is about the right size for each filter.

The screen results should be viewed from a distance of between four and eight feet (depending on the size of the TV screen) with the red filter held over the right eye and the blue filter over the left eye. The two separate coloured images should appear fused into one about midway between the TV and your eyes. A little patience may be necessary before the brain accepts the two discordantly coloured images as one.

The whole image, with the exception of the oval horizon line, the E/W line and the E and W markers, should appear on one plane. The horizon line and the E/W line should appear to project both towards and away from you with reference to this plane. Similarly, the letter W should appear close to you and the letter E further away. You will notice that the program title remains firmly fixed on the TV screen — this is because its colour is black (INK 0). Because black is in reality colourless, the filters have no effect on this portion of the display and so the 3D effect is absent.

The limitations

Despite the relative complexity of the screen COPY, the screen display has to be kept simple with as few as possible of the lines of the two coloured images intersecting.

This is because the Spectrum display will only support one INK colour per character square. When a second INK colour is overlaid, that portion of the design initially coloured red (INK 2) will be rendered blue (INK 1) if it falls in the same character square. This means that some parts of the design become visible with the wrong filter and the 3D image becomes fragmented.

To minimise this effect, the red image is PLOTted first: when viewed through the blue filter, this image plays a secondary role by only appearing faintly. The main image (in blue) appears boldly with as few spurious lines as possible from the red image to cause distraction. Despite the faintness of the red righthand image it is sufficiently effective when coupled with the blue lefthand image to give the illusion of 3D.

All colour TVs produce tints (colour plus white) and shades (colour plus black) from a blend of three colours — red, blue and green — via the three colour 'guns' in the TV tube. To ensure the maximum colour separation of the 3D images, only primary colours are used. The background colour should be passed by both colour filters equally to minimise eye strain. Hence the selection of colours (blue (INK 1) and red (INK 2) with a background colour of magenta — ie blue plus red — using PAPER 3).

The program

The program is quite straightforward and has sufficient REM statements for guidance — just enter, RUN it and view the results. The two coloured images are drawn between Lines 60 and 140, the variables a and al giving the necessary separation using the same FOR/NEXT f loop.

```
9 REM ************************
10 REM 3D Celestial Sphere - TV
11 REM REM ************************
20 BORDER 3: PAPER 3: INK 0: CLS
30 PRINT "3D Celestial Sphere - use red & blue filters"
40 LET a=112: LET a1=160: LET b=80: LET c=79
50 REM ************************
60 INK 2: REM Red LH image
```

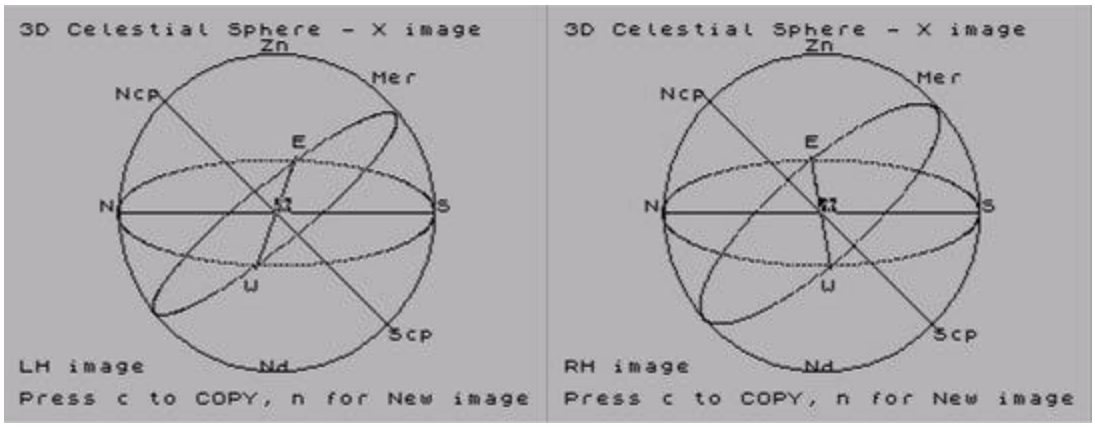
```
70 REM **********************
80 CIRCLE a1,b,c
90 PLOT a1-c,b: DRAW c*2,0
100 PLOT a1,b: DRAW -60,60: DRAW 120,-120
110 FOR f=0 TO PI*2 STEP .02
120 LET x=SIN f*c: LET y=COS f*c/3: PLOT a1+x,b+y: NEXT f
140 IF a=a1 THEN GO SUB 220: INK 0: PAPER 7: STOP
150 REM **********************
160 INK 1: REM Blue LH image
170 REM **********************
180 LET a1=a: GO TO 80
200 REM Blue E/W line+letters
220 PLOT a1,b: DRAW 10,28: DRAW -20,-56
230 PRINT AT 7,14;"E";AT 16,12;"W";AT 10,3;"N";AT 10,15;"O";AT 10,24;"S"
240 PRINT AT 4,5;"Np";AT 19,21;"Sp"
250 REM **********************
260 REM Red E/W line+letters
270 REM **********************
280 INK 2: PLOT 160,b: DRAW -2,28: DRAW 10,-56
290 PRINT AT 7,18;"E";AT 16,20;"W";AT 10,9;"N";AT 10,21;"O";AT 10,30;"S"
300 PRINT AT 4,11;"Np";AT 19,27;"Sp"
310 RETURN
9900 REM **********************
9990 SAVE "3D sphere"
```

Celestial Sphere – X-eyed 3D

This second program for producing a 3D image – again of the celestial sphere – uses an entirely different technique. A colour TV plays no specific part in it but the ZX printer does. If a ZX printer is not available, then simply use the printed screen COPY in Figure 2.2 to experiment with.

Figure 2.2 3D Celestial Spheres

Hold the book at reading distance or a little further away and cross your eyes while viewing. The two seperate images should fuse into one, giving a three dimensional picture with depth.



The system involves preparing two separate DRAWings, from two separate viewpoints, of an object appearing to possess depth. When these drawings are placed side by side and viewed by crossing the eyes (hence X-eyed) the two images can be fused into one, producing a striking 3D illusion. It is estimated that about one third of the population can do this trick without undue eyestrain and I hope that around the same percentage of Spectrum users will be able to do it!

As this is the only 3D system not requiring any viewing aid, its inclusion in this book seems justified. The system has the merit (over the TV/colour filter system just discussed) that the viewing material can be of considerable complexity — the problem of overlapping INK images does not arise.

The two COPYs produced via the ZX printer and this program should be laid out with some care (use Figure 2.2 for guidance). The notation in the bottom left corner of each COPY shows the relative position of each drawing. The centres of the two CIRCLES should be between 70mm and 100mm apart — the precise dimension is not critical. What is vital is that the N/S lines running horizontally through each drawing should be parallel. Use a rule to check that this is so before the COPY is firmly fixed to heavy duty white paper with spray mountant.

Viewing the image

The mounted COPYs for viewing should be placed at normal reading distance (or a little further away in a good light) without any shadows on them The COPYs should be looked at squarely — neither tilted nor rotated. Cross your eyes, perhaps using a finger briefly midway between eye and paper and the two images should fuse into one. The system does require that the eyes converge on a midway point but are focused on the drawings. Results are easier to obtain if you don't over-concentrate. Keep viewing periods brief.

The program

The program is a modification of the 3D Celestial Sphere program. Even if a ZX printer is not available it is well worth keying in – even on the screen, the drawings have a 3D quality as they are PLOTted. If you intend to modify the previous program to produce this version then first delete Lines 20 and 30 and add as a direct command:

```
BORDER 7: PAPER 7; INK 9: CLS: LIST ENTER
```

This will clear the distracting magenta PAPER colour which was essential for the previous background colour.

A new Line 20 is entered, containing the DATA for a little figurine POKEd in UDG CHR$ 154. This is FLASHed in the centre of the screen (instead of a letter O) and represents our observer in the middle of the Celestial Sphere. Two ellipses are PLOTted in this program — to represent the horizon and the celestial equator. The latter comes in two versions according to the INPUT in Line 275, ie a righthand (RH) or lefthand (LH) image, and is inclined at 45° to the horizontal. The variables t (tilt) and zx (semi-major axis) control the shape of the ellipse and variable z is fixed for the inclination. Line 500 allows for a COPY to be made or a reRUN of the program for the second COPY via a one-touch INKEY$ command.

```
9 REM **********************
10 REM 3D Celestial Sphere - X
11 REM **********************
20 DATA 24,8,62,93,157,20,20,54: FOR f=0 TO 7: READ a: POKE USR CHR$ 154+f,a: NEXT
f
30 PRINT "3D Celestial Sphere - X image"
35 PRINT AT 1,15;"Zn";AT 21,15;"Nd";AT 11,5;"N";AT 11,26;"S";AT 11,16; FLASH 1;
OVER 1;CHR$ 154
40 PRINT AT 4,6;"Ncp";AT 3,22;"Mer";AT 19,23;"Scp"
45 PRINT #0; FLASH 1;"Plot horizon"
50 REM **********************
70 LET a1=128: LET b=80: LET c=79
80 CIRCLE a1,b,c
90 PLOT a1-c,b: DRAW c*2,0
100 PLOT a1,b: DRAW -60,60: DRAW 120,-120
110 FOR f=0 TO PI*2 STEP .02
120 LET x=SIN f*c: LET y=COS f*c/3: PLOT a1+x,b+y
125 NEXT f
259 REM **********************
260 REM input RH or LH image
261 REM **********************
275 INPUT "RH or LH image (r/l)? "; LINE c$
276 PRINT AT 21,0;("RH" AND c$="r")+("LH" AND c$"r")+" image"
280 IF c$"r" THEN PLOT a1,b: DRAW 10,28: DRAW -20,-56: PRINT AT 7,17;"E";AT
16,14;"W"
285 IF c$="r" THEN PLOT a1,b: DRAW -5,28: DRAW 10,-56: PRINT AT 7,15;"E";AT
16,16;"W"
300 PRINT #0; FLASH 1;"Plot star on celestial equator"
400 LET t=18: LET z=50: LET zx=61
410 IF c$="r" THEN LET t=30: LET zx=60
420 LET z=1/SIN ((.1+z)/180*PI)
```

```
421 LET t=1/SIN ((.1+t)/180*PI)
430 FOR n=0 TO PI*2 STEP .02
440 LET aa=SIN n*zx
450 LET bb=COS n*zx/t+aa/z
460 PLOT INT (a1+aa),INT (b+bb)
470 NEXT n: INPUT ""
499 REM **********************
500 PRINT #0;"Press c to COPY, n for New image": PAUSE 0
510 IF INKEY$="c" THEN COPY : INPUT "": GO TO 500
520 RUN
9900 REM **********************
9990 SAVE "3Dsphxx"
```

The celestial sphere explained

Letter O (our figurine in the previous program) represents an observer in the middle of a sphere. This observer stands on a horizontal plane at the intersection of the N/S and E/W lines. The point above his head is called the zenith (Zn) and below his feet the nadir (Nd). Through him passes a second but vertical plane, called the meridian (Mer), marked N, Zn, S, Nd etc.

The observer can define any point on the sphere in terms of 'azimuth' (horizontal) bearing from the north point of the horizon E=90°, S=180°, W=270°, and so on for any intermediate position. The vertical bearing is measured in 'altitude' from 0° (horizon) to 90° (zenith). He can also use negative values, eg the nadir's altitude is -90°. The zenith and nadir are effectively the poles of the horizon coordinate system. All these reference points stay fixed – our observer is quite simply on home ground.

The observer's sphere of horizon coordinates is shared by a second sphere called the star or celestial sphere. This second sphere is usually inclined to the first (unless our observer resides at the north pole of the Earth!) and has a virtually identical system of measuring angles, but these are called by different names. Our observer resides at a latitude of 45° north so the axis of the celestial sphere (which remains parallel to the Earth's axis passes through him at this angle marked by the line Np (Ncp in the X-eyed program) and SP (or Scp). The star Polaris marks the north celestial pole in the sky.

Inclined at 90° polar axis is the equatorial plane which cuts through the observer's E/W lines. In the 3D Celestial Sphere X program, this line is drawn as if by a star rising in the east, reaching its greatest altitude as it crosses the southern meridian and then setting in the west. The Star's progress continues even when below the observer's horizon, and 23 hours and 56 minutes later it returns to its starting point. (On the Spectrum this is speeded up to about a minute or so!). This period is used to calculate sidereal (star) time which is 4 minutes adrift

from the Earth-based time of 24 hours, so the two return to sychronisation. In this way a different starry sky is presented as the seasons go by.

The star sphere is divided horizontally by a line parallel to the equator called Declination (Dec). It is measured in degrees from 0° (celestial equator) to 90° (celestial pole): south of the celestial equator, the values are negative. The lines running at 90° to the Declination lines are called Right Ascension (RA) and are similar to lines of longitude on Earth but are measured in hours instead of degrees. Each hour of RA equals 15° so that 24 hours equal 360° – a complete circle of the star sphere. Lines of RA are measured from right to left across the sky.

The RA of a star when it crosses the meridian also marks the sidereal time at that moment. This is a point worth remembering. It is only necessary to calculate the sidereal time for the day and hour (using the sidereal time program) to find which stars are due south – use a star atlas for guidance. As the stars (and the Sun, Moon and planets) are at their greatest altitude when on the meridian, they can be seen to best advantage then, particularly through a telescope.

Star Point

It is often of interest to know where a particular star or planet can be found in the sky, and this short program does just that. Because of its brevity, it does assume that the star coordinates in terms of Right Ascension and Declination are known beforehand. In the case of a star, a star atlas like *Norton*will give the information and for a planet reference can be made to *Sky & Telescope* or the *BAA Handbook* (see appendix). Alternatively, the planetary positions can be predicted by the **Planetary Ephemeris program contained in Chapter 6**.

Equatorial to horizon coordinate conversión

What the program does is convert the star or planet's equatorial coordinate position (RA and Dec) to horizon coordinates in terms of the azimuth and the altitude above that horizon. **Figure 2.2**, earlier in this chapter, clarifies the inter-related spheres involved. Other relevant facts are the date and the time the observation is to be made and the latitude of the observer. This information is the initial INPUT at the beginning of the program.

The next section of the program, form Lines 80 to 120, works out the sidereal time prior to the INPUT of the star name, etc. The INPUT of the RA takes the form of, for example:

12.56 (12h 56m)

The decimal point must be included. In Line 300, this is converted to hours and decimal hours via the variable rh. The actual conversion to horizon coordinates is executed in Lines 410 and 390 via the variables az (azimuth) and al (altitude).

Using the results

The program results are PRINTed in the final section (Line 440) with the star name and coordinated position in RA and Dec highlighted with PAPER 5. If the star is below the observer's horizon, then Line 445 FLASHes this information.

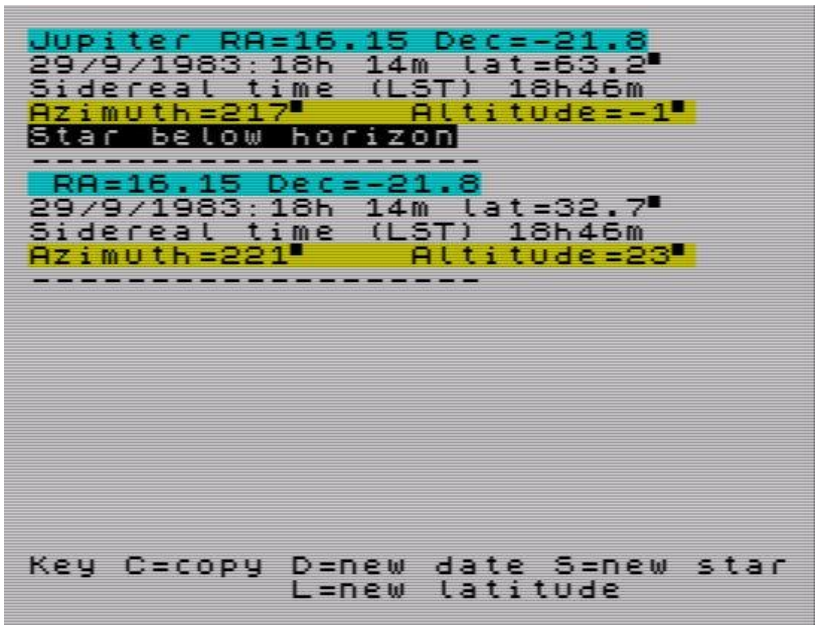
The program then provides an option to INPUT a new date or a new star for the existing date. If the latter option is taken, then the GOTO 130 command is effected and the new star's data has to be INPUT afresh. You will usually require information on a number of stars, and perhaps planets, at a particular date and hour and this option speeds up the process – each successive result SCROLLing up the screen. See Figure 2.3.

Now the azimuth and altitude of perhaps a series of stars is at hand, being either noted or COPYed off the screen – how are they located in the sky? Some 'field work' is now essential from a convenient vantage point of perhaps garden or terrace with a fairly unobstructed skyline (see Figure 2.4).

If you choose bright stars for the program, then you should be able to identify them in the sky, with the help of a star atlas or a COPY from the Starmaps program.

Figure 2.3

The azimuth and altitude of Jupiter for the same time on the same day, but from different latitudes. In the upper example, for latitude 63.2N Jupiter has just set (-1°) but from latitude 32.7N it is still 23° above the horizon.



Elusive planet Mercury

Planet Mercury shows us why familiarity with the local horizons is essential. The planet is invariably only visible in a bright sky within a hour of sunset or sunrise, when the stars can give no guidance as to its location. So plotting its position on a star atlas may be pointless.

In the northern hemisphere, the most favourable time of year to find the planet is on a spring evening in the western sky and on an autumn morning before sunrise in the eastern sky.In the southern hemisphere, the favourable seasons for viewing are reversed, ie autumn for the evening appearance and spring for the morning. Mercury is never more than 20° from the Sun.

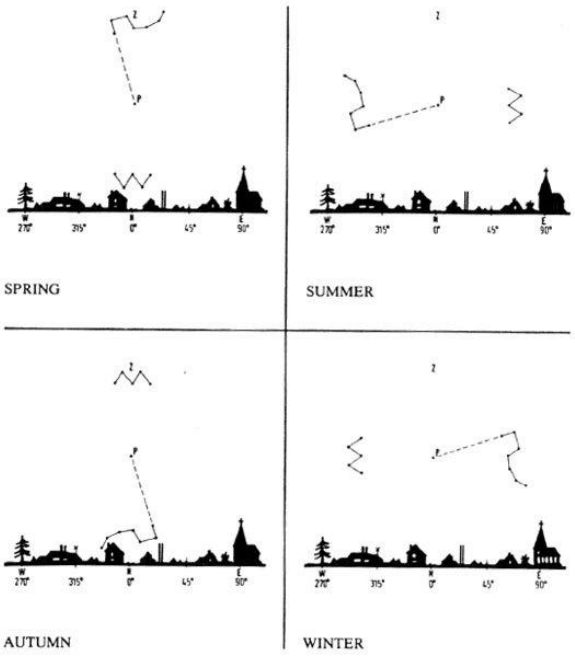
Mercury's RA and Dec can be found in the *BAA Handbook* where the 'elongation', or distance from the Sun in degrees, provides useful guidance as to whether a search is likely to be fruitful. Alternatively the **Planetary Ephemeris program (Chapter 6)** can be used to predict RA and Dec of Mercury and the **Solar System trek program (also in Chapter 6)** for suitable elongation. Here it is advisable to INPUT a number of dates, say at three-day intervals, before they are transferred to our Star Point program (in case some prove unfavourable).

Note: The famous astronomer Copernicus (1473 – 1543) is reputed never to have seen Mercury despite several attempts to do so. Perhaps the mists that arose from the River Vistula in Thorn, Poland, where Copernicus resided, thwarted him. Using this program, you may have a better chance.

Figure 2.4 A Local Skyline about Midnight

The sketches show the Plough (Big Dipper) and Cassiopeia (the 'W') for each season with Polaris (P) midway between and fixed above the north point. The Pole Star's altitude above the horizon is equal to the observer's latitude (about 51° for London). Z marks the zenith – directly above the observer's head.

The dotted line from the 'Pointers' should aid identification of Polaris. Suitable landmarks should then be found along the whole horizon for future reference using a fixed vantage point.



```
9 REM **********************
10 REM AltAz Star Point
11 REM **********************
30 RESTORE : DATA 0,16,16,124,16,16,0,124: FOR f=0 TO 7: READ d: POKE USR "a"+f,d:
NEXT f
60 INPUT "Date yyyy,mm,dd";TAB 5;y;TAB 10;mm;TAB 13;d: IF mm>12 OR d31 THEN GO TO
60
65 INPUT "Local time: hour (0-23) ";th;TAB 12;"min (0-59) ";mi: LET tt=th+mi/60
69 REM **********************
70 REM Julday/Sidereal Time
71 REM **********************
80 LET yy=y: LET m=mm
90 IF m>2 THEN LET m=m+1: GO TO 110
100 LET y=y-1: LET m=m+13
110 LET j=INT (365.25*y)+INT (30.6001*m)+d+1720982
120 LET g=6.63627+6.570982e-2*(j-2443144): LET ts=g-INT (g/24)*24
129 REM **********************
130 INPUT "Star/planet name", LINE a$
140 INPUT "Right Asn (hh.mm)";TAB 11;ra
150 INPUT "Dec (\add.d)";TAB 5; LINE d$
160 LET dc=VAL d$: IF ABS dc>=90 THEN LET dc=dc-.1
180 INPUT "Your lat (\all.l)";TAB 10; LINE l$
181 REM **********************
190 LET l=VAL l$: IF ABS l>=90 THEN LET l=l-.1
280 PRINT PAPER 5;a$;" RA=";ra;" Dec=";d$
290 PRINT d;"/";mm;"/";yy;":";th;"h ";mi;"m lat=";l$;"\' "
300 LET c=360: LET r=180/PI: LET lr=l/r: LET f=100/60: LET dr=dc/r: LET rh=INT
ra+(ra-INT ra)/f
310 LET t=tt
330 LET s=t+ts+t/1436*4
340 IF s>24 THEN LET s=s-24
350 IF s0 THEN LET az=c-az
440 PRINT "Sidereal time (LST) ";: LET h=st: GO SUB 530: PRINT PAPER
6;"Azimuth=";INT (.5+az);"\' ","Altitude=";INT (.5+al);"\' "
445 IF al<0 THEN PRINT FLASH 1;"Star below horizon"
469 REM **********************
470 REM select
471 REM **********************
480 PRINT #1;"Key C=copy D=new date S=new star L=new latitude"
```

```
485 PRINT "--------"
490 PAUSE 0: LET b$=INKEY$
500 GO TO (b$="l")*180+(b$="c")*510+(b$="s")*130+(b$="d")*60
510 COPY : GO TO 490
519 REM **********************
520 REM decimal hrs=hr mn
521 REM **********************
530 PRINT INT h;"h";INT ((h-INT h)*60+.5);"m": RETURN
9900 REM **********************
9990 SAVE "starpoint" LINE 1
```

Star Tracker

This program expands the previous Star Point program and, using the full Spectrum screen with colour, PLOTs a star or planet on a representation of the sky over a 24-hour period. It includes DATA on twenty selected bright stars fairly evenly distributed over the star sphere. This considerably eases the task of finding such information. (The program does have the option of omitting this DATA, perhaps for inclusion at a later date. This effectively reduces the listing by one third. Do so by omitting Line 125 and all lines from 1000 onwards.)

Figure 2.5

The daily track of the star Vega across the skies of Reykjavik plotted at hourly intervals. The short bar marks the start and finish of the plot at midnight GMT.

Figure 2.6

From Perth (Australia), Vega only appears briefly above the northern horizon.

Figure 2.7

Adjacent to the North Pole, Vega remains at an almost constant altitude above the whole horizon.

Figure 2.8

From Nairobi, Vega cricles the northern horizon in an anticlockwise direction.

Test examples

The sample screen COPYs, Figures 2.5, 2.6, 2.7 and 2.8, show one star (Vega) on one particular day (21 August 1984), as PLOTted over a 24-hour period for various latitudes. These are:

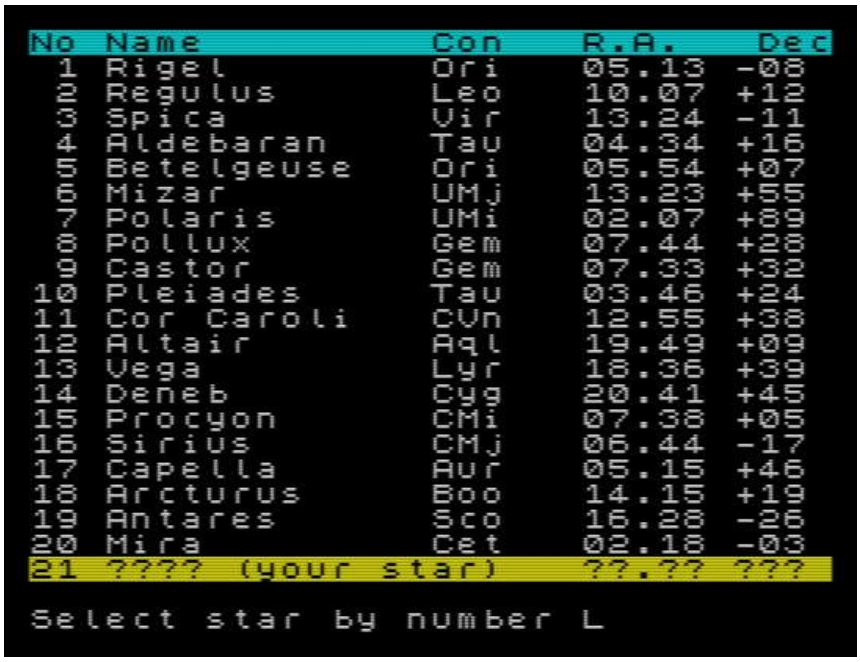
+67.7° N around the latitude of Reykjavik, Iceland
-32.1° S around the latitude of Perth, Australia
+89.0° N around the latitude of adjacent to North Pole
-02.1° S around the latitude of Nairobi, Kenya

INPUTting the information

Any one of the 20 stars listed can be selected by simply entering the star number as shown in Figure 2.9. It is not necessary to know the star's coordinates (RA and Dec) on the celestial sphere as these are stored in the program as DATA and used automatically. If the star you are interested in is not listed, INPUT 21 and enter the name, RA and Dec when prompted by the program. This data is available from any star atlas. If you wish to enter a planet, again use INPUT of 21 but consult an astronomical almanac or use the **Planetary Ephemeris program in Chapter 6** to find its locations for the selected date.

Figure 2.9

Selecting a star by number. Option 21 allows the INPUT of an alternative star or planet.



Any date can be INPUT, but for the stars, which are considered to be fixed to the celestial sphere, you need not worry about the year, as they tend to repeat the same display on the same day each year: and so they repeat themselves for the start and finish of the PLOT routine on the same day each year. This is not the case with the planets, which are the wanderers of the sky.

The screen display

A Mercators projection is used to represent the sky and some distortion is inevitable in converting a spherical surface to the flat TV display. (You may recall how large Greenland and Antartica appeared on school maps using this projection.)

The zenith at +90° (the point immediately above the observer's head) and the nadir at -90° (the point below his feet) become imaginary lines at the top and bottom of the display. The encircling 360° of the horizon at 0° altitude is marked N, E, S, W and back again to N through the middle of the screen, dividing the sky from the ground below.

The star or planet is PLOTted at hourly intervals using INK 9 (as a contrast to the PAPER colour tone) and so is visible even when below the observer's horizon. This interval can be amended to any value by changing the STEP (set at 1 — for one hour — in this program) in Line 320.

Distant shores

Unless you have resided in the opposing hemisphere and are familiar with the night sky, the motion of a star as accurately depicted in this program may come as something of a surprise. For example, a star like Vega will move from east to west in a clockwise direction in the northern hemisphere but in an anticlockwise direction in the

southern hemisphere. From the equator, stars may rise vertically to the zenith from the eastern horizon and appear to 'jump' (it is a line and not a point remember) across the screen and set in the west. At the north pole all stars and planets cruise from left to right parallel to the horizon, never rising or setting. The same applies at the south pole but the motion is in the reverse direction from right to left. Stars that refuse to rise at the north pole are permanently above the horizon at the south pole and vice versa.

All these effects can be demonstrated with this program. A star that never sets at a given latitude is said to be 'circumpolar' and of course the Sun (which is the nearest star) is so placed in the 'land of the midnight sun at certain latitudes (+ or -) during that hemisphere's summer. Test it!

```
9 REM ************************
10 REM Star Tracker
11 REM ************************
30 RESTORE : DATA 0,16,16,124,16,16,0,124: FOR f=0 TO 7: READ d: POKE USR "a"+f,d:
NEXT f
40 DIM z$(320)
50 PRINT "Enter the following"
60 INPUT "Date yyyy,mm,dd";TAB 5;y;TAB 10;mm;TAB 13;d: IF mm>12 OR d31 THEN GO TO
60
69 REM ************************
70 REM JulianDy/Sidereal Time
71 REM ************************
80 CLS : LET yy=y: LET m=mm
90 IF m>2 THEN LET m=m+1: GO TO 110
100 LET y=y-1: LET m=m+13
110 LET j=INT (365.25*y)+INT (30.6001*m)+d+1720982
120 LET g=6.63627+6.570982e-2*(j-2443144): LET ts=g-INT (g/24)*24
125 GO TO 1000: REM star list
130 INPUT "Star/planet name", LINE a$: IF LEN a$>10 THEN PRINT #0; FLASH 1;"Too
big!": PAUSE 100: GO TO 130
140 INPUT "Right Asn (hh.mm)";TAB 11;ra
150 INPUT "Declination (\add.d)";TAB 13; LINE d$
155 IF d$(1)"+" AND d$(1)"-" THEN GO TO 150
160 LET dc=VAL d$
170 IF ABS dc>89.9 THEN GO TO 150
180 INPUT "Your latitude (\all.l)";TAB 15; LINE l$
185 IF l$(1)"+" AND l$(1)"-" THEN GO TO 180
190 LET l=VAL l$
200 IF ABS l>89.9 THEN GO TO 180
```

```
209 REM **********************
210 REM print sky projection
211 REM **********************
220 BORDER 0: PAPER 1: INK 9: CLS
230 PRINT AT 1,0;"zenith loctime GST az alt"
240 PRINT PAPER 4;AT 11,0;z$: FOR n=1 TO 20 STEP 2: PAPER 4 AND n>10
250 PRINT AT n+1,0;100-n*10
260 NEXT n
270 PRINT AT 11,0;"n-ne-+e-se-+s-sw-+w-nw-n-10 horizon"
280 PRINT PAPER 5;AT 19,3;a$;" RA=";ra;" Dec=";d$
290 PRINT "-90 nadir ";d;"/";mm;"/";yy;" lat=";l$: PAPER 1
300 LET c=360: LET r=180/PI: LET lr=l/r: LET f=100/60: LET dr=dc/r: LET rh=INT
ra+(ra-INT ra)/f
309 REM **********************
310 REM main'timer'loop T=hrs
311 REM **********************
320 FOR t=0 TO 24: BEEP .01,40
330 LET s=t+ts+t/1436*4
340 IF s>24 THEN LET s=s-24
350 IF s<0 THEN LET s=s+24
360 LET st=INT (s*100)/100
369 REM **********************
370 REM calc azimuth,altitude
371 REM **********************
380 LET hr=((s-rh)*15)/r
390 LET al=r*ASN (SIN dr*SIN lr+COS dr*COS lr*COS hr)
400 LET ar=al/r: LET v=SIN hr
410 LET az=r*ACS ((SIN dr-SIN lr*SIN ar)/(COS lr*COS ar))
420 REM decimal hrs=he mn
429 REM **********************
430 REM plot startrack on sky
431 REM **********************
440 PRINT AT 0,7;: LET h=t: GO SUB 530: PRINT AT 0,15;: LET h=st: GO SUB 530
445 PRINT AT 0,22;INT az;"\' ";TAB 27;INT al;"\' "
450 LET z=az/5.7*4: LET a=al/5*4: PLOT z,a+84: NEXT t
460 DRAW FLASH 1;1,0
469 REM **********************
470 REM beep & select option
471 REM **********************
```

```
480 PRINT #1;"Key C=copy D=new date S=new star L=new latitude"
490 FOR n=.1 TO 5 STEP .1: BEEP n/100,n*10: NEXT n: PAUSE 0
500 POKE 23693,56: GO TO
(INKEY$="l")*180+(INKEY$="c")*510+(INKEY$="s")*1000+(INKEY$="d")*60
510 COPY : GO TO 490
519 REM **********************
520 REM decimal hrs=he mn
521 REM **********************
530 PRINT INT h;"h";INT ((h-INT h)*60+.5);"m": RETURN
999 REM **********************
1000 REM starlist
1001 REM **********************
1010 CLS : RESTORE 2000: GO TO 2100
1014 REM **********************
1015 REM *input star no
1016 REM **********************
1020 INPUT "Select star by number ";t: IF t21 THEN GO TO 1020
1030 IF t=21 THEN GO TO 1030
1040 LET a$=k$(t): LET ra=VAL r$(t): LET d$=t$(t): LET dc=VAL d$(2 TO ): IF
d$(1)="-" THEN LET dc=-dc
1050 GO TO 180
1999 REM **********************
2000 DATA "Rigel","Regulus","Spica","Aldebaran","Betelgeuse",
"Mizar","Polaris","Pollux","Castor"
2020 DATA "Pleiades","Cor Caroli","Altair","Vega","Deneb",
"Procyon","Sirius","Capella","Arcturus","Antares","Mira"
2024 REM **********************
2025 REM Constellations
2026 REM **********************
2030 DATA "Ori","Leo","Vir","Tau","Ori","UMj","UMi","Gem",
"Gem"
2040 DATA "Tau","CVn","Aql","Lyr","Cyg","CMi","CMj","Aur",
"Boo","Sco","Cet"
2044 REM **********************
2045 REM RA & Dec of stars
2046 REM **********************
2050 DATA "05.13","-08","10.07","+12","13.24","-11","04.34",
"+16","05.54","+07","13.23","+55"
2060 DATA "02.07","+89","07.44","+28","07.33","+32","03.46",
```

```
"+24","12.55","+38","19.49","+09"
2070 DATA "18.36","+39","20.41","+45","07.38","+05","06.44",
"-17","05.15","+46","14.15","+19","16.28","-26","02.18","-03"
2089 REM **********************
2090 REM List Stars
2091 REM **********************
2100 DIM k$(20,11): DIM e$(20,4): DIM r$(20,6): DIM t$(20,3)
2105 PRINT PAPER 5;"No Name Con R.A. Dec"
2110 FOR f=1 TO 20: READ k$(f): PRINT (" " AND f<10);f;" ";k$(f): NEXT f
2120 FOR f=1 TO 20: READ e$(f): PRINT AT f,16;e$(f): NEXT f
2130 FOR f=1 TO 20: READ r$(f),t$(f): PRINT AT f,22;r$(f);t$(f): NEXT f
2140 PRINT PAPER 6;"21 ???? (your star) ??.?? ??? "
2150 GO TO 1020
9900 REM **********************
9990 SAVE "Startrack" LINE 1
```

Chapter 3 – The Moon (The Moon's Phases)

The lifegiving Sun gets a raw deal in the public eye and the reason is not too difficult to find — the weather. The Sun completely dominates the scene and man has absolutely no control over these events.

In contrast the Moon has no such public relations problems. It is probably the first object in the sky to set a young mind wondering — 'What is it?' — 'Where is it?'. For many the Moon is the embodiment of all that is mysterious and strange beyond the cosy confines of home and yet it is just the stepping stone to the exploration of a beautiful universe. Even if we can never visit these places in person we can learn to understand them and perhaps view them aided with modest binoculars or a small telescope. The Moon is a perfect starting point.

The Moon's Phases

The Moon is our nearest neighbour in space and the only world (apart from Earth) that man has set foot upon — during the late 1960s with the Apollo missions. The program that follows is one of the shortest in this book but the Spectrum graphics more than compensate for its brevity. The complete phases of the Moon from new moon to full moon and back again (via a shrinking crescent) to new moon are shown, covering one monthly cycle. Figures 3.1 and 3.2 are typical examples.

Figure 3.1
The Moon one day old

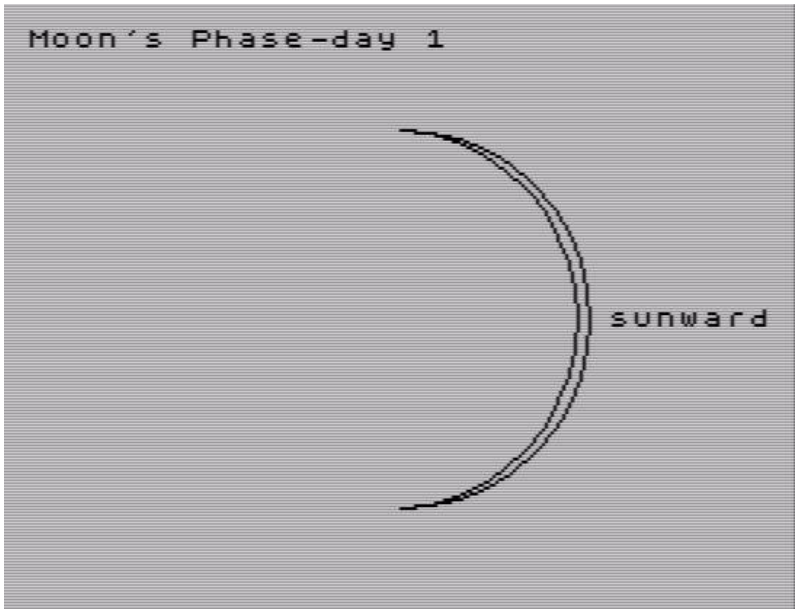
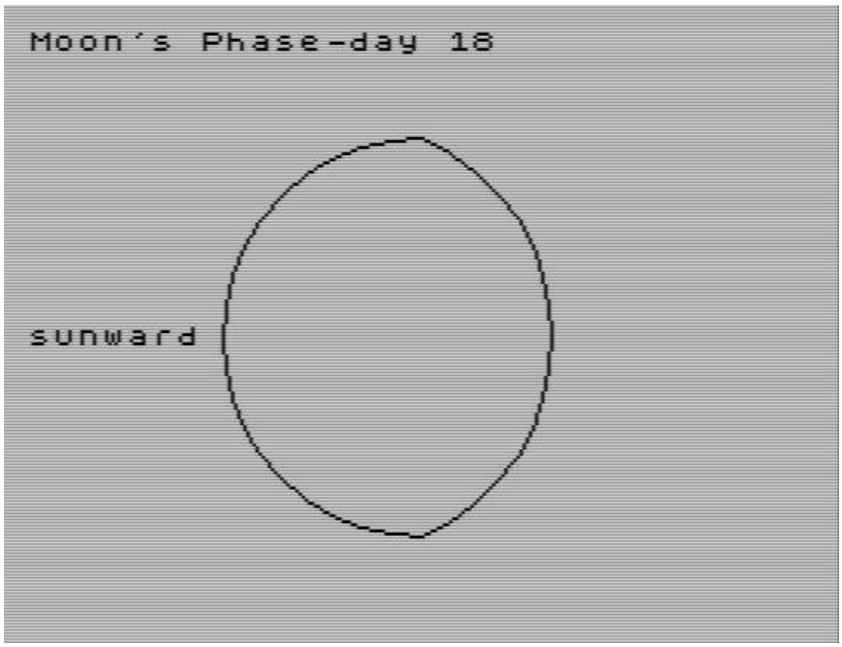


Figure 3.2
The Moon after 'Full'



The Moon's appearance

Artists are renowned for including the crescent moon in their works and invariably getting its appearance completely wrong! The phases of the Moon are measured from new moon to the next new moon, and in reality the Moon is invisible at this stage due to its apparent proximity to the Sun. The only true new moon that can be seen is when it passes directly in front of the Sun during an eclipse of the Sun, so that the Moon's silhouette is revealed.

The commonly ascribed new moon is when the crescent first becomes evident, about two days later, in a bright western sky as a letter C backwards. The crescent grows in the days that follow, becoming brighter as more of the lunar surface is illuminated as it moves to the left away from the Western skyline.

About fourteen days later the Moon appears as a very bright circular disc due south at midnight — full moon. It is now directly opposite to the Sun (hidden below the northern horizon). Continuing to move to the left amongst the starry background, its phase diminishes until fourteen days later it is lost in the bright dawn sky having shrunk to a fine crescent again — this time like a true letter C (ie not reversed). Astronomers find it convenient to regard the crescent phase as a bow — an arrow shot from it would pierce the Sun. Artists please note!

The program

The program uses a simple FOR/NEXT d loop to DRAW the two arcs which represent each phase. Line 80 first PLOTs a reference point on which to start the DRAWing and then DRAWs the bright 'limb' or edge, that is nearest to the Sun's direction. Line 70 uses a conditional PRINT statement to identify this limb with the word 'sunward'. Line 110 then DRAWs the 'terminator' or sunrise/sunset line on the Moon's surface.

Because the Moon is a near-perfect spherical globe, the limb is DRAWn with the value PI (via the variable P in Line 40) as a semicircle. Line 50 renders this value as negative after the fourteenth day (full moon) so that this initial arc is DRAWn on the left of the screen instead of on the right, and in the opposite curvature. The formula in Line 110 controls the terminator line from semicircular (new or full moon) to a straight line (first or last quarter moon). Lines 90 and 100 ensure the terminator arc is DRAWn with a positive (to right) or negative (to left) curvature.

The Moon is effectively a globe and, although the terminator line advances at about 13° each day (28 days x 12.86° = 360°) as seen from Earth, foreshortening occurs so that the daily motion of this line is not constant. It appears to advance more rapidly about the time of first or last quarter moon whilst in the centre of the disc. About new or full moon the terminator is seen at such acute angles that the actual movement is much compressed — to the point where the precise occasion of full moon can be misjudged by at least a day or so.

This program mimics this effect solely by the formula in Line 110. As an experiment, try amending the variable x in Line 30 and the value in the formula (the last part of the expression), ie 25.

The program is intended to indicate the varying phases of the Moon and no particular accuracy is claimed for the precise shape of the terminator line or for the timing interval of the monthly cycle. These two aspects are unrelated and are explained below.

The true profile of the terminator line is an ellipse but the Spectrum's DRAW command is used instead (where the third part of the expression produces the curvature) to ensure rapid results. The errors produced by the arc when compared with the ellipse are not excessive. The major part of the error, such as it is, occurs adjacent to the polar regions of the DRAWing from a latitude of about 60° to 75° and then only for a few days about first or last quarter moon. This compromise was deemed acceptable.

The FOR/NEXT d loop in Line 40 accounts for a 28-day period to represent the monthly cycle of phases. This period is precisely divisible by four, so that each quarter moon is displayed. In reality the Moon's 'mean synodic period' (average from new moon to new moon) is 29d 12h 44m or about 29.53 days, which means that the program RUNs about 5% *fast* on the real thing.

```
9 REM ***********************
10 REM Moon's Phase
11 REM ***********************
30 LET x=2.5: LET r=PI/180
40 FOR d=0 TO 28: LET p=PI
50 CLS : IF d>14 THEN LET p=-p
60 PRINT "Moon's Phase-day ";d
70 PRINT AT 12,(25 AND d7 THEN LET b=b-14
110 DRAW 0,-130,x*ATN (r*b*25)
120 PAUSE 50: NEXT d
```

```
9900 REM **********************
9990 SAVE "phaseM"
```

The Moon's Phases – In Orbit

The lifegiving Sun gets a raw deal in the public eye and the reason is not too difficult to find — the weather. The Sun completely dominates the scene and man has absolutely no control over these events.

This program is based upon the previous Moon's Phases program but extended to PLOT both Earth and Moon in orbit about each other and to provide a full description of each phase. The latter is done via the conditional PRINT statement in Line 290. Line 310 handles conditions of potential eclipses and Line 320 the terminator line of the waxing and waning Moon. REM statements are liberally included to show the general structure of the program.

The Earth/Moon system

The Moon is quite the most unique body in the solar system in possessing the largest mass in relationship to the body it orbits (the Earth).

The mass of a satellite is typically between 1/1,000,000th and 1/4000th that of the primary. The Moon is l/80th the mass of the Earth so that the Earth and Moon behave like a. twin planetary system, each orbiting about the other around the common centre of gravity. This latter point is not at the centre of the planet (as it is for all other planets) but about half way to the planet's centre some 2000 miles beneath the surface on a line joining the centres of each body.

The program mimics this effect but for clarity the diameters of both the Earth and Moon have been enlarged 100-fold. Otherwise, to true scale on the Spectrum screen, they would be reduced to a few pixels across and would be virtually invisible. A single flickering pixel beneath the Earth's surface marks the centre of gravity of the Earth/Moon system and is produced by Line 250 with the OVER 1 command.

Shadows and eclipses

The shadows from both the Earth and the Moon are cones stretching away from the direction of the Sun. In the case of the Moon, this shadow can (at times of new moon) just fall on the Earth's surface. An observer located immediately under this cone would see the Sun completely blotted out in a total eclipse and day turn into night with the stars coming into view. This event is rare and short-lived — it is barely more than a few minutes before the Earth's axial rotation and the Moon's orbital progress carry the shadow away at about 1400 mph. Only Concorde can (and has done so) keep up with the eclipse shadow stretching the event into 40 minutes or so whilst a whole continent is traversed.

Eclipses of the Moon occur when the Earth's shadow falls on the Moon. This can only occur at full moon and, because the Earth's shadow is so huge even at a range of 384,000 km (Moon's mean distance), the Moon can be completely engulfed. Potentially, given clear skies, half the world's population (in the hemisphere which would be in darkness) could see this event, in contrast to the few privileged thousands for a total eclipse of the Sun.

Eclipses do not occur at each new and full moon because the Moon and its orbit are more closely bound to the Sun than to the Earth. Only when the Moon crosses the line between Earth and Sun (the ecliptic) at a point called the 'node' can an eclipse occur.

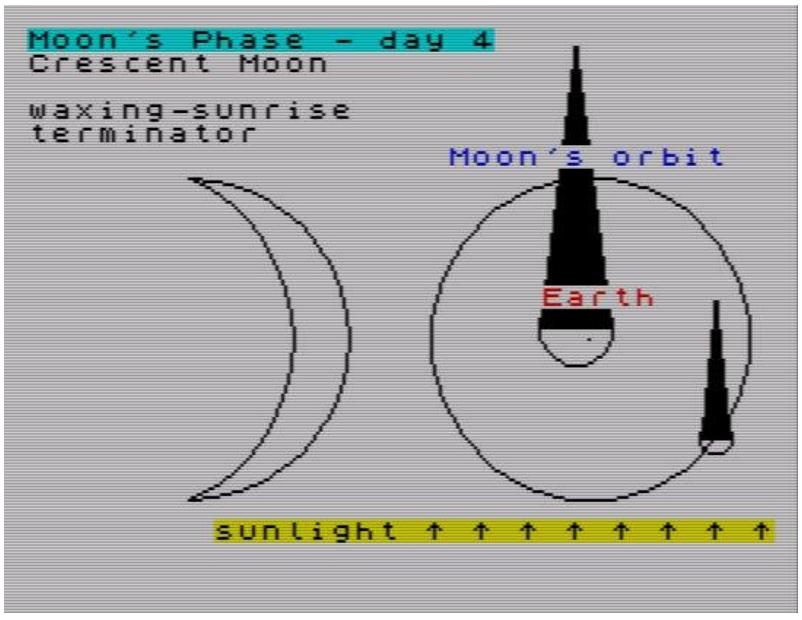
The program

Once the program has been entered and RUN, a complete monthly cycle is displayed. The user then has the option to INPUT any day or decimal day for display. The economical conditional GOTO in Line 470 automatically reRUNS the program (ie GOTO 0) if a day 28 is entered. COPYs of the screen display can be made at any time during the program RUN as follows:

```
BREAK: COPY: CONTINUE
```

Figure 3.3 is a screen COPY obtained this way.

Figure 3.3
Split display show's the Moon's phases, and Earth and Moon orbiting around a common centre of gravity.



The program RUNs in blissful silence but, if this proves unnerving, then the appropriate BEEPs (as well as colour) can be incorporated in the display.

```
10 REM Phases of the Moon
15 BORDER 7: PAPER 7: INK 9: CLS
20 LET g=192: LET h=69: LET j=55: LET k=52: LET m=100: LET z=0
40 FOR d=0 TO 28
60 CLS : CIRCLE g,h,j
80 LET p=PI: LET a=(d-7)/14*p
90 LET c=g+j*COS a
100 LET e=h+j*SIN a
110 CIRCLE c,e,5.5
120 REM ******************
121 REM draw earth in orbit
122 REM ******************
130 LET c1=g-6*COS a
140 LET e1=h-6*SIN a
150 CIRCLE c1,e1,13
159 REM ********************
160 REM draw moons shadow
161 REM ********************
170 FOR n=.5 TO 5
180 PLOT c+n,e: DRAW 0,k-10*n
190 PLOT c-n,e: DRAW 0,k-10*n
200 NEXT n
220 FOR n=.5 TO 13
230 PLOT c1+n,e1: DRAW 0,m-8*n
240 PLOT c1-n,e1: DRAW 0,m-8*n
250 PLOT OVER 1;g,h
260 NEXT n
269 REM ********************
270 REM describe phases
271 REM ********************
280 PRINT PAPER 5;"Moon's Phase - day ";d
290 PRINT FLASH 1;("New" AND (d=0 OR d=28))+("1st" AND d=7)+("Last" AND d=21)+("
Quarter" AND (d=7 OR d=21))+("Full" AND d=14)+("Crescent" AND (d>0 AND d21 AND
d<28))+("Gibbous" AND d14 AND d>7 AND d<21)+" Moon"
299 REM ********************
300 REM eclipses and terminator
301 REM ********************
310 PRINT FLASH 1;("Eclipse of Sun possible" AND (d=0 OR d=28))+("Eclipse of Moon
possible" AND d=14)
```

```
320 PRINT ("waxing-sunrise" AND d14)'"terminator"
330 PRINT INK 2;AT 11,22;"Earth";AT 5,18; INK 1;"Moon's orbit"
335 PRINT PAPER 6;AT 21,8;"sunlight ^ ^ ^ ^ ^ ^ ^ ^"
339 REM ******************
340 REM draw bright light
341 REM ******************
350 IF d>14 THEN LET p=-p
360 PLOT j,14: DRAW 0,j*2,p
369 REM ******************
370 REM draw terminator
371 REM ******************
380 LET b=d-7: LET x=2.5
390 IF b>7 THEN LET b=b-14
400 LET n=x*ATN (PI/180*-b*25)
410 PLOT j,14: DRAW 0,j*2,n
419 REM ******************
420 REM jump out of loop (z=1)
421 REM ********************
430 IF z=1 THEN GO TO 450
440 PAUSE 400: NEXT d: LET z=1
450 INPUT "Select day: ";d
459 REM ********************
460 REM conditional goto
461 REM ********************
470 GO TO (d>=0 AND d<=28)*60
9900 REM ******************
9990 SAVE "moon" LINE 1
9999 REM ******************
```

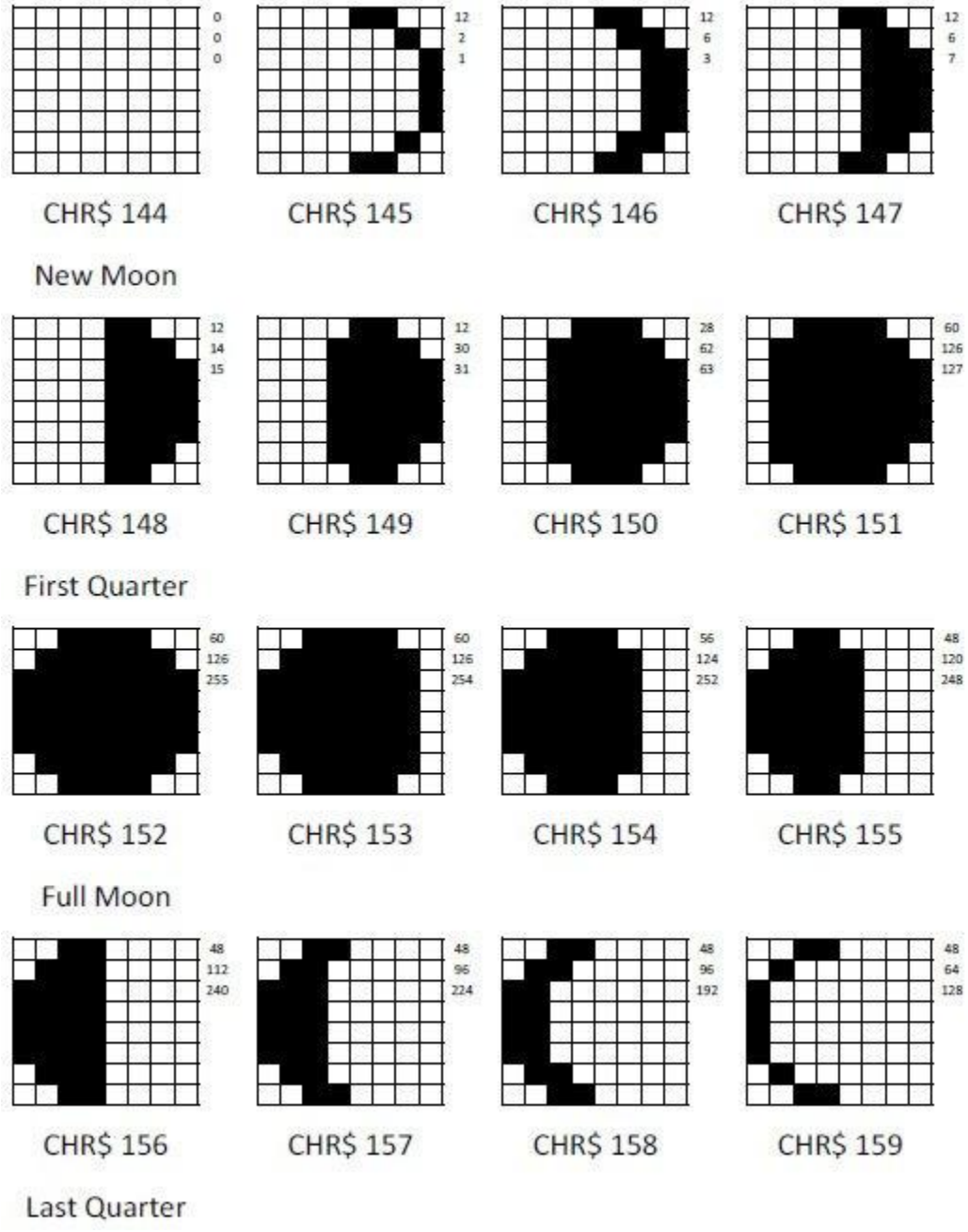
UDG Moon's Phases

When the changing phases of a moon are needed for display in rapid succession and a single character square is large enough, it can be useful to use the UDG character set. Obviously, not every conceivable phase can be indicated but a sufficient selection can be, at least, to make the effect convincing.

The following program, which may be incorporated into a longer program as a subroutine (see the Moons of Mars in Chapter 6), is designed for this purpose. It defines sixteen UDG characters from CHR$ 144 to CHR$ 159

inclusive. The successive phases are from ‘new moon’ (CHR$ 144) to ‘full moon’ (CHR$ 152) and back again to a narrow crescent (CHR$ 159) immediately before ‘new moon’ again. Figure 3.4 details each pixel position of the complete set.

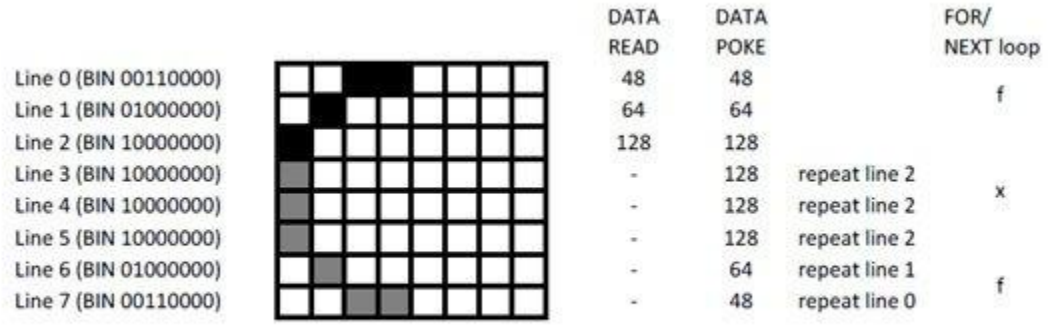
Figure 3.4



To redefine all sixteen characters requires POKEing 128 items of DATA (16×8 lines). In this program the DATA contains only 48 values — sufficient for the first three lines of each character. As the lower section of each character is a ‘mirror image’ of the upper portion, the program does the necessary duplication of DATA as shown in Figure 3.5.

Figure 3.5

Final Crescent phase – CHR$ 159

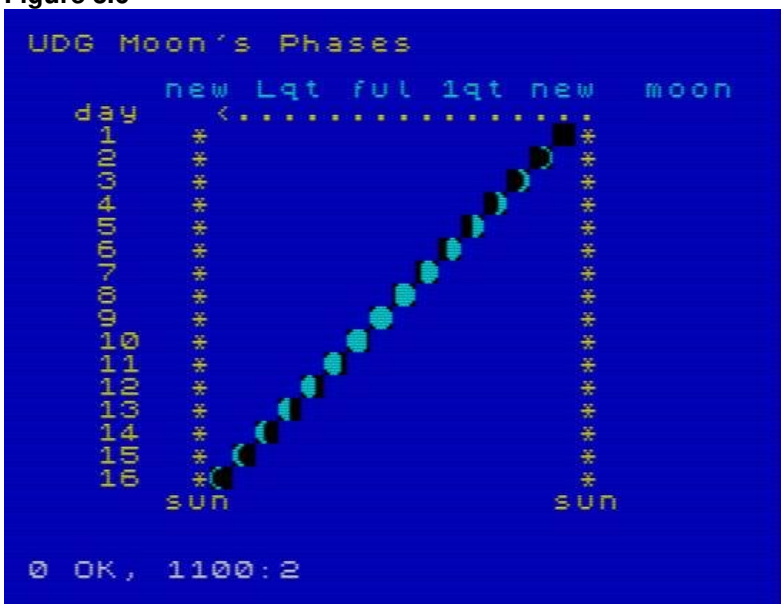


Program RUN

Enter the complete program and RUN it to see the results of your efforts. Lines 110 to 220 define the new UDG character and, from Line 1000, there is a demonstration PRINTing to show the daily changes in the moon's phases (see Figure 3.6).

Note that the increasing phase is PRINTed further to the left — away from the Sun — for each day represented, until the final crescent phase immediately to the right of the Sun. Full moon is PRINTed at maximum distance from the Sun in the centre of the display. This correctly mimics the appearance and movement of all the major moons in the solar system in a direct or anticlockwise direction as seen from above (to the north) of the moon's orbit.

Figure 3.6



New graphics

A simple procedure to redesign any characters of this type is as follows:

1) Sketch out the new design on squared paper.

2) Enter as a direct command in the Spectrum the BIN values, eg

PRINT BIN 00000110 ENTER (decimal value 6-lineO)

PRINT BIN 00000111 ENTER (decimal value 7 – line 1)

PRINT BIN 00001111 ENTER (decimal value 15-line 2)

3) Substitute the new values into the DATA, remembering the first three values are for CHR$ 144 and the last three values are for CHR$ 159.

```
99 REM ************************
100 REM UDG moons phases
101 REM ************************
110 DATA 0,0,0,12,2,1,12,6,3,12,6,7,12,14,15,12,30,31,28,62,63
120 DATA 60,126,127,60,126,255,60,126,254,56,124,252,48,120,248
130 DATA 48,112,240,48,96,224,48,96,192,48,64,128
139 REM ************************
140 REM poke UDG moons
141 REM ************************
150 FOR n=0 TO 15: FOR f=0 TO 1
160 READ p
170 POKE USR CHR$ (144+n)+f,p
180 POKE USR CHR$ (144+n)+7-f,p
190 NEXT f
200 READ c: FOR x=2 TO 5
210 POKE USR CHR$ (144+n)+x,c
220 NEXT x: NEXT n
999 REM ************************
1000 REM test printing
1001 REM ************************
1010 BORDER 1: PAPER 1: INK 6
1020 CLS
1030 PRINT "UDG Moon's Phases"
1040 PRINT AT 20,6;"sun";TAB 23;"sun";AT 2,6; INK 5;"new Lqt ful 1qt new moon ";
INK 6;"day"
1050 FOR n=0 TO 15: LET z=n+4
1060 PRINT AT 3,23-n;"<."
1070 PRINT AT z,3;n+1;AT z,7;"*"
1080 PRINT AT z,24;"*"
1090 PRINT AT z,23-n; PAPER 0; INK 5;CHR$ (144+n): PAUSE 50: NEXT n
```

```
1100 INK 0: PAPER 7
9900 REM ***********************
9990 SAVE "udgmoon"
```

Chapter 4 – Satellites (Launch Your Own Satellite)

Launch your own satellites

The launch of Sputnik I by the USSR in 1957 started the space age and added a new word to the common vocabulary — satellite. In the astronomical context, this simply means a moon and our Moon (capital M of course) is the only natural satellite of Earth.

Since Sputnik I, literally thousands of artificial satellites have been sent into Earth orbit. A few have even been sent to orbit other planets like Venus, Mars and our Moon. On a clear night it is now rare for some 30 minutes to elapse before a satellite passes overhead on its silent progress about Earth — a point of light moving quite rapidly on a slight arced course amongst the stars.

The brighter satellites are usually in lower orbits moving from west to east across the sky (never in the reverse direction). The satellites in polar orbits (passing over the north and south pole at each revolution) are usually fainter and almost perfectly aligned to the Earth's axis so that their movement is in a precise north/south or south/north direction.

Launching satellites can cost millions of pounds each so, before applying for a job at Mission Control, Houston, practise on the simulator programs to follow!

Earth Orbit

This is a modelling program which enables you to place a satellite in orbit about the Earth. There are two parameters to INPUT, the initial height of the satellite above the surface of the Earth and the initial velocity, and so there are literally thousands of permutations to try, no two being precisely identical. The orbit is PLOTted in real time and the screen presentation is automatically rescaled to contain the orbit on the screen.

To ensure that the satellite is of maximum size as it traces around the Earth, only half the orbit is PLOTted — the second half is a precise mirror image of the first half and no particular advantage is gained by completing the orbit. Once the half-orbit is completed, the full orbital period in minutes, hours and days is displayed at the top right of the screen. Throughout the PLOTting, at the bottom left is the 'elapse' time in minutes and the relative x and y coordinate positions as PLOTted. Bottom right shows the initial height and velocity as INPUT.

Of course successful insertion into orbit is not automatically achieved — as space scientists well know. A balance between the initial height and velocity must be made — particularly so for a neat circular orbit.

There is a further problem to consider. Although the satellite will happily orbit the centre of the Earth (this is the centre of the Earth's gravitational field) the Earth's surface sometimes gets in the way and the satellite crashes. The Spectrum BEEPs a polite warning and awaits further INPUT instructions.

There is a second possibility of failure — when you have so high an initial velocity that the satellite is lost to outer space. This particular program does not recognise this condition and will keep rescaling the display, via line 560 (if… YY> 175),each time the satellite reaches the top of the screen, constantly hoping that the satellite will return towards Earth. In this case you must BREAK the program and reRUN it. An option to COPY the screen via the ZX printer is included for your successful orbits.

The Kepler plot

This program PLOTs the orbits according to Kepler's Laws of Planetary Motion (which applies to any lesser body orbiting a larger body), with an elliptical trace and the centre of the Earth as one focus. The satellite speeds up as it gets close to Earth to compensate for the stronger gravitational field experienced. With the following INPUTs, it is possible to test that the program gives reasonably accurate results and is a fair simulation of a body orbiting Earth.

1) Initial height 200 miles
Initial velocity 286 miles/minute
= Orbital period 88 minutes

This is the minimum period for the lowest satellite, like Sputnik I.

2) Initial height 240,250 miles
Initial velocity 37 miles/minute
= Orbital period 27.37 days (mean sidereal period)

This matches our natural satellite — the Moon.

Test examples

The above INPUTS represent neat circular orbits. More often than not, however, an ellipse is formed. If the orbit PLOT touches the lefthand side of the screen the display is not rescaled but left to continue the PLOTting as a mirror image back towards right again to maintain maximum image size.

The screen COPYs (Figures 4.1, 4.2 and 4.3) show typical results from the program.

Figure 4.1

This orbit simulates our natural satellite, the Moon.

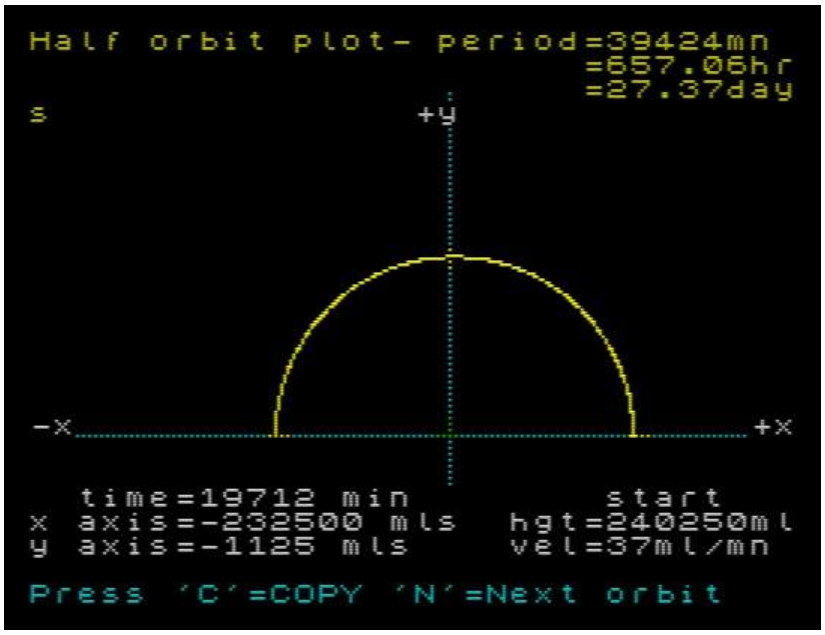


Figure 4.2

Typical orbit plotted from apogee (point adjacent + x sign).

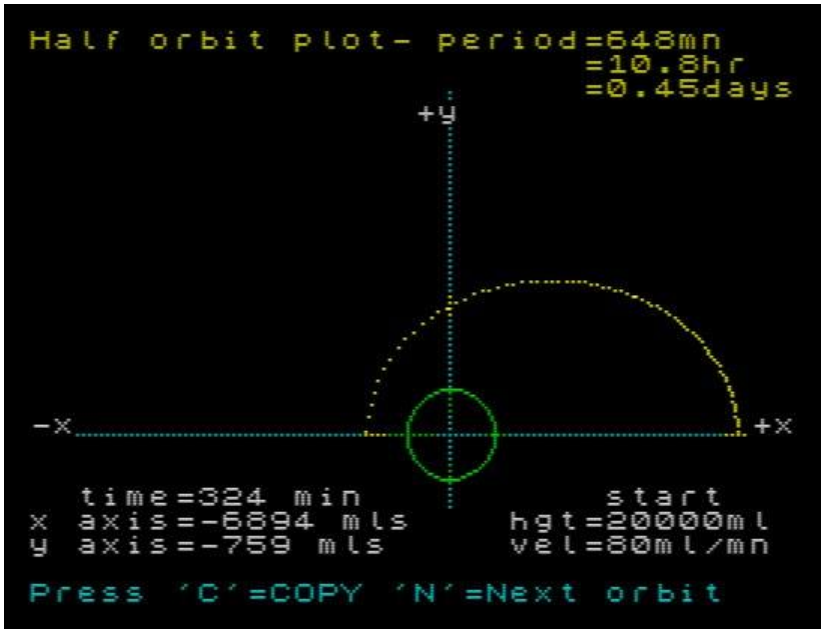
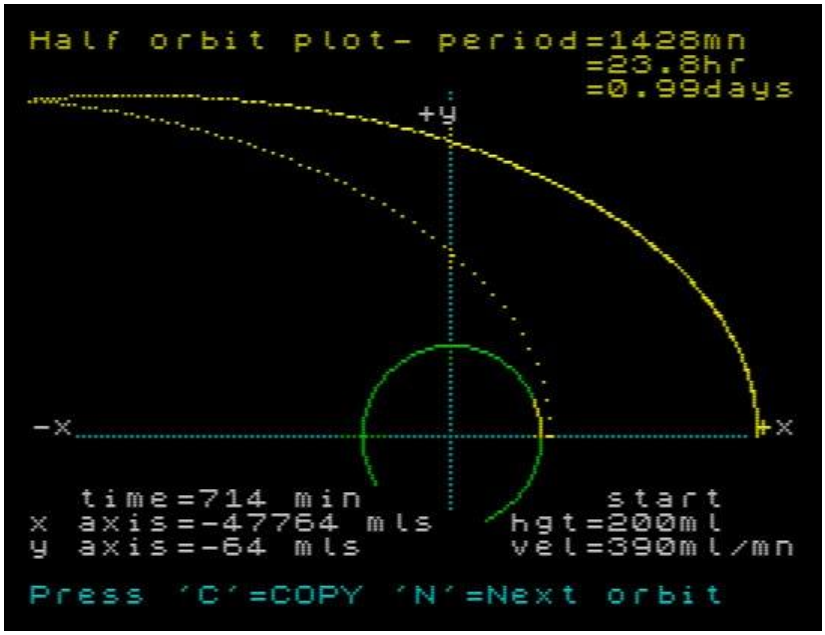


Figure 4.3

Typical orbit started adjacent to globe (perigee). The orbit is mirrored by left hand side of display, plotting towards the right (apogee).



Mission control

As an experiment, try 'designing' orbits for the Russian and American space programs in long range communications. The West has produced geostationary satellites placed at just sufficient distance from the Earth so as to have a 24-hour orbital period and to remain fixed above the equator at a particular longitude. This would appear to be the ideal solution and was first proposed by Arthur C. Clarke in the 1940s in an article in *Wireless World*.

The Russian approach has been, or was in the 1970s, to send craft into highly elliptical orbits but again of 24-hour orbital periods. As has been demonstrated, such satellites would be constantly changing their velocity according to their current distance from Earth and would remain moderately stationary (so as not to need a tracked aerial) for only a few hours per day, whilst at the point of apogee (the most remote part of the orbit from Earth). Many permutations can be tried on your Spectrum.

```
9 REM ***********************
10 REM Earth Orbit
11 REM ***********************
15 BORDER 0: PAPER 0: INK 9: CLS
20 LET se=7912/2: LET hw=32.12
40 LET dn=.68181818: LET z=1
70 LET g=hw*dn*se*se: LET t=0
80 INPUT "Height above Earth in miles",eh
90 INPUT "Velocity in miles/min",sp: IF sp>999 THEN GO TO 90
100 LET tm=2/z: LET h=tm
120 LET w=sp: LET x=eh+se
140 LET y=0: LET v=0
```

```
160 FOR n=0 TO 255 STEP 2
170 PLOT INK 5;140,n-100
180 PLOT INK 5;n,40: NEXT n
190 PRINT AT 16,0;"-x";AT 16,30;"+x";AT 3,16;"+y"
200 PRINT AT 19,24;"start";TAB 20;"hgt=";eh;"ml";TAB 20;"vel=";INT w;"ml/mn"
210 CIRCLE INK 4;140,40,30*z
220 LET r=x: LET s=y
230 LET h4=h/4
240 LET x=x+h4*v: LET y=y+h4*w
260 GO SUB 440
270 LET x=r: LET y=s
280 LET h2=h/2
290 LET v=v+h2*b: LET w=w+h2*c
310 GO SUB 500
320 FOR t=0 TO 4e6 STEP tm
330 LET x=x+h*v: LET y=y+h*w
350 GO SUB 440
360 IF d>se THEN GO TO 390
370 FOR n=45 TO -30 STEP -15: BEEP .5,n-30: NEXT n: PRINT INK 3;AT 0,0;"Velocity
too low - CRASH"'"in ";t;" min"
380 GO TO 610
381 REM **********************
390 LET v=v+h*b: LET w=w+h*c
410 GO SUB 500: NEXT t
440 LET e=x*x+y*y
450 LET d=SQR e: LET a=-g/e
470 LET b=a*x/d: LET c=a*y/d
490 RETURN
500 PRINT AT 19,2;"time=";t;" min "
510 PRINT "x axis=";INT x;" mls "
517 REM **********************
520 PRINT "y axis=";INT y;" mls "
521 REM **********************
530 LET xx=x/125*z+140
540 LET yy=y/125*z+40
550 IF x<0 AND y255 OR xx175 THEN LET z=z/2: CLS : PRINT INK 3;"Rescale": GO TO 100
569 REM **********************
570 PLOT BRIGHT 1; INK 6;xx,yy
571 REM **********************
```

```
580 RETURN
600 PRINT INK 6;AT 0,0;"Half orbit plot- period=";t*2;"mn";AT 1,23;"=";INT
(t/.3)/100;"hr";AT 2,23;"=";INT (t/.3/24)/100;"days"
609 REM ***********************
610 PRINT #0; INK 5;"Press 'C'=COPY 'N'=Next orbit": PAUSE 0
620 IF INKEY$="c" OR INKEY$="C" THEN COPY : INPUT "": GO TO 610
630 RUN
631 REM ***********************
9990 SAVE "Eorbit" LINE 1
```

Satellite Orbit

This program is based on the Earth Orbit program, but is extended to cope with modelling a satellite orbit around any planet, or even our Moon.

In addition to INPUTting the start height and initial velocity of the satellite, it is necessary to ENTER details of the planet itself, ie the diameter (in kilometres) and relative mass compared to Earth (Earth = 1). This information is included in the initial display (Figure 4.4) but you need not be confined to it. Various data should be tested including exotic features like tiny diameters and excessive mass. You should not expect the program to cope with all permutations, particularly for the 'massive' planet. Insufficient computation plus very rapid orbiting may produce strange results.

Figure 4.4
Sample worlds for a satellite to orbit.

The program recognises if a satellite is in a 'parabolic' or 'hyperbolic' orbit (open-ended) and thus lost to space.

Similarly the Spectrum BEEPs a warning if the initial injection velocity proves too low and the satellite crashes on the surface of the planet.

The program specifically recognises one planet (Saturn) and DRAWs in the encircling ring system if this name is ENTERed in the N$. For the sake of clarity, it is advisable to place your satellite outside the Saturnian ring system.

The CLS command is not used, except against a new planet, and so multiple orbits can be overlaid upon one another until you are satisfied with a particular orbit. The information contained at the corners of the display, starting with the top left, is planet data, orbital period individually in minutes, hours and days and, at the bottom of the screen, the elapse time in minutes, x and y relative coordinate positions and the initial height and velocity in kilometres. An option to COPY the screen to the ZX printer is included. See Figures 4.5 and 4.6 for examples.

The program is an excellent way of gaining insight into the behaviour of, for example, the moons of Jupiter or Saturn, where a stronger gravitational field operates than on Earth.

To demonstrate, Jupiter's moon Io is a similar distance from the centre of Jupiter as is our Moon from the centre of the Earth. Io orbits Jupiter in 1.8 days and our Moon takes 27.3 days (mean sidereal periods) to orbit the Earth. Try testing some set examples with data from a text book but do not expect any great accuracy as the results should be regarded as informative rather than precise.

Figure 4.5

The program recognises 'Saturn' and draws a ring system. Unfortunately, this satellite is launched from within the rings with insufficient velocity and crashes immediately on to the planet.

Figure 4.6

Third time lucky with these orbit attempts around a small mythical world. One satellite is lost to space, one crashes on the far side of the planet.


```
10 REM Satellite Orbit
20 GO SUB 1000
30 LET n$=a$(n)
40 LET k=d(n): LET m=m(n)
50 IF n150000 OR k9999 THEN GO TO 140
170 LET disk=(rd/125*16)*z
180 IF disk>40 THEN LET z=z/2: GO TO 170
190 IF n$="Saturn " AND disk*2.5>40 THEN LET z=z/2: GO TO 170
200 PRINT PAPER 4;AT 1,0;n$
210 PRINT INK 5;"diam=";k;"km"'"grav=";m;"xE"
220 LET tm=.125/z: LET h=tm
230 LET w=vl: LET x=hi+rd
240 LET ho=x: LET y=0: LET v=0
250 FOR n=0 TO 255 STEP 2
260 PLOT 140,n-100
270 PLOT n,40: NEXT n
280 PRINT AT 16,0;"-x";AT 16,30;"+x";AT 3,16;"+y"
290 PRINT AT 19,23;"initial";AT 20,19;"hgt=";hi;"km";AT 21,19;"vel=";INT w;"km/mn"
```

```
300 INK 5
310 FOR j=0 TO PI/2 STEP .1
320 LET px=INT (SIN j*disk)
330 LET py=INT (COS j*disk)
340 PLOT 140-px,40+py
345 DRAW 0,-py*2
350 PLOT 140+px,40+py
355 DRAW 0,-py*2: NEXT j
360 CIRCLE 140,40,disk: INK 9
380 IF n$="Saturn " THEN FOR n=1.5 TO 2.5 STEP .2: CIRCLE 140,40,disk*n: NEXT n
390 LET r=x: LET x=x+h/4*v
400 LET s=y: LET y=y+h/4*w
410 GO SUB 580
420 LET x=r: LET v=v+2/4*b
430 LET y=s: LET w=w+h/4*c
440 GO SUB 610
450 FOR t=0 TO 4e4 STEP tm
460 LET x=x+h*v: LET y=y+h*w
470 GO SUB 580
480 IF x>0 AND y>2*ho THEN PRINT FLASH 1; PAPER 2;AT 0,0;"Velocity too high-LOST to
space ": GO TO 510
490 IF d>rd THEN GO TO 530
500 PRINT FLASH 1; PAPER 3;AT 0,0;p$;AT 0,0;"Velocity too low-CRASH in ";t;"mn "
505 PLOT FLASH 1;xx,yy
510 FOR n=45 TO -30 STEP -15: BEEP .5,n-30: NEXT n
520 PAUSE 50: PLOT FLASH 0;xx,yy: GO TO 740
530 LET v=v+h*b: LET w=w+h*c
540 PAPER 0
550 GO SUB 610
560 NEXT t
570 GO TO 130
580 LET e=x*x+y*y: LET d=SQR e
590 LET a=-g/e: LET b=a*x/d
600 LET c=a*y/d: RETURN
610 PRINT AT 19,0;"time=";t;"mn "
620 PRINT "x ax=";INT x;"km "
630 PRINT "y ax=";INT y;"km "
640 LET xx=x/125*16*z+140
650 LET yy=y/125*16*z+40
```

```
660 IF x<0 AND y<0 THEN GOTO 690
670 IF xx>255 OR xx<-255 OR yy>175 THEN LET z=z/2: CLS : PRINT AT 0,0;p$;AT
0,0;"Re-scale=";z: GO TO 170
680 PLOT xx,yy
685 RETURN
690 FOR n=0 TO .5 STEP .02: BEEP n/4,n/5: NEXT n
700 LET hr=INT (t/.3)/100: LET dy=INT (t/.3/24)/100
710 PRINT PAPER 6;AT 0,0;"Half orbit plot-period=";t*2;"mn"
720 PRINT AT 1,22;" ";AT 2,22;" "
730 PRINT AT 1,22;"=";hr;"hr";AT 2,22;"=";dy;"dy"
740 PRINT #0; INK 9; PAPER 4;"z=COPY:p=new Planet:o=new Orbit "
750 PAUSE 0: IF INKEY$="o" THEN PRINT AT 0,0;p$: GO TO 130
760 IF INKEY$="z" THEN LPRINT : COPY : GO TO 750
770 RUN
1005 BORDER 0: PAPER 0: INK 9: CLS : RESTORE
1010 LET zz=1: DIM p$(32)
1015 DIM a$(12,7): DIM d(12): DIM m(12)
1020 DATA "Moon",3476,.165
1030 DATA "Mercury",4878,.377
1040 DATA "Venus",12104,.90
1050 DATA "Earth",12756,1
1060 DATA "Mars",6794,.379
1070 DATA "Pallas",532,.022
1080 DATA "Jupiter",142800,2.69
1090 DATA "Saturn",120000,1.19
1100 DATA "Uranus",52000,.93
1110 DATA "Neptune",48400,1.22
1120 DATA "Pluto",3000,.2
1125 PRINT " PAPER 6;" Satellite Modelling Program ""
1130 PRINT PAPER 5;"No Name Diam(km) Mass(Earth=1)"
1140 FOR n=1 TO 11: READ a$(n),d(n),m(n)
1150 PRINT (" " AND n<10);n;" ";
1155 PRINT a$(n);TAB 12;d(n);TAB 23;m(n)
1160 NEXT n: PRINT PAPER 6;"12 ??? ??? ??? "
1170 PRINT '"Enter Planet no"
1180 INPUT n: IF n12 THEN GO TO 1180
1200 RETURN
```

Chapter 5 – Solar System Orbits (Orrery)

Astronomy and orbits go together like strawberries and cream. Banish orbits and the whole universe as we know it would literally collapse! It is the perpetual motion of moon about planet, planet about star, star about galaxy and so on, that gives the universe its form and existence. And we must not forget that the greatest galaxies in the universe are themselves composed of the smallest things in creation — electrons in orbit about atomic nuclei.

Orbits are the embodiment of perpetual motion — a concept that has fired man's imagination and remained resolutely beyond his grasping hand as he apparently seeks something for nothing. Man's smoothest jet engine is a very crude mechanism in comparison with the billions of orbits of our Moon about the Earth — the oily precision of which would make an engineer green with envy. But then nature is not impetuous and of course has time — more precisely, all the time in the universe — to get it right.

In the final analysis, orbits mean survival: survival for a feeble body from the gravitational bully in the neighbourhood, eager to swallow up any candidate that gets too close. The orbit is the fine tightrope that the lesser body follows to keep its distance and is perfectly balanced by the gravitational pull of the greater body that refuses to release the bondage. In the case of the Sun, its mass (to which its gravity is proportional) is 1000 times greater than all the planets and moons put together. They have no choice but to orbit the Sun, and lesser bodies no choice but to orbit them. The Spectrum and its graphics makes modelling some of these orbits easy.

Orrery

In 1715 the Earl of Orrery had made for himself the mechanical model of the solar system that bears his name today. At that time, only the six planets known from antiquity (Mercury through to Saturn) were included on his model. Uranus, Neptune and Pluto had yet to be discovered. Each planet was a globe suspended on a horizontal wire from beneath a central globe representing the Sun — the length of the wire representing the radius of the orbit. All this was contained in a superbly crafted cabinet with a handle to crank the planets into action.

Figure 5.1
An Orrery for the Earth-like planets

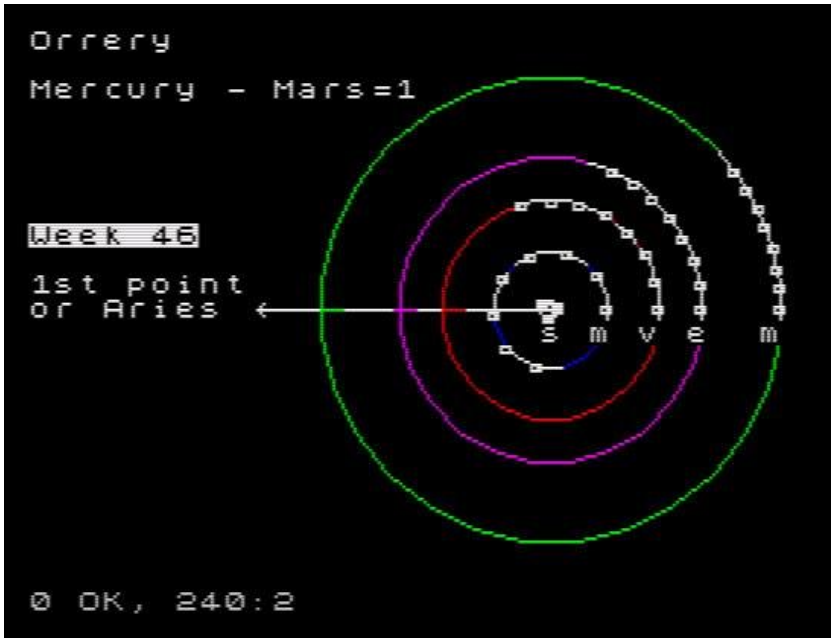
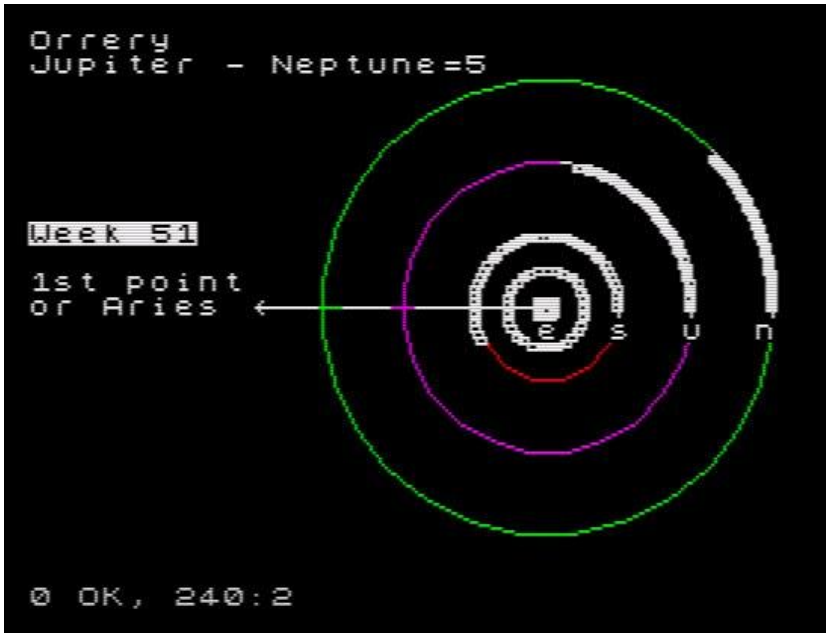


Figure 5.2
An Orrery for gas giant planets



The program
The following program is a computer version of that device and simulates the scale and relative movement of the planets about the Sun. The planets Mercury to Mars and Jupiter to Neptune are included using two separate scales, which may be selected by INPUT of either 1 or 5.

Included on the screen is a timer which counts in weeks as the planets advance along their respective orbits from a common starting line. You will notice how much slower the progress is for the outer planets which, in reality, are

not subjected to the same high gravitational field as, for example, Mercury. Two sample COPYS are shown in Figures 5.1 and 5.2

Line 25 contains the DATA for the orbit radii which is DIMensioned and READ into the a(f) array in Line 50. Variables h and g are the vertical and horizontal coordinate PLOT positions for the Sun and centre of the orbits. Lines 60 and 70 PRINT the two options for selection and Line 90 economically overprints the one not selected with the b$ of 24 blank spaces. The remaining PRINT line serves to title the display that follows without the need to use the CLS command. Lines 120 to 130 use a FOR/NEXT n loop to CIRCLE the selected orbits. Note how the value of variable b affects the FOR/NEXT n loop:

if b = 1 then FOR n = 1*1 (ie l)TO 1 + 4(ie5)
or if b = 5 then FOR n = 1*5 (ie 5) TO 5 + 4(ie9)

A similar FOR/NEXT n loop is used in Lines 180 to 240 to CIRCLE the planets in orbit. The STEP in the FOR/NEXT d loop in Line 150 controls the length of the program RUN.

The screen display
The display is more effective if shown on a black screen using the direct command:

BORDER 0: PAPER 0: INK 7: CLS: RUN

The orbits are sufficiently far apart so as not to share the same character square. Each can be coloured for greater
clarity by amending Line 130 to:

CIRCLE INK n – (b AND b = 5) + 1; g,h,a(n): NEXT n

```
10 BORDER 0: PAPER 0: INK 7: CLS : PRINT "Orrery"
25 DATA 19,36,50,76,3,13,24,48,75
30 BRIGHT 1: DIM b$(24)
40 LET h=83: LET g=171
50 DIM a(9): FOR f=1 TO 9: READ a(f): NEXT f: RESTORE
60 PRINT "Jupiter – Neptune=5"
70 PRINT "Mercury – Mars=1"
80 INPUT "Select 1 or 5",b: IF b1 AND b5 THEN GO TO 80
90 PRINT AT b/5+1,0;b$
100 PRINT AT 10,0;"1st point"'"or Aries <";TAB 21;"*"
110 PLOT g-5,h: DRAW -90,0
120 FOR n=b TO b+4
130 CIRCLE INK n-(b AND b=5);g,h,a(n): NEXT n
140 PRINT AT 12,21;("s m v e m" AND b=1)+("ej s u n" AND b=5)
150 FOR d=1 TO 400 STEP 50/b
```

```
170 PRINT FLASH 1;AT 8,0;"Week ";INT (1+d/50*2^2.7)
180 FOR n=b TO b+4
190 LET v=1/a(n)^1.4*100
200 LET e=d/400*v*PI
210 LET c=g+a(n)*COS e
220 LET s=h+a(n)*SIN e
230 CIRCLE INT c,INT s,1.5
240 NEXT n: NEXT d
```

Bode's Law

In 1772, the German astronomer Johann Bode demonstrated that a simple mathematical progression — eg 0 + 4, 3 + 4, 6 + 4, 12 + 4,… — could explain the average distance of successive planets from the Sun — where Earth had a value of 10 (6 + 4). At the time, only six planets were known to exist, not surprisingly, they fitted reasonably well with what came to be known as 'Bode's Law'.

The discovery of Uranus beyond Saturn in 1781 and the minor planet Ceres, between Mars and Jupiter, in 1801 confirmed the Law by falling nicely into place. However, the discovery of Neptune in 1846 and Pluto in 1930 did not uphold the Law — Pluto occupying the orbit allocated by Bode for Neptune as the following short program demonstrates. Perhaps you can improve on Bode to provide a better explanation?

The program

The upper half of the screen LISTs and DRAWs semi-orbital arcs calculated from the variable 'bode' (z + 4); the lower half the actual values as READ from DATA. The display is scaled to maximum size for the DRAW routine where PI = semi-circle, and the program stops with an error message (integer out of range) in attempting to DRAW the final arc for Pluto according to Bode's Law. Thus Neptune's actual orbit, in the lower display, fits the space between Uranus and Neptune as allocated by Bode and shown in the upper display. Study of the numerical values displayed confirms this as Figure 5.3 shows.

Figure 5.3

Note: This program, in demonstrating Bode's Law, refers to average distances from the Sun. Most planetary orbits are not neat circles but slightly elliptical — highly so in Pluto's case causing it to overlap Neptune's orbit. The program Pluto's Orbit, later in this chapter, fully explains this situation.

```
10 REM Titius-Bode's Law
20 LET x=185: LET y=88
30 LET z=0: DIM a$(32*11)
40 PRINT PAPER 6;a$; PAPER 1;a$: FLASH 1
50 PRINT AT 0,1;"Titius-Bode's Law"
60 PRINT AT 11,1;"Actual dist"
70 FLASH 0: PAPER 6: INK 9
80 FOR n=1 TO 10
90 LET bode=z+4
100 READ p$,a
110 PRINT AT n,0;p$;TAB 8;bode
120 PRINT AT n+11,0; PAPER 1;p$;TAB 8;a
130 PLOT x-a/6,y
140 DRAW a/3,0,PI
150 PLOT x-z/6,y
160 DRAW z/3,0,-PI
170 LET z=z+z
180 IF n=1 THEN LET z=3
190 NEXT n
200 DATA "Mercury",3.9,"Venus",7.2,"Earth",10,"Mars",15.2,"Ceres",
```

```
26.5,"Jupiter",52,"Saturn",95.4,"Uranus",192,"Neptune",301,"Pluto",
395
```

Kepler's Orbits

Johannes Kepler, between 1609 and 1618, found that planets, and in fact all orbiting bodies, follow three laws of planetary motion:

1. The orbit of the planet is an ellipse with the Sun at one focus of that ellipse.
2. The planet moves in its orbit at such velocity that its radius vector (the angle as seen from the Sun to any point on the orbit) sweeps out an equal area in a similar time interval — see Figures 5.4 and 5.5.
3. The orbital period squared is proportional to the mean (average) distance from the Sun cubed.

The following program demonstrates the first two of Kepler's Laws of Planetary Motion by PLOTting elliptical orbits with the spacing between the individual PLOT positions at equal time intervals. The PLOTted pixel (representing the orbiting planet) is seen to 'accelerate' as it passes close to the Sun (the small CIRCLE in the centre of the screen) particularly as the orbits become more elliptical (higher eccentricity) against the values INPUT for initial velocity.

These INPUT values are purely arbitrary. Effectively, the higher the value the more circular the orbit, the lower the value (Figure 5.6) the more eccentric the orbit and the closer the planet passes to the Sun. The first position PLOTted marks the point of 'aphelion', ie the most remote part of the orbit from the Sun. The point of the orbit closest to the Sun is called 'perihelion'.

Figure 5.4: Kepler's 1st and 2nd Laws of Planetary Motion

A planet PE (shown in a highly elliptical orbit for clarity) sweeps out an equal area (shaded) in an equal time interval — its velocity constantly changing according to its current distance from the Sun.

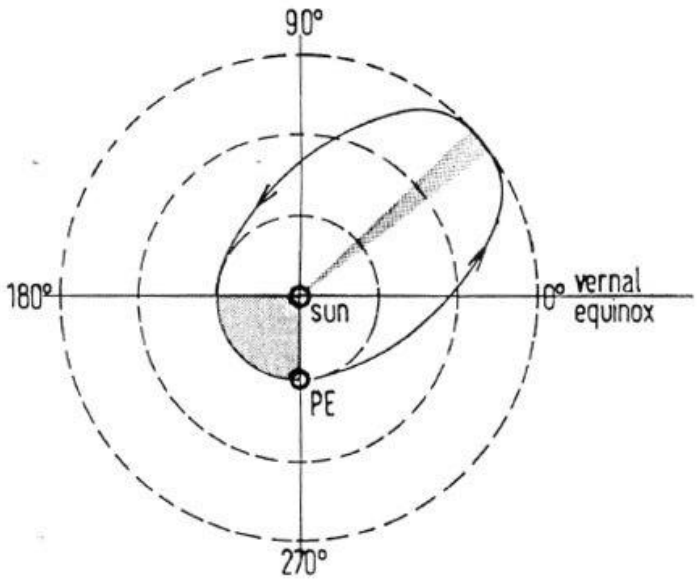
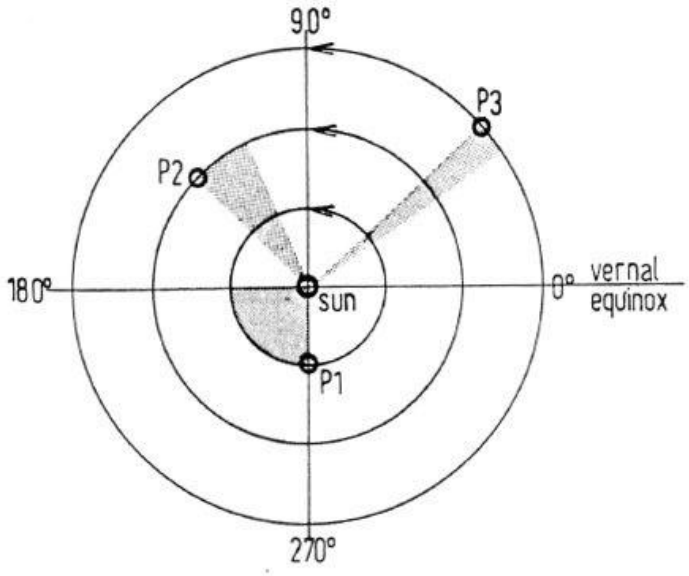


Figure 5.5: Kepler's 3rd Law of Planetary Motion

Each planet sweeps out an area (shaded) in an equal time interval proportional to the orbit radius. In this simplified solar system with the orbits equally spaced, P1 covers a complete quadrant whilst P2 covers 0.35 of a quadrant ($1/\sqrt{2^3}$) and P3 covers 0.19 of a quadrant ($1/\sqrt{3^3}$).



If an INPUT of >12 is ENTERed then odd things start happening with the PLOTted results. These have nothing to do with Kepler's Laws — but with short-comings of the program. They have, however, been deliberately left in to demonstrate three effects that can happen to a body in orbit about, or passing close to, a more massive planet.

An INPUT of 12 will cause rotation of the whole orbit so that the aphelion position (and also the perihelion position although this is not obvious) moves in a clockwise direction. This effect mimics the orbit rotation of any planet but even in the case of Mercury, the planet closest to the Sun and with the highest velocity of all the planets, it is only detectable over hundreds of complete orbits. (See Figure 5.7)

Figure 5.6

Lower INPUT values produce more elliptical orbits.

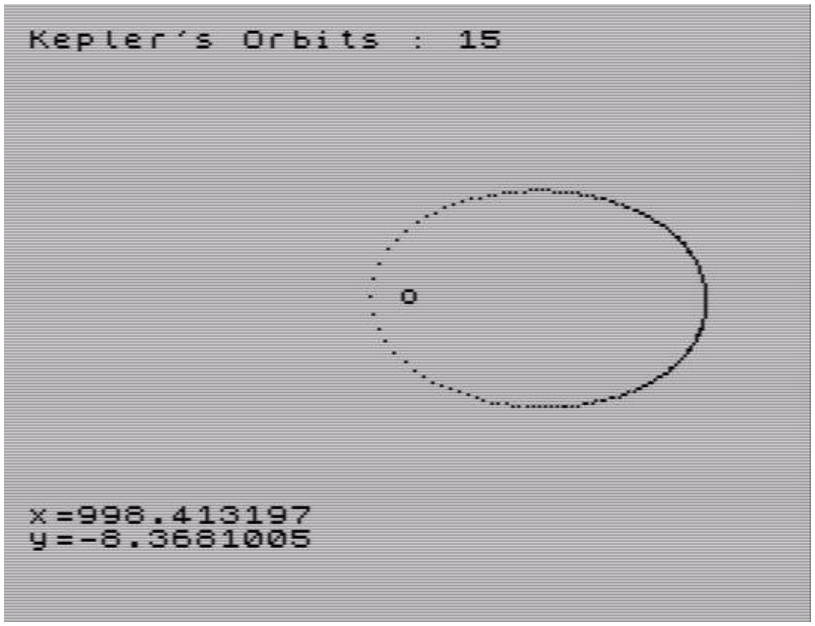
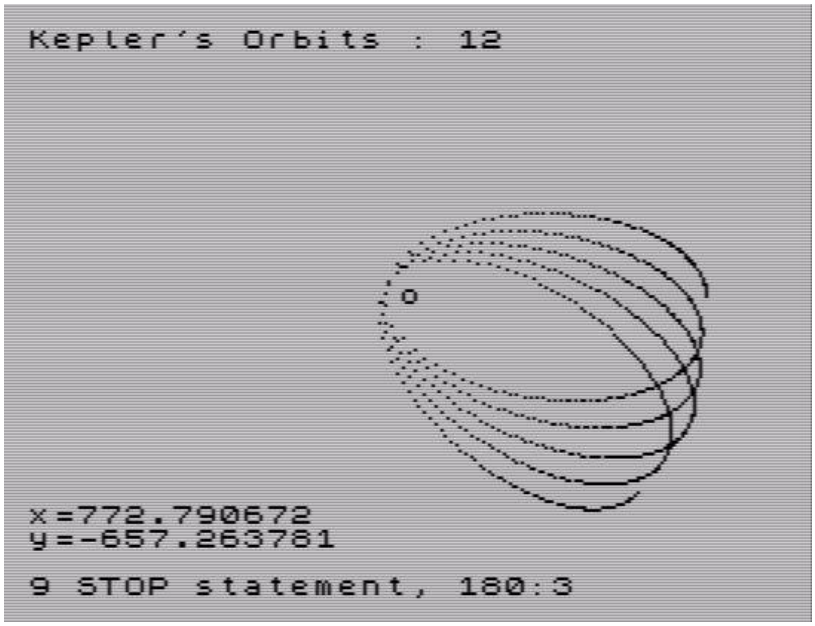


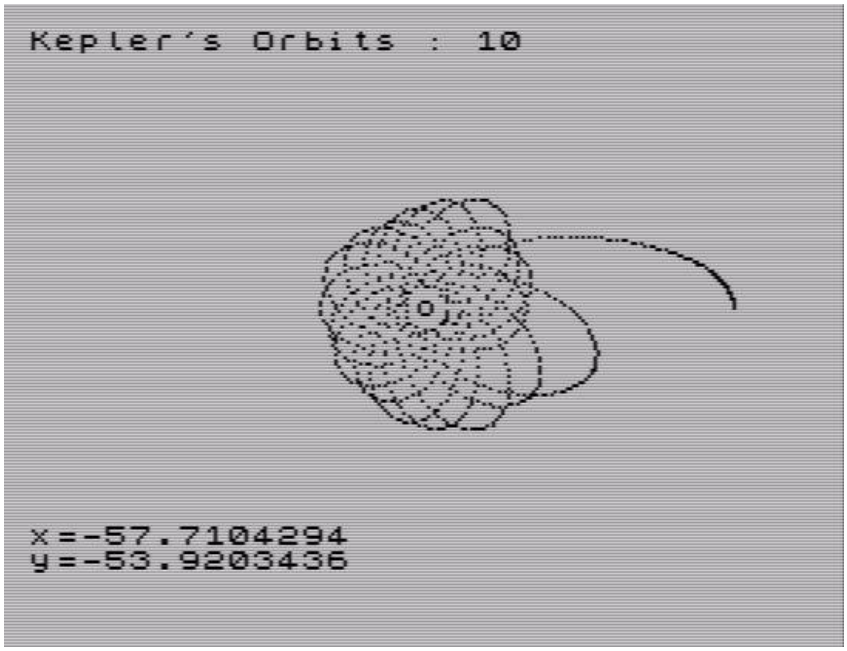
Figure 5.7

Rotation of orbit.



An INPUT of 10 will cause the capture of the body into a smaller and faster orbit. It is believed that many comets have been captured into smaller orbits by passing sufficiently close to the massive planet Jupiter and, to a lesser extent, Saturn. This obviously assumes a third body which our program does not contain but it remains of interest to demonstrate an effect. (See Figure 5.8)

Figure 5.8
Capture of larger orbit into a smaller orbit.



An INPUT of 7 or less will cause the planet to pass so close to the Sun that the planet is ejected out of the solar system in what is called a sling-shot manoeuvre. In reality no planet would be subjected to this indignity — if such a thing ever occurred it would have been billions of years ago — there is no place for rebel planets today. However space scientists use this effect to send spacecraft from one planet to the next in a game of planetary billiards, poaching a little of the host planet's gravitational field to accelerate the craft to high velocities impossible to achieve by rocket power from Earth.

An explanation
Why does this program produce these effects? Quite simply because the program does not calculate with sufficient accuracy the position of the planet when close to the Sun. The minutest error is added to the next computation producing the effects displayed. The situation is not assisted by the fact that, at the point of perihelion, the vertical or y coordinates as PLOTted pass from positive to negative values. All in all, a case of using a potential disadvantage to good effect, even if not good mathematics!

Amending the program
Once the program has been RUN a few times, try amending (one at a time) some of the variables. The most important variable to change is h, which controls the STEP intervals in the FOR/NEXT loops to RUN the program.

If it is made smaller then the orbit will be PLOTted with greater accuracy, particularly at perihelion position. However it will now PLOT incredibly slowly whilst the planet is near aphelion. You change the value — you take your choice.

This particular program is used a number of times elsewhere in this book — usually with modifications to suit particular purposes.

```
10 PRINT "Kepler's Orbits : ";
30 INPUT "Velocity (5 to 30): ";w
40 PRINT w: CIRCLE 128,85,2
50 LET h=.5: LET g=1000000
60 LET x=g/1000: LET y=0
70 LET i=h/4: LET v=0
80 LET r=x: LET s=y
90 LET x=x+i*v: LET y=y+i*w
100 GO SUB 200
110 LET x=r: LET y=s: LET o=h/2
120 LET v=v+o*b: LET w=w+o*c
130 GO SUB 300
140 FOR t=0 TO 400 STEP h
150 LET x=x+h*v: LET y=y+h*w
160 GO SUB 200
170 LET v=v+h*b: LET w=w+h*c
180 GO SUB 300: NEXT t: STOP
200 LET e=x*x+y*y: LET d=SQR e
220 LET a=-g/e: LET b=a*x/d
240 LET c=a*y/d: RETURN
300 PLOT x/10+128,y/10+85
305 PRINT AT 20,0;"x=";x'"y=";y
310 RETURN
```

Orbit Foci

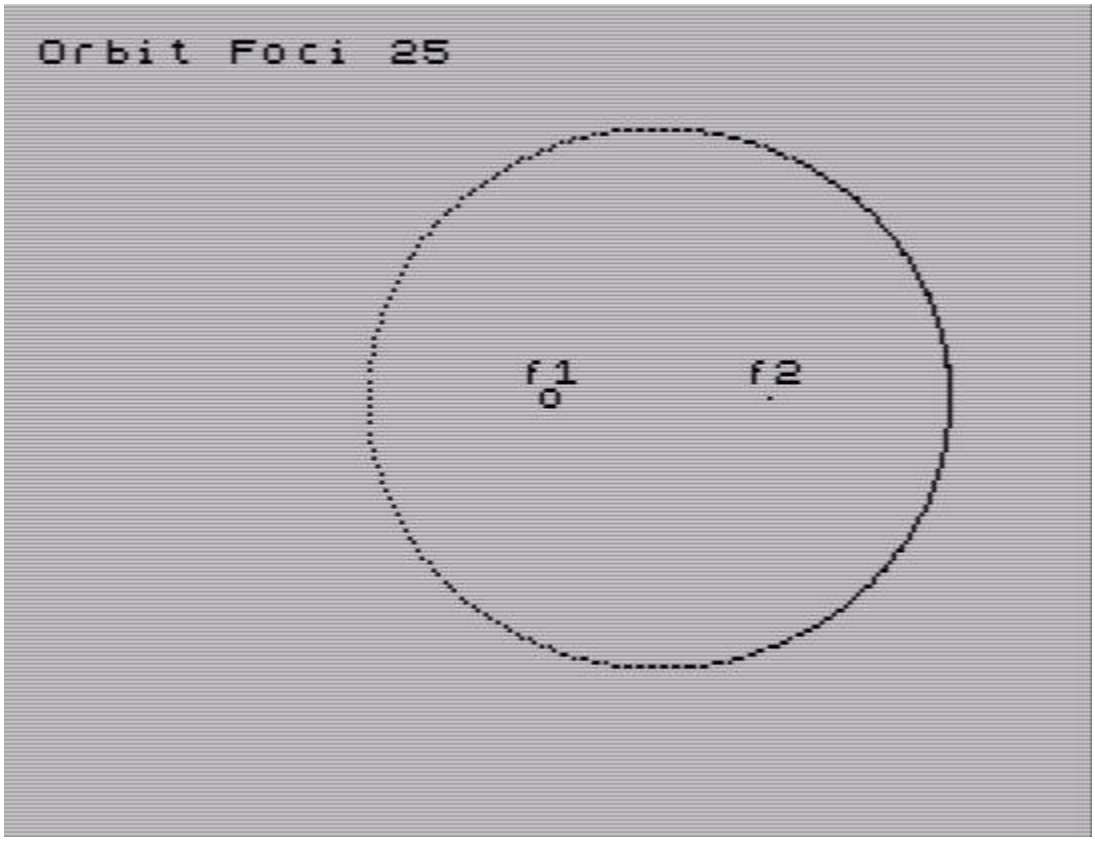
In the previous program, Kepler's Orbits, reference is made to the Sun being at one focus of an elliptical orbit. Where is the second or 'empty' focus as it is called?

Quite simply it is at an equal distance measured from the Sun to the perihelion position (the nearest point in orbit to the Sun) but transferred to the aphelion position (remotest point in orbit to the Sun). If you have already keyed in the Kepler's Orbits program, then the minor modifications from Line 300 in this program are all that is

necessary before RUN-ning the revision. This version will PLOT the second focus position once a half-orbit is PLOTted and the position of perihelion is known. Figure 5.9 demonstrates.

Figure 5.9

The program plots the second, or empty, focus (f2) of any selected elliptical orbit. The Sun is at the first focus (f1).



This program is quite important in demonstrating that the Spectrum, via the program, is PLOTting a true elliptical orbit. There is a simple test to check results. RUN the program and measure, either on the video screen or a paper COPY if you have access to the ZX printer, the total distance around a triangle formed between any point on the orbit and the two foci. No matter what point you choose on a given orbit the total distance will be the same.

You are following the same procedure, in reverse, as the gardener who insists on making perfect oval (ie elliptical) flower beds in a lawn. He uses two stakes (representing the two foci) and a loop of string about them drawn taut (representing your triangle) to trace out the shape. Quite obviously the total length of the string remains constant, as should your measurements.

```
10 PRINT "Orbits Foci ";
30 INPUT "Velocity (13 to 30): ";w
40 PRINT w;AT 10,15;"f1": CIRCLE 128,85,2
50 LET h=.4: LET g=1000000
60 LET x=g/1000: LET y=0
```

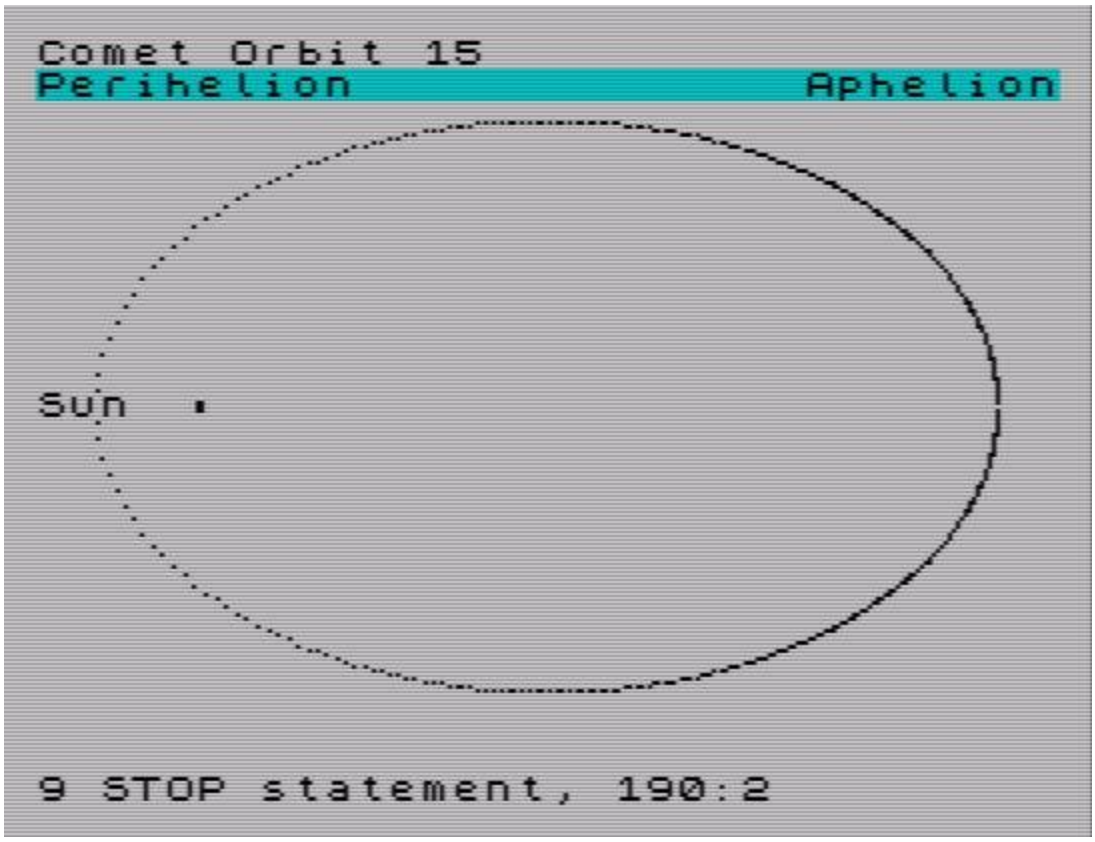
```
70 LET i=h/4: LET v=0
80 LET r=x: LET s=y: LET z=0
90 LET x=x+i*v: LET y=y+i*w
100 GO SUB 200
110 LET x=r: LET y=s: LET o=h/2
120 LET v=v+o*b: LET w=w+o*c
130 GO SUB 300
140 FOR t=0 TO w*7 STEP h
150 LET x=x+h*v: LET y=y+h*w
160 GO SUB 200
170 LET v=v+h*b: LET w=w+h*c
180 GO SUB 300: NEXT t: STOP
200 LET e=x*x+y*y: LET d=SQR e
220 LET a=-g/e: LET b=a*x/d
240 LET c=a*y/d: RETURN
300 PLOT x/10+128,y/10+85
310 IF x=-50 AND x>z THEN PLOT 228+x/10,85: PRINT AT 10,(228+x/10)/8-1;"f2"
320 LET z=x
330 RETURN
```

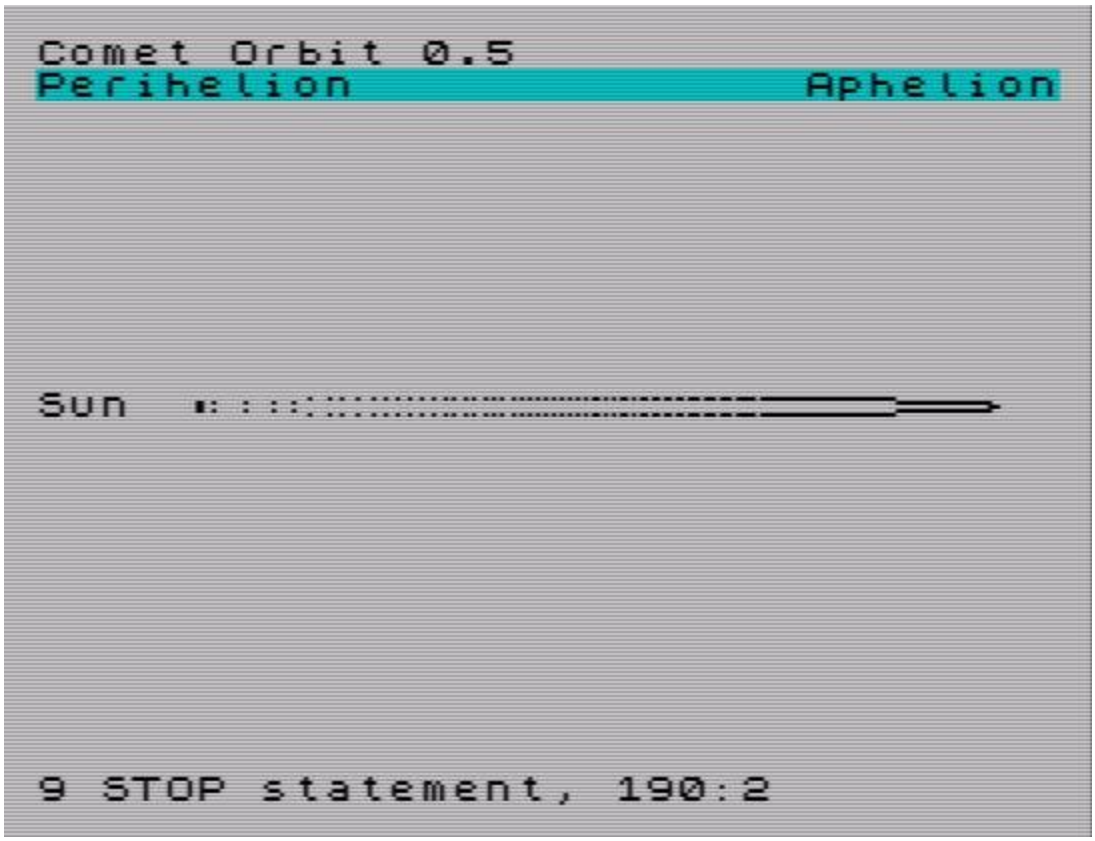
Comet Orbit

This program is the second variation of Kepler's Orbits. This time we're dealing with highly eccentric orbits which can usually be ascribed to comets and meteor streams. It is generally recognised that regular meteor displays (shooting stars in the Earth's atmosphere), like the Perseids in August of each year, are the debris from comets, strewn along each comet's orbit.

The program allows INPUT values from 0.5 (a highly eccentric orbit, essentially of two parallel lines) to 17 (a full oval) filling the Spectrum screen (see Figure 5.10). To avoid the erroneous if interesting effects of calculating the cometary position close to the Sun, only half the orbit is PLOTted. The lower half of the orbit is a mirror image of the upper portion — Lines 200 and 210 coping with their respective halves. The program STOPs when y becomes negative (once the half orbit is computed).

Figure 5.10
Two extremes of cometary orbit in plan view from full oval to highly eccentric as sample INPUTS.


Comet Orbit 15
Perihelion
Aphelion
Sun
9 STOP statement, 190:2


Comet Orbit 0.5
Perihelion
Aphelion
Sun
9 STOP statement, 190:2

The program demonstrates very effectively Kepler's 2nd law of Orbital/ Planetary Motion (see the Kepler's Orbits program) in the way in which a comet spends most of its time moving very slowly whilst remote from the Sun and only bursts into activity at perihelion passage, as it is called.

```
10 PRINT "Comet Orbit ";
30 INPUT "Value (.5 to 17): ";w
35 PRINT w' PAPER 5;"Perihelion
Aphelion"
40 PRINT AT 11,0;"Sun": CIRCLE 40,83,1
50 LET h=.2: LET g=1e6
60 LET x=g/1e3: LET y=0
70 LET i=h/4: LET v=0
80 LET r=x: LET s=y: LET z=0
90 LET x=x+i*v: LET y=y+i*w
100 GO SUB 160
110 FOR t=0 TO 300
120 LET x=x+h*v: LET y=y+h*w
130 GO SUB 160
140 LET v=v+h*b: LET w=w+h*c
150 GO SUB 190: NEXT t: STOP
160 LET e=x*x+y*y: LET d=SQR e
170 LET a=-g/e: LET b=a*x/d
180 LET c=a*y/d: RETURN
190 IF y<0 THEN STOP
200 PLOT 40+x/5,y/5+83
210 PLOT 40+x/5,-y/5+83
220 RETURN
```

Halley's Comet

Halley's Comet is without doubt the most famous comet of all time and, as a visit to our part of the solar system is due shortly, a program would not be inappropriate.

1066 and all that

Edmund Halley (1656 – 1742) did not discover this comet but was the first to notice that the bright comets seen in 1531,1607 and 1682 had practically identical orbital data and were one and the same object reappearing in the skies about every 75 years.

Halley's Comet can now be traced back to 611 BC, via Chinese records, but perhaps the most famous reference of all in European history is its depiction in the Bayeux Tapestry of 1066 with the inscription of INTIMIRANT STELLA. Every return of the Comet since King Harold's reign has been recorded and this is most unusual as the longevity of comets is measured in hundreds rather than thousands of years. This indicates that Halley's Comet is a substantial body able to survive repeated visits to the inner solar system and the relatively great heat radiated upon it from the Sun.

You should be asking the question: How do comets survive repeated crossings of all the planets without collison? Well, comets (or rather, the survivors after over 5000 million years) have learned to avoid the orbital plane which all planets occupy with highly-inclined orbits. The comet crosses this danger zone for only a few days each perihelion passage.

The program

The program depicts one complete orbit of Halley's Comet beginning in 1948 _ the year the Comet started its current journey towards both Earth and Sun from beyond the orbit of Neptune. The Comet will return to this aphelion position again in about 2023. The perihelion passage occurs on 10th February 1986 and the program PAUSEs briefly at this point. (See Figure 5.11.) During mid-November 1985 the comet is a binocular object below the Pleiades.

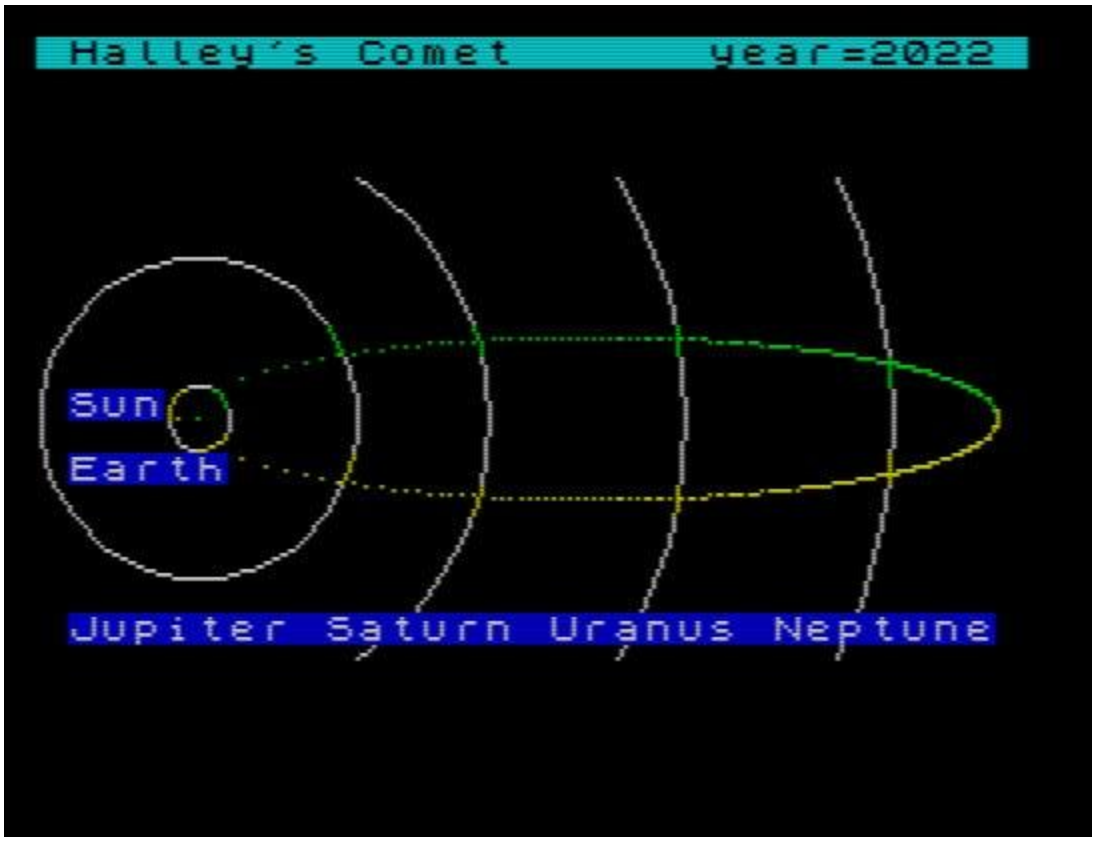
This program is a variation of the Comet Orbit program, but uses one specific orbit produced by the line LET w = 4.5. Again, only one half of the orbit is computed and PLOTted — the inward journey — but each x and y coordinate position is entered into two arrays, x(t) and y(t). These are then used in a second FOR/NEXT loop to PLOT the Comet's journey back into deep space.

It is necessary to add PAUSE 10 to this second FOR/NEXT loop to slow the PLOTting down to the same speed as the first loop. This indicates the rapidity with which the Spectrum can PLOT pixel positions once the actual position has been computed and SAVEd in an array. Try pressing any key to cancel the PAUSE statement to see what happens.

Screen display

At the top of the screen is denoted the current year against the Comet's progress — the Comet itself is PLOTted in a different coloured pixel for the inward and outward journeys (for clarity) using inverse graphics on a black screen (BORDER 0: PAPER 0: INK 9).

Figure 5.11
The path of Halley's comet over a 75-period. Closest approach to the Sun occurs in February 1986.




```
10 REM Halley's Comet
20 BORDER 0: PAPER 0: INK 7: CLS : PAPER 5: INK 9
30 PRINT " Halley's Comet year= "; FLASH 1;" "
40 PAPER 1
50 PRINT AT 11,1;"Sun"
60 PLOT 40,80: GO SUB 390
70 PAPER 5
80 LET w=4.5
90 DIM x(170): DIM y(170)
100 LET h=.212: LET g=1e6
110 LET x=g/1e3: LET y=0
120 LET i=h/4: LET v=0
130 LET r=x: LET s=y: LET z=0
140 LET x=x+i*v: LET y=y+i*w
150 GO SUB 230
160 FOR t=1 TO 170
170 LET yr=1948+INT (t/4.5)
180 PRINT AT 0,26;yr
190 LET x=x+h*v: LET y=y+h*w
```

```
200 GO SUB 230
210 LET v=v+h*b: LET w=w+h*c
220 GO SUB 260: NEXT t: STOP
230 LET e=x*x+y*y: LET d=SQR e
240 LET a=-g/e: LET b=a*x/d
250 LET c=a*y/d: RETURN
260 PLOT INK 4;40+x/5,y/5+80
270 LET x(t)=x/5: LET y(t)=y/5
280 IF y<0 THEN GO TO 300
290 RETURN
300 PLOT OVER 1;40+x/5,y/5+80
310 PRINT #0; FLASH 1;" Comet at perihelion passage "
320 PAUSE 300: INPUT ""
330 FOR t=169 TO 1 STEP -1
340 LET yr=2023+INT (-t/4.5)
350 PRINT AT 0,26;yr
360 PLOT INK 6;40+x(t),-y(t)+80
370 PAUSE 10: NEXT t: STOP
390 CIRCLE 40,80,8
400 CIRCLE 40,80,40
410 FOR n=1 TO 3: READ a,b: PLOT 40+a,20: DRAW 0,120,b: NEXT n
420 PRINT AT 13,1;"Earth"
430 PRINT AT 18,1;"Jupiter Saturn Uranus Neptune": RETURN
440 DATA 40,2,105,1.1,160,.9
```

Pluto's Orbit

On 21st January 1979, Pluto relinquished to Neptune the dubious honour of being the most remote planet from the Sun by crossing within Neptune's orbit. Pluto will regain the 'title' in March 1999 by once more crossing Neptune's orbit — this time in an outward direction.

A collision between the two planets is unlikely as Pluto's orbit is inclined 17° to the general plane of the planets including Neptune. During this twenty-year period, Pluto will pass up to 10 AU (10 astronomical units = 10 x Earth to Sun distance) above Neptune's orbit — a gap almost big enough to contain the complete orbit of the giant planet Jupiter. The planets Pluto and Neptune are themselves well separated — Neptune last overtook Pluto in the 1890s as each moved ponderously along its orbital track in periods of 165 years and 248 years respectively.

Screen display

The following program PLOTs to scale the relative positions of Pluto and Neptune from the year 1880 to 2128 — one complete orbit for Pluto. Both a plan and a section are displayed of the overlapping orbits, with a tiny flashing CIRCLE in the centre representing the Earth's orbit (at the very centre of which is, of course, the Sun). The passing years are recorded in the top right of the screen and Pluto is PLOTted in a green pixel (a darker shade in monochrome) on a black background (PAPER 0: BORDER 0:) for clarity. The display PAUSEs briefly to PRINT six significant comments against specific dates relevant to the progress of the two planets.

Pluto's elliptical orbit

Neptune's orbit has the lowest eccentricity of all the planets, being a near-perfect circle. Because of this, its program requirements are minimal — just Line 340 both to calculate and PLOT its orbit, where variable f is used to increment the PLOT position.

In contrast, the bulk of the program from Line 110 is used to calculate Pluto's elliptical orbit, which has the highest eccentricity of the major planets with a value of 0.25. Pluto moves faster when closest to the Sun (by wider spacing of the PLOT positions) but, because this presentation is to scale and overlaps Neptune's orbit, it may not be obvious at this time.

Lines 300 and 310, both of which PLOT Pluto's orbital progress, appear almost identical except for the expression at the end of Line 310:

"…y/200 + 10 "

where /200 effectively compresses the y or vertical axis by a factor of 200 so that, instead of a second full ellipse being PLOTted, it is reduced to a near edge-on view of the orbit as shown in the lower part of the display.

Experiments with Pluto display

Once the program has been RUN a few times, using the screen COPY (Figure 5.12) to check that it works correctly, SAVE the program on tape. Now try changing some of the variables to see what effect it has. (These amendments will invariably corrupt an accurate presentation which is why the program should be SAVEd first.)

Figure 5.12
The orbits of Pluto and Neptune from two viewpoints.

The following variables can be tested for effect:

xx x coordinate (horizontal) of the Sun

yy y coordinate (vertical) of the Sun

f position and increment steps to PLOT Neptune (Lines 110 and 340)

h STEP value to main FOR/NEXT loop

t FOR/NEXT main loop starting at value 0 (3 o'clock start)

w relative eccentricity of Pluto's orbit

The value 13 (Lines 300 and 310) and the value 55 (Line 340) control the radius of each planet's orbit.

```
10 REM Pluto's Orbit
20 LET t=0: BORDER 0: PAPER 0: INK 7: CLS : GO SUB 400
30 PRINT INK 5;"Pluto's Orbit"
40 PRINT AT 11,0;"plan";TAB 16;"Sun";TAB 25; INK 5;"Pluto"
50 PRINT AT 15,14;"Neptune"
60 PRINT """edge-on"'" view"
70 LET xx=140: LET yy=92
80 PLOT xx-55,0: DRAW 110,20,.1: DRAW -110,-20,.1
90 CIRCLE FLASH 1;xx,yy,2
100 CIRCLE FLASH 1;xx,10,2
```

```
110 LET f=12.3: LET w=28.7
120 LET h=.8: LET g=1e6
130 LET x=g/1000: LET y=0
140 LET i=h/4: LET v=0
150 LET r=x: LET s=y
160 LET x=x+i*v: LET y=y+i*w
170 GO SUB 260
180 LET x=r: LET y=s: LET o=h/2
190 LET v=v+o*b: LET w=w+o*c
200 GO SUB 300
210 FOR t=0 TO 155 STEP h
220 LET x=x+h*v: LET y=y+h*w
230 GO SUB 260
240 LET v=v+h*b: LET w=w+h*c
250 GO SUB 300: NEXT t: STOP
260 LET e=x*x+y*y: LET d=SQR e
270 LET a=-g/e: LET b=a*x/d
280 LET c=a*y/d: RETURN
300 PLOT INK 4;x/13+xx,y/13+yy
310 PLOT INK 4;x/13+xx,y/200+10
320 LET tt=1880+INT (t*1.6): PRINT AT 0,20;"year=";tt
340 PLOT BRIGHT 1;xx+COS f*55,yy+SIN f*55: LET f=f+.05
360 IF tt=1888 OR tt=1929 OR tt=1979 OR tt=1999 OR tt=2040 OR tt=2127 THEN GO SUB
500
370 RETURN
410 LET b$="1889 - Neptune 'overtakes' Pluto"
420 LET c$="1930 - Pluto discovered"
430 LET d$="Pluto inside Neptune's orbit Jan 1979"
440 LET e$="Pluto outside Neptune's orbit Mar 1999"
450 LET f$="Neptune completes orbit: 164 yrs"
460 LET g$="Pluto completes orbit: 248 yrs"
470 DIM a$(40): RETURN
500 PRINT BRIGHT 1;AT 1,0;(b$ AND tt=1888)+(c$ AND tt=1929)+
(d$ AND tt=1979)+(e$ AND tt=1999)+(f$ AND tt=2040)+
(g$ AND tt=2127)
510 PAUSE 250: PRINT AT 1,0;a$: RETURN
```

Solar Apex

Even as you sit quietly at your Spectrum, you and, I hope, the whole room are moving rapidly across the universe. Nor are you moving constantly in a given direction but in a whole series of loops and whirling spirals. This series would read:

1. Rotation about the Earth's axis.
2. Earth's rotation about Earth/Moon axis.
3. Earth's rotation about the Sun.
4. Sun's rotation about the galaxy.
5. Galaxy's rotation about the Local Group of Galaxies (LG of G).
6. LG of G rotation about the Virgo Super Cluster of Galaxies (VSCG).
7. VSCG rotation about the universe.

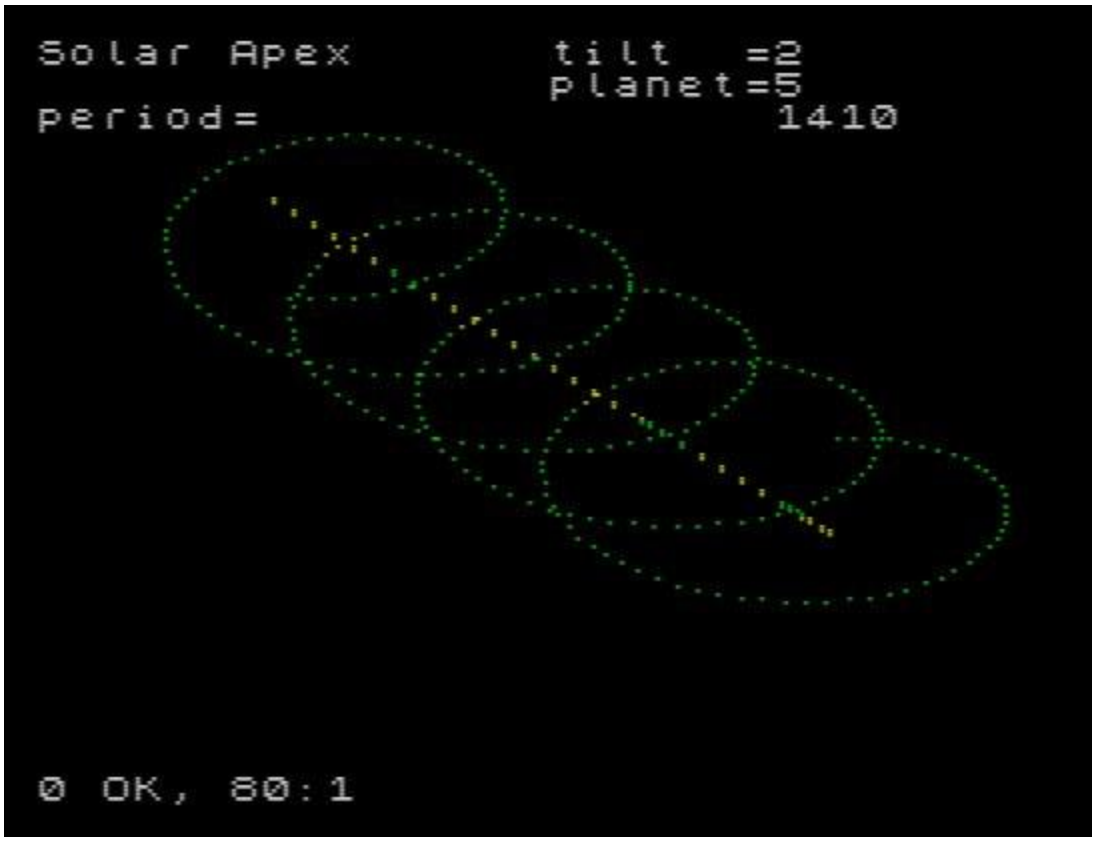
Most astronomers and cosmologists theorise in this way, but categories 5 and 6 are pure speculation and category 7 improbable under currently accepted theories of the universe's creation which conform to the idea of a Big Bang, where all groups of galaxies are still moving apart from the initial explosion. Of course if the universe is 'closed' and the galaxies finally come to rest they will probably reverse their motion back to the point of the Big Bang – in which case the galaxies are in orbit about this point even if their motion is in a straight line there and back (a maximum of one orbit only). A case of a shell falling back into the muzzle that fired it!

Down to Earth

The following program is a little less rarified and combines categories 3 and 4 to show the corkscrew motion of a planet towards a point in the sky, near the star Vega, called the Solar Apex. This is the direction in space in which the Sun is moving at a speed of 275 km/sec as it orbits our galaxy. The Sun is the straight line trace across the screen and the helical trace that of the planet, the orbit of which can be tilted at an angle for effect.

The various helices as PLOTted have a modest three-dimensional quality to them. Figure 5.13 is a typical example.

The two INPUT conditions control the orbit tilt, 1 for near edge-on, 10 for plan view whilst the planet INPUT alters the orbit radius. The period displayed is purely arbitrary and is controlled by the orbit radius merely to indicate that the helix will be finer and will occur in a shorter period of time on a smaller orbit.




```
5 BORDER 0: PAPER 0: INK 9
10 CLS : PRINT "Solar Apex",
15 PRINT "tilt =";
20 INPUT "1 to 10",z: PRINT z
25 PRINT ,"planet=";
30 INPUT "1 to 5",d: PRINT d,"period=";: LET d=d*10: LET x=100
40 FOR f=0 TO PI*9 STEP .1
45 PRINT AT 2,23;INT (d*f)
50 LET a=x-f*5+x: LET b=f*3+d
60 PLOT INK 6; OVER 1;a,b
70 PLOT INK 4;a+SIN f*d,b+COS f*d/z
80 NEXT f
```

****NB the keen-eyed among you will notice that the 'Period' label in the image is located in the wrong place. This is because the code for this in Line 30 should read: …**PRINT d',"period=";**… not as shown in the listing. This is my mistake not the author's.*

Chapter 6 – The Planets (Solar System Trek)

From antiquity, man recognised five planets or wanderers in the sky — Mercury, Venus, Mars, Jupiter and Saturn. Each was endowed with some feature in human nature which was admired or feared — such as naming the red planet, Mars, after the god of war. On a more practical level, men probably suspected that these were also worlds like their own, but too far away to see clearly. In 1609 Galileo, using his telescope (the first time this instrument had been used in astronomy), confirmed that this was so. Today colour TV cameras sit on the surface of Mars directly controlled by a computer joystick held 100 million miles away on planet Earth. Such is the pace of science.

Although we now know that the planetary environments are more hostile to the human frame than our forebears could have ever imagined, the planets continue to be fascinating objects in the Sun's family.

The programs that follow combine a variety of interests from simulating a scene on the Martian surface to computing the precise positions in our sky where the planets can be found; from using the Spectrum in Computer Aided Design (CAD) to judging the planetary images seen through a small telescope.

Solar System Trek

Fancy a trip to other worlds via your computer? This program lets you do just that and enables you to 'view' the solar system as seen from the skies of any planet (including Earth), on any date. You can even beat the mythical Icarus by viewing the planets from the surface of the Sun or perhaps from Jupiter during a spacecraft's fly-by.

The program contains all the necessary data to compute the various planetary positions ('ecliptic longitude'), the constellation in which each planet appears and the angular separation from the Sun ('solar elongation'). This is displayed in both table and graphic form — the latter as a 360° panoramic strip of sky centred on the Sun.

The computation and display take only a few seconds and are deliberately slowed down to make the information easier to assimilate. Good use is made of the Spectrum's colour and graphics and an option to COPY the screen via the ZX printer is included.

The display — Mercury to Neptune

The initial display lists the planets and DRAWs the orbits to two scales — one for the Earth-like 'rock planets', Mercury to Mars, and one for the remote 'giant gas planets', Jupiter to Neptune. Despite the program's simplicity, it is sufficiently accurate for you to identify the planets as seen from your back garden, assuming your 'viewpoint' is Earth and you have chosen a day with a clear evening. A star-atlas like Norton will be useful in finding the constellations.

The exceptions to this are the remote planets Uranus, Neptune and Pluto which are all too faint to be seen without a telescope and, even then, are indistinguishable from stars. Pluto is excluded from the program because its orbit is highly elliptical and inclined 17° to the general plane of the planets called the ecliptic. Circular orbits of zero inclination are therefore assumed — Mercury and Mars prove to be the least accurate but are only so over long periods of time.

The result from a program of this type is called an 'ephemeris' and it may be of interest to discuss the principles behind such work.

A plan of the solar system could be likened to a giant clock with eight hands of varying length — the outer tip of each hand representing a major planet. Each hand will sweep-out approximately the same area (shown shaded in **Figures 5.4 and 5.5 in the previous chapter**) in the same time interval. Thus the further a planet is from the Sun, the slower it moves and the longer it takes to complete an orbit.

If you know the position of the planets on an epoch or reference date, you only need to wind the hands backwards or forwards to locate the planets on any other date — past, present or future. If your viewpoint is the Sun, each planet will appear projected on to the background constellations, ie signs of the Zodiac, equal to the planet's heliocentric (suncentred) longitude. If your viewpoint is a planet, then the computer performs the necessary triangulation to deduce the revised positions. Use the sample screen displays in Figure 6.1 to check your results.

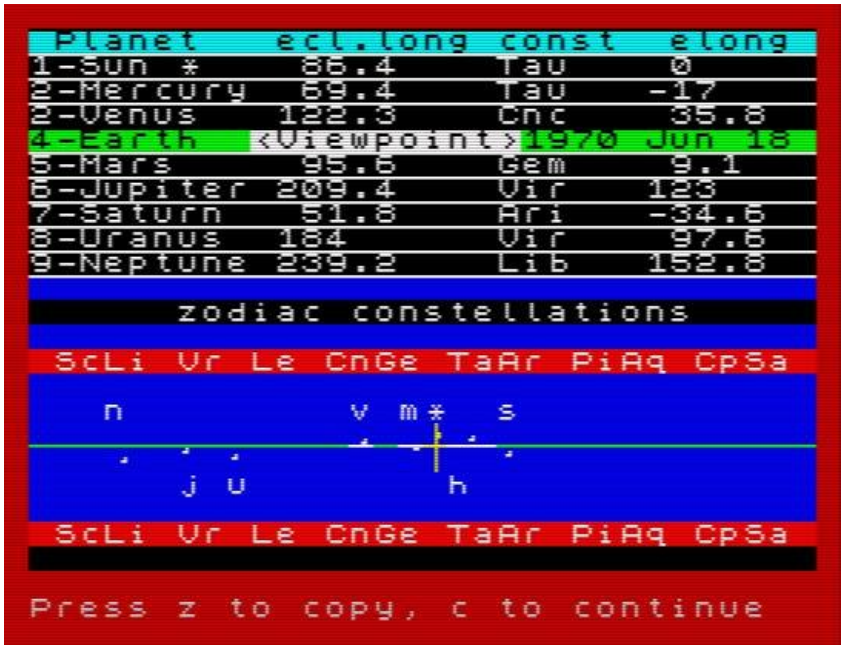
The program

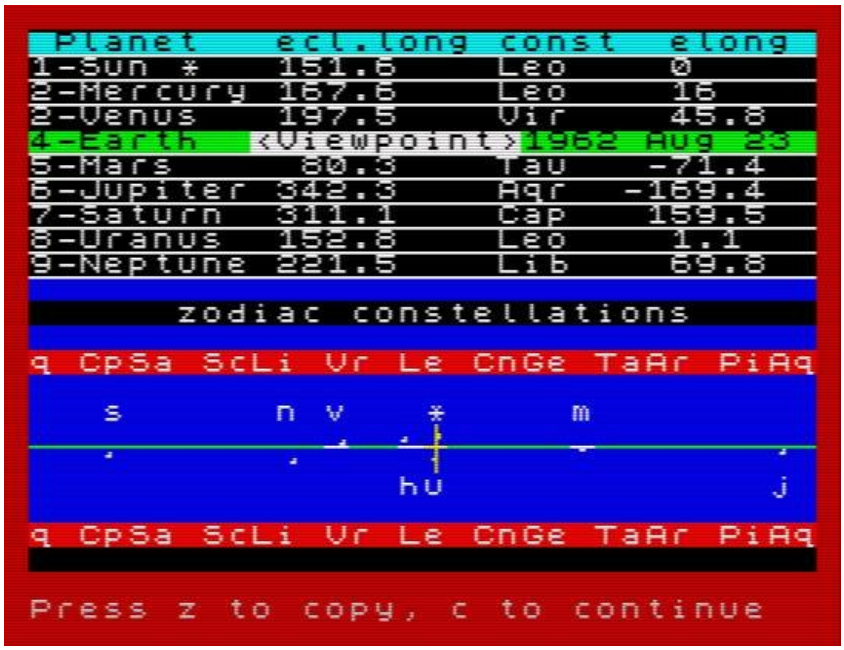
The REM statements show the general structure of the program with the DATA held from Line 1000. This program was originally designed for a ZX81 and I still have a liking for slicing string arrays for data! Be sure to double-check that these arrays are correctly entered — the smallest error will produce wrong results.

In the graphic displays, a '*' symbolises the Sun and 'h' (for Hermes — the classical Greek name) stands for Mercury, to avoid confusion with 'm' for Mars. The ecliptic longitude (eel. long) gives the planet's angular distance from the First Point of Aries, ie 0° (measured eastwards from 0° to 360°) and the solar elongation (elong) gives the angular distance from the Sun, ie 0°. A minus figure indicates that the planet is to the right of the Sun.

Figure 6.1

Check your program against these results.





Lines 420 and 440 separate the planets into two groups — those nearer to the Sun (inner planets) and those further from the Sun (outer planets) from the chosen viewpoint — and computes their positions accordingly. Under test it will be noted that, as seen from Earth, the inner planets Mercury and Venus never stray far from the Sun whilst all the outer planets can be found anywhere along the ecliptic. Conversely, from Neptune all the planets become inner planets with Mercury to Mars never more than a fraction of a degree from the Sun — virtually undetectable to a Neptunian!

```
10 REM Solar System Trek
20 BRIGHT 1: GO SUB 1000
```

```
40 LET l=1: CLS : BORDER RND*3
50 PRINT PAPER 5;"Solar System Trek ";CHR$ 127;" "
60 PRINT AT 14,10;"mve m j su n";AT 15,9;"*";AT 15,21;"*"
70 PRINT AT 19,0; PAPER 4;" rock planets "; PAPER 5;" giant gas planets"
80 LET a=0: LET ax=23
90 FOR n=1 TO 9: BEEP .1,9: IF n=6 THEN PAUSE 50: GO SUB 800: PAUSE 50: LET a=96:
LET ax=20/17
100 CIRCLE 75+a,51,a(n)*ax
110 PRINT AT n,0; PAPER 6-(2 AND n>1)+(1 AND n>5);y$(n): NEXT n
120 GO SUB 790
140 INPUT "Enter planet no",k: IF k9 THEN GO TO 140
150 BORDER k/2: LET j=k
160 PRINT PAPER 1;AT 10,0;b$
170 PRINT AT 11,0; PAPER 0; INK 7;" zodiac constellations "
180 PLOT 0,40: DRAW INK 4;255,0
200 INPUT "Date (yyyy,mm,dd)";TAB 6;y;TAB 11;m;TAB 14;d: IF y12 OR d>31 THEN GO TO
200
210 LET k$=o$(m*3-2 TO m*3)
220 PRINT AT k,9; FLASH 1;""; FLASH 0; PAPER 6-(2 AND k>1)+(1 AND k>5); INK 9;y;"
"+k$;" ";d;" "+(" " AND d2 THEN LET b=(m+1)*30.6-62-l: GO TO 280
270 LET b=(m-1)*(63-l)/2
280 LET dy=INT (b+d)
290 LET ed=INT ((y-ep)*u+dy+.5)
310 LET pp=c*(ed/t(j))+l(j)
320 LET qe=(pp/e-INT (pp/e))*e
340 FOR n=1 TO 9: IF n=9 AND n=j THEN GO TO 710
350 IF n=j THEN NEXT n
360 LET p=c*(ed/t(n))+l(n)
370 LET q=(p/e-INT (p/e))*e
380 IF j=1 THEN GO TO 440
400 IF a(n)<a> a(j) THEN LET el=q+r*ATN (SIN ((q-qe)/r))/(a(n)-COS ((q-qe)/r))
430 GO TO 450
440 LET el=q
450 IF ele THEN LET el=el-e
470 IF el>e OR el180 THEN LET b=b-e
540 IF b=e THEN LET el=el-e
560 LET v=1+INT (el/30)
570 PRINT AT n,0;y$(n);
580 PRINT TAB 10;(" " AND el<9);(" " AND el-100);(" " AND b>=0 AND b=10 AND
```

```
b<100);b
610 GO TO (n=1)*620+(n1)*650
620 LET w=30-sun/12: IF w>=0 THEN LET w=w+1
630 LET r$=m$(w TO )+m$( TO w)
640 PRINT INK 7; PAPER 2;AT 13,0;r$;AT 20,0;r$
650 LET z=0: LET nn=n/2
660 IF nn=INT nn THEN LET z=3
670 PRINT INK 7; PAPER 1;AT 15+z,b/12-16;z$(n)
690 INK 9: PLOT INT (132-b/1.5),44-n: DRAW 1,1: DRAW 0,-1
700 BEEP .02,n*3: NEXT n
720 IF j=1 THEN PLOT 130,38: DRAW INK 6;4,4: GO TO 740
730 PLOT 132,32: DRAW INK 6;0,15
740 GO SUB 790
750 PRINT #0;"Press z to copy, c to continue": PAUSE 0
760 IF INKEY$="z" THEN COPY : INPUT "": GO TO 750
770 GO TO 40
790 FOR n=175 TO 90 STEP -8: PLOT 0,n: DRAW 255,0: NEXT n: RETURN
800 CIRCLE 171,51,2: PLOT 171,53: DRAW -90,33: PLOT 171,49: DRAW -90,-33: RETURN
1010 DIM a(9): DIM l(9): DIM t(9): DIM y$(9,9): DIM b$(32*10)
1020 LET u=365.2654
1030 LET ep=1975: LET e=360
1040 LET r=180/PI: LET rr=e/PI
1050 LET c=e/u
1060 LET f=1e3: LET g=1e4
1070 LET o$="JanFebMarAprMayJunJulAugSepOctNovDec"
1080 LET z$="*hvemjsun"
1090 LET m$=" Le CnGe TaAr PiAq CpSa ScLi Vr"
1100 LET l$="000000320663310975099534249629355214104173205783249915"
1110 LET t$=".00001.24085.615211.00001.880911.86229.45884.012164.79"
1120 LET a$="000001003871007233010000015237052028095388191818300579"
1130 LET p$="1-Sun * 2-Mercury2-Venus 4-Earth 5-Mars 6-Jupiter7-Saturn 8-Uranus 9-
Neptune"
1140 LET c$="Psc Ari Tau Gem Cnc Leo Vir Lib Sco Sgr Cap Aqr "
1150 FOR n=1 TO 9: LET x=n*6
1160 LET a(n)=VAL a$(x-5 TO x)/g
1170 LET l(n)=VAL l$(x-5 TO x)/f
1180 LET t(n)=VAL t$(x-5 TO x)
1190 LET y$(n)=p$(n*9-8 TO n*9)
1200 NEXT n: RETURN
```

Planetary Ephemeris

There are degrees of accuracy. If you try to key in a more accurate program than the basic ephemeris program below, the length can increase alarmingly. A ten-fold increase in accuracy, say from 10′ (minutes) of arc error to 1′ of arc error may increase the LLISTing size three-fold. Both degrees of accuracy would place a planet in the same telescopic field of an amateur instrument magnifying, say, x 50, if it was pointed at the correct sky coordinates. It would be obvious that the object was a planet as the disc could be seen (with the possible exceptions of Uranus and Neptune — where the apparent star would be seen to move, over a couple of evenings, against the true stellar background, and be revealed as a planet.

There is a further and more pertinent point to consider. In the case of a book or magazine LLISTing which has to be keyed in, you could rapidly reach the situation where it was virtually impossible to debug keying in errors (particularly with complex formulae). This is a major reason for the relative briefness of the LLISTings in this book.

Occultations

Probably the ultimate in ephemeris computations is occultation work (the Moon or a planet passing in front of a star). The (usually professional) astronomer may need to compute the positions of each body with such accuracy that he would also have to INPUT the precise point on the Earth's surface within a metre or so of where the observation is to be made. Obviously such a program is only suitable for the massive memory of a mainframe or minicomputer rather than a micro like the Spectrum. Perfection is achievable at a price — both of hardware and mathematical expertise.

Having said this, some Spectrum user plus Microdrive will no doubt disprove it! Good luck to him or her. Dedication and single-mindedness could win the day.

Orbital elements

Now to cases and the program which follows. What is its accuracy and where are its shortcomings? First, a comparison with the Solar System Trek program, which would be regarded as relatively inaccurate — merely within a degree or so of the correct positions because only three parameters are used for computation. These are:

1. Period in years.
2. Radius of orbit in AU (astronomical units).
3. Position on orbit at epoch date (1975.0).

Thus, as previously mentioned, only circular orbits of zero inclination to each other are assumed, which is not precisely the case which exists. For greater precision three additional orbital elements must be considered:

1. Eccentricity of orbit.
2. Inclination to the ecliptic (Earth's orbital plane).
3. Location of ascending node (point where an inclined orbit crosses the ecliptic in a northerly direction).

For this particular program the epoch date has been upgraded to 1980.0 for item 3, and items 4, 5 and 6 are included.

For eccentricity, this program only solves Kepler's Equation to the first term, for elliptical orbits to an eccentricity of about 0.1. Mars, Mercury and Pluto, in ascending order, have larger values than this, and so their computed positions are more inaccurate than those of other planets.

Finding the planets

If the intention of this program is ultimately to find the planets in the sky, then this presents few problems except for Pluto. In reality Pluto is so faint in stellar terms — about magnitude 14 — that few amateur astronomers have seen or photographed it. A telescope of about 30 cm aperture is needed for visual use and one of about 10 cm aperture when it is being used as a guided camera to record this most elusive object. Pluto is however retained in the program as a matter of general interest, despite a typical error of j° in its computed position. If a telescopic lens of about 60 cm focal length is used on to 35 mm film, Pluto will still be captured on the exposure (if perhaps at the edge of the frame) whilst aiming at the predicted location in the sky produced by the program. One additional reason for Pluto's large error is its current perihelion passage which effectively exaggerates the program's shortcomings by placing Pluto behind (to the west) of its true position.

Pertabations

There is an additional factor which has been ignored in the program for the sake of brevity, namely planetary pertabations — the gravitational influence of each planet with the others which pulls a planet from a true path about the Sun. To include the major (and massive) culprits Jupiter and Saturn would increase the LLISTing length at least two-fold, without reference to the remaining seven planets. Notwithstanding these omissions, the program is accurate to within a few minutes of RA (time) and Dec (arc), especially for planets other than Mercury and Pluto.

RUNning the program

Once the program has been keyed in and is RUNning, the results should be checked against the sample screen displays, Figures 6.2 and 6.3. These should match precisely for the dates shown. With regard to the RA and Dec noted in s (seconds of time) and ” (seconds of arc) respectively, these should be assumed as icing on the cake for, as I've already pointed out, the program accuracy does not approach this level. However, it does not reach the absurd situation of some ephemeris programs, which imply such levels of accuracy that you are required to INPUT the hour and minute of the day, in addition to the date, when the predicted errors exceed a day's movement of the planet concerned against the star background.

Figure 6.2

The planetary positions on the summer solstice, 1985.

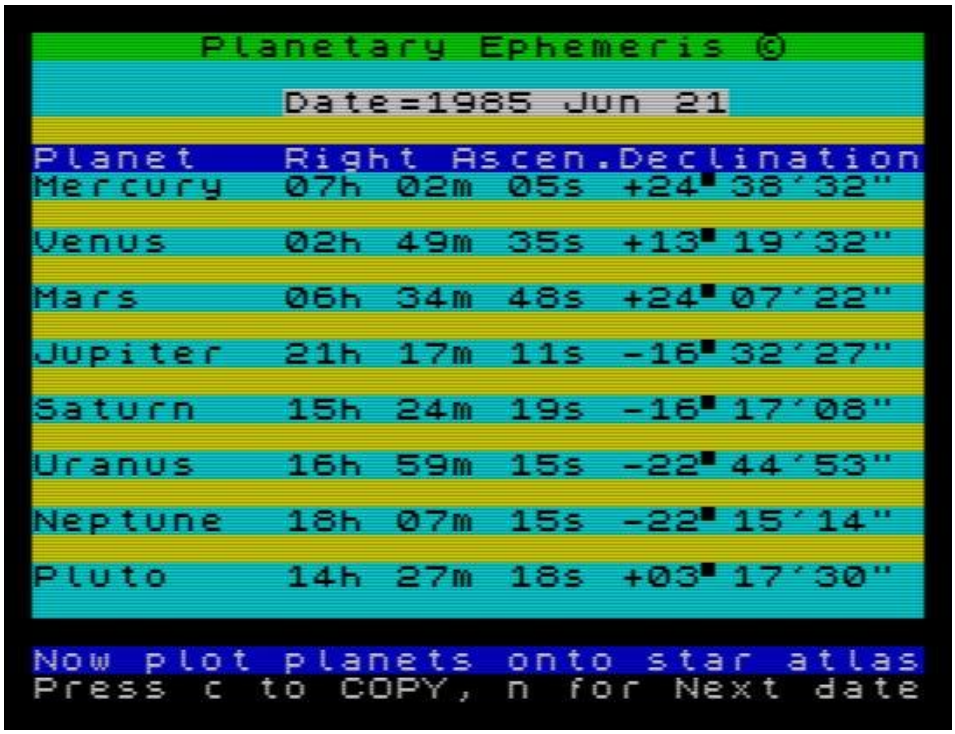


Figure 6.3

The planetary position on Christmas Day, 2014

Using star atlases

To be of practical use, particularly if you are unfamiliar with the night sky, the predicted positions should be plotted on a star atlas like Norton or Virion. The latter is printed to a larger scale and is more suitable if used with a small telescope.

To confuse matters a little, each atlas is prepared for a different epoch. This means that the reference grids of RA and Dec are very slightly different, as would be the apparent positions as plotted. In practice, the computed errors may exceed the difference of the reference grid between 1950.0 for Norton and 2000.0 for Tirion — the relevant epoch dates.

Checking on the past

Once you have INPUT a few current dates, and perhaps COPYed them to the ZX printer if you have this option, try entering some dates from the past. The program remains accurate to around one degree over several centuries. The principle errors are caused because we have not allowed for the pertabation effects. If the INPUT date is before the 15th October 1582 then add 10 days to the date to bring it into line with the current (Gregorian) calendar (**see Chapter 1**).

The following exercise relating to historical dates may be of interest. Test them against the program to find out where certain planets were on these dates using a star atlas for guidance.

INPUT these dates:

1610 January 10 : Galileo discovers moons of Jupiter (approximate date).
1610 February 8 : Undiscovered Uranus in Jupiter's star field.
1659 November 28 : Huyghens discovers Syrtis Major landmark on Mars at 7pm that evening. Mars's rotation accurately known to $^1/_{50}$ second aided by this observation.
1781 March 13 : Herschel discovers Uranus.
1846 September 23: Galle discovers Neptune in place predicted by Adams and Le Verrier from pertabations on Uranus.
1930 February 10 : Tombaugh discovers Pluto after 10-year search.

It is purely coincidental that the above events, (except for Neptune in Aquarius) occur in Taurus and (for Pluto) in adjacent Gemini. It is interesting to wonder if Galileo with his relatively poor-quality telescope could have discovered Uranus 170 years before Herschel, as some historians suggest. Many famous astronomers between the respective dates observed Uranus and recorded it as a star. Only Herschel with his superior reflector telescope immediately recognised a tiny disc that moved over the following evenings against the star background.

Thus it can be demonstrated that the program can be of practical value in finding the planets in the sky and is of some interest in a historical context. Historians have also suggested that the Star of Bethlehem could have been one of three possibilities:

- A new star or nova.
- A comet.

- A conjunction of bright planets.

Item c can be tackled by the program although it is not ideally suited, as it is a matter of trial and error to find a convenient conjunction (grouping together) of at least three bright planets for a given date. The four brightest planets are Venus, Mars, Jupiter and Saturn — Jupiter must be one of the candidates. Negative dates can be INPUT, like -11 which represents the year 12BC.

A better solution to this type of problem is to rewrite the program using FOR/NEXT loops and/or incremental days to PRINT to the screen when conjunctions occur, starting at a set date — say January 1 10BC until January 1 10 AD. Any pair of planets which form a suitable conjunction should be within about 1° of each other.

```
10 REM Planetary Ephemeris
30 GO SUB 2000
50 BORDER 0: PAPER 5: INK 9: CLS : PRINT PAPER 4;" Planetary Ephemeris ";CHR$ 127;"
"
60 PRINT " PAPER 6;b$
70 FOR f=1 TO 8: PRINT PAPER 6;b$: PRINT a$(f*7-6 TO f*7)
80 NEXT f
90 PRINT PAPER 1;AT 4,0;"Planet Right Ascen.Declination"
110 GO SUB 1000
130 LET we=98.83354
140 LET ee=.016718
150 LET z=cd*ed/1.00004: GO SUB 770: LET ne=z
160 LET z=ne+we-102.596403: GO SUB 770: LET me=z
170 LET z=ne+rr*.016718*SIN (me/ra)+we: GO SUB 770: LET le=z
180 LET ve=le-102.596403
190 LET re=(1-ee^2)/(1+ee*COS (ve/ra))
220 FOR n=1 TO 8
230 LET z=cd*ed/x(1,n): GO SUB 770: LET np=z
240 LET mp=np+x(2,n)-x(3,n)
250 LET z=np+rr*x(4,n)*SIN (mp/ra)+x(2,n): GO SUB 800: LET lp=z
260 LET vp=lp-x(3,n)
270 LET rp=x(5,n)*(1-x(4,n)*x(4,n))/(1+x(4,n)*COS (vp/ra))
280 LET psi=ra*ASN (SIN ((lp-x(7,n))/ra)*SIN (x(6,n)/ra))
290 LET yx=lp-x(7,n)
300 LET y=SIN (yx/ra)*COS (x(6,n)/ra)
310 LET x=COS (yx/ra)
320 LET ct=ra*ATN (y/x)
330 LET q=ct: GO SUB 820: LET ct=q
340 LET l1=ct+x(7,n)
```

```
350 LET r1=rp*COS (psi/ra)
360 IF n>2 THEN GO TO 430
380 LET ll=le-l1
390 LET a=ra*ATN ((r1*SIN (ll/ra))/(re-r1*COS (ll/ra)))
400 LET z=180+le+a: GO SUB 800: LET lam=z
410 GO TO 470
430 LET ll=l1-le
450 LET z=l1+ra*ATN ((re*SIN (ll/ra)/(r1-re*COS (ll/ra)))): GO SUB 800: LET lam=z
470 LET bet=ra*ATN (r1*TAN (psi/ra)*SIN ((lam-l1)/ra)/(re*SIN ((l1-le)/ra)))
490 LET la=lam/ra
500 LET ec=23.441884/ra
510 LET be=bet/ra
520 LET y=SIN la*COS ec-TAN be*SIN ec: LET x=COS la
530 LET rh=ra*ATN ((SIN la*COS ec-TAN be*SIN ec)/(COS la))
550 LET q=rh: GO SUB 820: LET rh=q
570 LET rx=rh/15: LET ry=INT rx
590 PAPER 5: PRINT AT n*2+3,9;
600 PRINT ("0" AND ry<10);ry;"h ";
610 LET mx=60*(rx-INT rx): LET my=INT mx: PRINT ("0" AND my<10);my;"m ";
620 LET sx=INT (60*(mx-INT mx)+.5): PRINT ("0" AND sx<10);sx;"s"
630 LET dc=ra*ASN (SIN be*COS ec+COS be*SIN ec*SIN la)
640 PRINT AT n*2+3,21;
650 IF dc=0 THEN PRINT "+";
670 LET dc=ABS dc
680 PRINT ("0" AND ABS dc<10);INT dc;CHR$ 130;
690 LET dm=60*(INT ABS dc-ABS dc)
700 PRINT ("0" AND ABS dm<10);INT ABS dm;CHR$ 39;
710 LET ds=60*(INT ABS dm-ABS dm)
720 PRINT ("0" AND ABS dsc)+(c AND z<0): RETURN
830 IF x0 THEN RETURN
850 LET q=q+c: RETURN
1010 INPUT "Date yyyy,mm,dd";TAB 5;y;TAB 10;m;TAB 13;d
1020 IF m12 OR d>31 THEN GO TO 1010
1030 LET m$="JanFebMarAprMayJunJulAugSepOctNovDec": LET m$=" "+m$(m*3-2 TO m*3)+" "
1040 LET d$=STR$ y+m$+STR$ d
1050 INPUT (d$+"-is date OK (y/n)? "); LINE q$
1060 IF q$="n" THEN GO TO 1000
1070 PRINT PAPER 7;AT 2,9;"Date=";d$
1110 LET j=INT (365.25*(y-(m<3)))+INT (30.6001*(m+1+12*(m<3)))+d-INT (y/100)+INT
```

```
(INT (y/100)/4)+1720996.5: LET ed=j-2444238.5
1130 RETURN
2010 REM period in years
2020 DATA .24085,.61521,1.88089,11.86224,29.45771,84.01247,
164.79558,250.9
2030 REM longitude at epoch
2040 DATA 231.2973,355.73352,126.30783,146.966365,165.322242,
228.070855,260.3578998,209.439
2050 REM longitude perihelion
2060 DATA 77.1442128,131.2895792,335.6908166,14.0095493,
92.6653974,172.7363288,47.8672148,222.972
2070 REM eccentricity
2080 DATA .2056306,.0067826,.0933865,.0484658,.0556155,
.0463232,.0090021,.25387
2090 REM major semi-axis
2100 DATA .3870986,.7233316,1.5236883,5.202561,9.554747,
19.21814,30.10957,39.78459
2110 REM inclination
2120 DATA 7.0043579,3.394435,1.8498011,1.3041819,2.4893741,
.7729895,1.7716017,17.137
2130 REM ascending node
2140 DATA 48.0941733,76.4997524,49.4032001,100.2520175,
113.4888341,73.8768642,131.5606494,109.941
2160 LET a$="MercuryVenus Mars JupiterSaturn Uranus NeptunePluto "
2180 DIM x(7,8): FOR n=1 TO 7
2190 FOR f=1 TO 8: READ x(n,f)
2200 NEXT f: NEXT n: DIM b$(32)
2220 LET c=360: LET cd=c/365.2422: LET ra=180/PI: LET rr=c/PI
2230 RETURN
```

The Moons of Mars

In Gulliver's Travels, published in 1735, satirical writer Jonathan Swift poked fun at contemporary astronomers by attributing superior instrumentation and discoveries to the mythical people of Laputa. At the same time Swift proved that he was no mean mathematician.

The Laputans discovered (amongst other things) ‘two lesser stars or satellites… revolving about Mars… at a distance of three times and five times the planet’s diameter.. .from the primary’s centre.. .in periods of 10 and 21 hours respectively….’

As the length of the Martian ‘day’ (24hr 37m) was first deduced by Cassini in 1666, Swift was probably fully aware that his fictional nearer moon to Mars would, as seen from the Martian surface, rise in the west and set in the east a few hours later. The second fictional moon would remain virtually fixed in the Martian skies, because the orbital period nearly matched the planet’s rotation period.

142 years later, in 1877, Professor Asaph Hall, using the 26-inch Naval Observatory refractor in Washington DC, turned Swift’s fiction into fact.

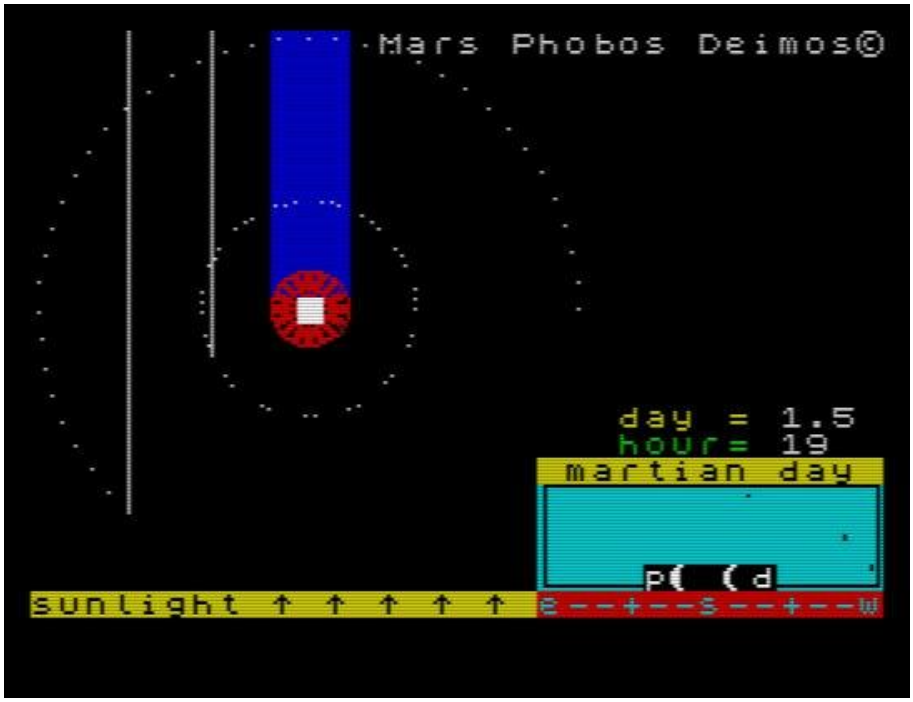
He discovered two moons of Mars, the inner moon indeed orbiting the planet faster than the planet’s axial rotation and the outer moon remaining above a given landscape for nearly three Martian days — passing through all its phases from new moon through full moon twice over. The currently recognised synodic orbital period (new moon to new moon) is 7hr 39m for Phobos and 30hr 21m for Deimos.

The screen display

The following program shows most of these factual features in an animated presentation, correct in both scale and relative motion of the three bodies. Incorporated into the display is a unique projection showing the skies as seen from the Red Planet’s surface, together with the changing phases of each moon as they orbit the planet. See Figure 6.4 for a typical display.

Figure 6.4

The positions of Phobos and Deimos are revealed by their shadows as they orbit Mars. Inset is the view from the Martian surface at local noon with the sun due south. Deimos is in the western sky. As in many of the programs in this book, they can only be fully appreciated in animation and colour.



The viewpoint is from space looking down on the north polar ice cap of Mars. Sunlight shines from the bottom of the screen with the shadow of the globe cast upwards into space. The planet is slowly rotating on its axis as indicated by a flickering line of longitude — the location of our observation post on the surface. Phobos and Deimos orbit the planet and their shadows in turn are cast up the screen into space.

At the bottom right of the screen is the mini-sky projection of the scene from the planet's surface. The view covers the southern horizon from east to west and the Sun and moons as they pass across the sky, all synchronised to the main display. Inset into the mini-sky projection is the current phase of each moon, marked 'p' for Phobos and'd' for Deimos. Immediately above is indicated the elapse time since the animation started, marked in days and hours.

Phobos and Deimos

The animation starts at a colourful sunset on the first day — Phobos and Deimos are due south and therefore centred in the mini-sky projection. Phobos moves rapidly to the left in overhauling the planet's rotation and soon sets on the eastern horizon. In contrast, Deimos moves very slowly westwards, remaining in the sky for over 30 Martian hours before setting, being overtaken by the Sun in the process. Phobos will arc across the skies from west to east many times in the days that follow — Deimos only putting in an appearance again at the end of the program RUN.

It is advisable to reRUN the program several times, preferably in colour, to glean the most from the various interacting displays. The following pointers may be of interest.

When an orbiting moon is at the top of the screen, it appears as a fully illuminated disc or full moon. When at the bottom of the screen, between the Sun and Mars, it is said to be a new moon and, because it is effectively

unilluminated as seen from Mars, it disappears briefly. If new moon occurs during the Martian daytime then the moon in question will pass briefly near the Sun, as shown on the mini-sky projection. The UDG CHR$ set from CHR$ 144 to CHR$ 159 is defined in the program to show the moons' phases in 16 steps.

The program RUN time is controlled by the FOR/NEXT n loop in Line 280 and RUNs for 5 days where PI* 10 equals 10 semicircular arcs of planetary rotation STEPped at half hour intervals by STEP PI/24. Test the program with different values instead of 10 and 24 in this line. A RUN longer than 5 days lacks synchronisation for Phobos.

It is also possible to change the size of Mars and the two orbits with the variable scale in Line 110. Currently it is set so as to contain Deimos's orbit on the Spectrum screen. The variables 'mars', 'ph' and 'de' are the diameter of Mars and the orbital radii of Phobos and Deimos respectively in kilometres, and should not be changed.

```
10 REM Moons of Mars
20 RESTORE : GO SUB 2000
40 DIM a$(3): DIM d$(13)
50 LET m$=" martian "
60 LET n$="Mars Phobos Deimos"
70 LET x=83: LET y=92: LET d=0: LET p=0: LET sky=0: LET sk=1
80 LET h=0
90 LET dd=4.7: LET pp=4.7
100 LET m1=4: LET m2=4
110 LET q=203: LET scale=290
120 LET mars=6790/2/scale
130 LET m=mars-1
140 LET ph=9350/scale
150 LET de=23487/scale
170 BORDER 0: PAPER 0: CLS
180 PRINT AT 12,9;n$;AT 15,22; INK 4;"hour=";AT 14,22; INK 6;"day ="
190 PRINT PAPER 6; INK 9;AT 16,19;m$+"day """"'"sunlight ^ ^ ^ ^ ^ "; PAPER 2; INK
5;"e-+-s-+-w"
200 LET pa=5: GO SUB 740
220 FOR n=0 TO 9: PRINT PAPER 1;AT n,9;a$: NEXT n
230 OVER 0: CIRCLE x,y,mars
240 FOR n=0 TO PI*2 STEP PI/24
250 PLOT x,y: DRAW INK 2;COS n*m,SIN n*m: NEXT n: PRINT AT 12,5;d$+d$; BRIGHT 1;
PAPER 7;AT 10,10;" "
260 PRINT AT 0,13;n$;CHR$ 127
280 FOR n=0 TO PI*10 STEP PI/24
290 IF h>24 THEN LET h=h-24
300 PRINT AT 15,28;h;" "
```

```
310 BEEP .01,40: LET h=h+.5
320 PRINT AT 14,28;sk-.5;" "
340 LET cn=COS n: LET cm=cn*m
350 LET sn=SIN n: LET sm=sn*m
360 IF sky/24=INT (sky/24) THEN LET pa=3: GO SUB 740: GO SUB 690
370 GO SUB 850
390 LET dc=COS d: LET ds=SIN d
400 LET pc=COS p: LET ps=SIN p
420 OVER 1: LET ddc=COS dd: LET dds=SIN dd: LET ppc=COS pp: LET pps=SIN pp: LET
sky=sky+1
440 LET dx=x+dc*de
450 LET dy=y+ds*de
460 LET px=x+pc*ph
470 LET py=y+ps*ph
490 FOR f=0 TO 1: PLOT x,y: DRAW INK 2;cm,sm: GO SUB 780
500 INK 9
520 PLOT dx,dy
530 DRAW 0,175-dy-(96 AND dyx-13 AND dx<x+13)
540 PLOT px,py
550 DRAW 0,175-py-(96 AND pyx-13 AND px15 THEN LET m1=m1-16
640 IF m2>15 THEN LET m2=0
650 OVER 0: NEXT n
660 PRINT #0; FLASH 1;" Press any key to run again ": PAUSE 0: GO TO 30
690 PRINT AT 16,19;
700 IF skINT sk THEN PRINT PAPER 6; INK 9;m$+"day ": LET pa=5
710 IF sk=INT sk THEN PRINT PAPER 5; INK 9;m$+"nght": LET pa=1
720 LET sk=sk+.5
740 FOR k=17 TO 20: PRINT PAPER pa;AT k,19;d$: NEXT k
750 INK 9: PLOT 154,9: DRAW 100,0: DRAW 0,30: DRAW -100,0: DRAW 0,-30
760 PRINT BRIGHT 1; PAPER 0; INK 7;AT 20,23;"p d"
770 RETURN
790 INK 9: IF sm<0 THEN PLOT q+cm*4.2,10+ABS sm*2.4: DRAW 0,1
810 IF dds<0 THEN PLOT q+ddc*50,12+ABS dds*20: DRAW 0,1
820 IF pps<0 THEN PLOT q+ppc*50,10+ABS pps*20: DRAW 0,1
830 INK 0: RETURN
850 PRINT PAPER 0; INK 7; BRIGHT 1;AT 20,24;CHR$ (144+ABS m1)+" "+CHR$ (144+m2):
RETURN
2010 DATA 0,0,0,12,2,1,12,6,3,12,6,7,12,14,15,12,30,31,28,62,63
2020 DATA 60,126,127,60,126,255,60,126,254,56,124,252,48,120,248
```

```
2030 DATA 48,112,240,48,96,224,48,96,192,48,64,128
2050 FOR n=0 TO 15: FOR f=0 TO 1
2060 READ p
2070 POKE USR CHR$ (144+n)+f,p
2080 POKE USR CHR$ (144+n)+7-f,p
2090 NEXT f
2100 READ c: FOR x=2 TO 5
2110 POKE USR CHR$ (144+n)+x,c
2120 NEXT x: NEXT n: RETURN
```

Jupiter's Satellites

In 1609, Galilei Galileo in Italy applied his newly-constructed telescope (invented by Han Lippershey in Holland the previous year) to the study of the heavens. His announcements astounded the civilised world but dismayed the Church.

One discovery in particular caused the gravest consternation with the authorities: 'the planet Jupiter had four attendant moons orbiting it…' — proof that not all heavenly bodies encircled the Earth and that the Earth (and especially the Vatican) was not the centre of the universe. At a time when science had reached a high pinnacle in Europe, Italy became isolated from the discussions for many decades thereafter. Today everyone can enjoy the thrill of discovering the moons of Jupiter with a modest pair of binoculars, and this program will help you do so.

Despite the modest length of this ephemeris-type program, it is accurate enough for you to identify and name the four moons for whatever date and hour you choose. The predictions prove virtually identical to those published in Sky & Telescope and the BAA Handbook, both authorities on this type of work.

Plotting the moons

The program has a good display and screen layout, best seen in colour. Once the program has been keyed in and RUN, the chosen date is requested of you for INPUT. You then have the option of computing and displaying the position of the moons for that date at two-hour intervals or for each day at midnight (0 hours universal time) for a period of twelve days. The layout (see Figures 6.5 and 6.6) is such that you can see the various moons 'swaying' back and forth across the planet Jupiter, fixed in the centre line of the screen display.

The bi-hourly motion of the moons is quite small but noticeable, particularly for the inner moons Io and Europa. The daily motion of the moons (with an INPUT of'd') is very marked in comparison and to help you identify each moon the 'configuration' as it is called, is displayed against each prediction as a series of numbers plus the letter 'J' for Jupiter. The numbers represent the order of the moons from Jupiter, 1 = Io, 2 = Europa, etc. Of course from

Earth, for which this program computes the apparent position of each moon, the order of the configuration may seem jumbled, 132J4, caused by viewing the Jovian system edge-on. A bird's-eye view would show all the moons following neat circles about Jupiter and maintaining their order of 1, 2, 3, 4 from the planet.

Finding Jupiter in the sky

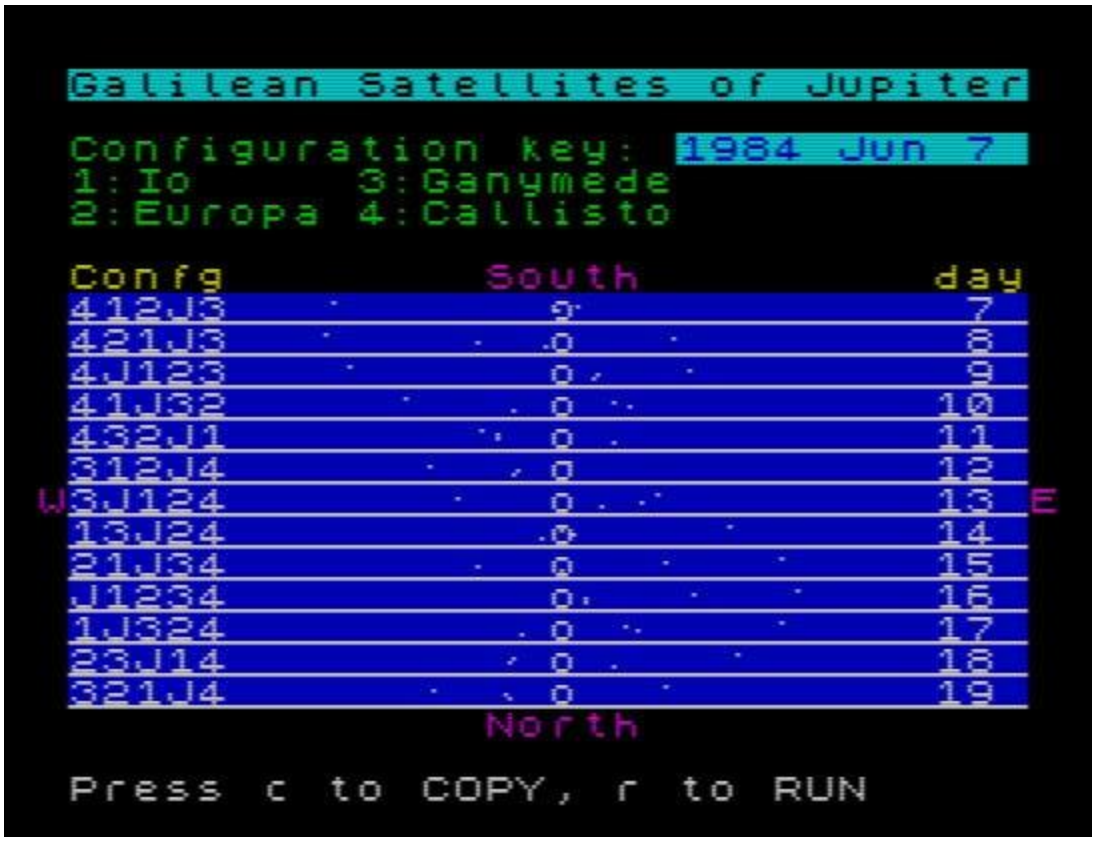
The planet Jupiter is currently visible low in the south-west skies from Britain and the rest of the northern hemisphere, after dusk and can be seen each summer and autumn night for some years to come low in the southern aspect. (The prospect for the southern hemisphere is even more favourable.) It is the brightest 'star' in that region of the sky — a powerful pair of binoculars or a small telescope will show the planet as a tiny disc and the moons in attendance as points of light shifting hourly and daily in time with your program. If the evening you choose to use this program is cloudy and you have a ZX printer why not make a COPY — the program contains this option — against INPUT'd' and save it for when skies clear.

Figure 6.5

The positions of Jupiter's moons for one selected day at 2-hour intervals

Figure 6.6

This plots the moons for a 12-day period. The daily motion of the moon is much more pronounced.




```
10 REM Jupiter's Satellites
20 BORDER 0: PAPER 0: INK 9: CLS
30 DIM x(5): DIM t$(5): DIM x$(30)
40 DEF FN z(i)=i-360*INT (i/360)
50 LET t$="1234J": LET s$=""
60 LET r1=PI/180
70 LET m$="JanFebMarAprMayJunJulAugSepOctNovDec"
90 PRINT PAPER 5;AT 1,1;"Galilean Satellites of Jupiter"
100 INPUT "Year: ";yr
110 PRINT '" Year = ";yr
120 LET d$=STR$ yr
130 INPUT "Month (1-12): ";mh: IF mh12 THEN GO TO 130
140 LET m$=m$(mh*3-2 TO mh*3)
150 PRINT " Month= ";m$
160 LET d$=d$+" "+m$
170 INPUT "Day: ";dy: IF dy31 THEN GO TO 170
180 PRINT " Day = ";dy
190 PRINT " Interval period=";
200 INPUT "Hours or days (h/d)? "; LINE a$
```

```
210 IF a$="h" THEN PRINT "2 hrs"
220 IF a$"h" THEN PRINT "daily"
230 PRINT #0; FLASH 1;" Date OK (y/n)?": PAUSE 0: IF INKEY$="n" THEN GO TO 10
240 LET d$=d$+" "+STR$ dy
260 CLS
270 PRINT PAPER 5;AT 1,1;"Galilean Satellites of Jupiter"
280 PRINT INK 4;AT 3,1;"Configuration key:"," 1:Io 3:Ganymede"," 2:Europa
4:Callisto"
290 PRINT PAPER 5; INK 1;AT 3,20;d$+(" " AND LEN d$<11)
300 INK 6: PRINT AT 7,1;"Confg"
310 PRINT AT 7,27;("UThr" AND a$="h")+(" day" AND a$"h"): INK 9
320 PRINT INK 3;AT 7,14;"South";AT 14,0;"W";AT 14,31;"E";AT 21,14;"North"
340 LET m=mh: LET y=yr
350 IF mh>=3 THEN GO TO 370
360 LET m=m+12: LET y=y-1
370 LET f=INT (y/100)-INT (y/400)
390 LET a=INT (365.25*(y+4712))-2415020
400 LET b=INT ((367*(m-1)+5)/12)
420 LET hr=0: FOR c=0 TO 12
430 LET d=dy+hr/24+a+b-f-.5
450 LET m=FN z(358.476+.9856003*d)
460 LET n=FN z(225.328+.0830853*d)
470 LET j=FN z(221.647+.9025179*d)
480 LET aa=1.92*SIN (m*r1)+.02*SIN (2*m*r1)
490 LET bb=5.537*SIN (n*r1)+.167*SIN (2*n*r1)
500 LET k=j+aa-bb
510 LET delta=SQR (28.07-10.406*COS (k*r1))
520 LET psi=ASN (SIN (k*r1)/delta)/r1
530 LET u1=FN z(84.5506+203.405862*(d-delta/173)+psi-bb)
540 LET u2=FN z(41.5015+101.291632*(d-delta/173)+psi-bb)
550 LET u3=FN z(109.977+50.2345169*(d-delta/173)+psi-bb)
560 LET u4=FN z(176.3586+21.4879802*(d-delta/173)+psi-bb)
570 LET x(1)=5.906*SIN (u1*r1)
580 LET x(2)=9.397*SIN (u2*r1)
590 LET x(3)=14.989*SIN (u3*r1)
600 LET x(4)=26.364*SIN (u4*r1)
610 LET x(5)=0
630 PAPER 1
640 PRINT AT c+8,1;x$;AT c+8,16; INK 6;"o"
```

```
650 PRINT AT c+8,28;
660 IF a$="h" THEN PRINT (" " AND hr<10);hr
670 IF a$"h" THEN PRINT (" " AND dymax THEN LET max=x(n): LET t=n
730 NEXT n
740 LET s$=s$+t$(t): LET x(t)=-50
750 NEXT o
760 PRINT PAPER 1;AT c+8,1;s$: LET s$=""
770 PLOT 8,104-c*8: DRAW 239,0
780 IF a$"h" THEN LET dy=dy+1
790 IF a$="h" THEN LET hr=hr+2
800 BEEP .1,30: NEXT c
820 PRINT #0;" Press c to COPY, r to RUN": PAUSE 0
830 IF INKEY$="c" THEN LPRINT : COPY : INPUT "": GO TO 820
840 RUN
```

The Rings of Saturn

Saturn is the most splendid of planets: it marked the boundary of the known solar system until 1781. It is a cold world — the Sun is no more than a brilliant star in its skies. But its remoteness has not stopped man from seeking a closer view, with the highly successful Voyager spacecraft of August 1981 returning images clearer than were ever possible from Earth-based telescopes (from a distance of about 900 million miles).

The beautiful ring planet has always impressed people, be they beginners or experts at the telescope. This program, and the two that follow are designed to satisfy different interests and demonstrate the graphic potential of the Spectrum.

The Rings of Saturn probably represents the most accurate computer simulation of Saturn ever attempted on a home micro. It is possibly even superior to many mainframe efforts, with the exception of NASA's Planetary Laboratories.

It features a full-screen solid image with all hidden lines deleted and the globe and ring system drawn accurately to scale. The user may tilt the planet and ring system at any angle up to 90°. If 0° is INPUT the planet is drawn as viewed directly over its equator with the rings shown edge-on. If 90° is INPUT a polar view is presented with the ring system completely encircling the planet. You may select a northern or southern aspect for the tilt. An INPUT of's' will show the underside of the rings and equator and visible pole for this aspect.

The sequence for drawing the planet is as follows:

1. Check image size; rescale if tilt >43°.
2. Draw globe correcting apparent oblateness for tilt.

3. Draw equator, correcting for tilt.
4. Draw semi-transparent ring system.
5. 'Trace' nearest edge of rings across globe according to N/S aspect.
6. Draw Cassini Division through ring system: clockwise for N aspect.
7. Delete Cassini Division if 'behind' planet.
8. Plot visible pole position correcting apparent oblateness for tilt.

Using the SCREEN$ command

The program contains an option to COPY the completed picture to the ZX printer or to SAVE the image on to tape with the SCREEN$ command. The latter is simplicity itself — it is only necessary to press 'p' (for picture) and start the recorder to SAVE the picture you have created. The angle of tilt is automatically SAVEd in the file name, eg sat -23.4. To recreate the picture, enter LOAD "sat – 23.4″SCREEN$ and the Spectrum will search and display the appropriate image off the tape.

Saturn and computer simulations

Unlike some uninformed computer presentations of Saturn, the globe is not circular (except for the polar view) and so the CIRCLE command cannot be used to draw the planet. This oval (or correctly termed oblate spheroid) shape is due to Saturn's rapid axial rotation in lOh 14m (Saturn's day) causing the equatorial 'bulge' and polar 'flattening'.

The Spectrum DRAWs the globe and ring system via the PLOT command using a rapid ellipse routine. In the case of the globe the reduction in oblateness from 10% to zero (full circle) is applied progressively through the change of tilt from 0° to 90°. Similarly the position of the equator and visible pole are correctly located according to the tilt of the planet and the oblateness as presented.

As seen from Earth, the appearance of the planet is limited to a maximum tilt of 26.73° (Saturn's axial tilt to its orbit about the Sun) in both north or south directions plus or minus 0.49°, depending on the relative positions of Earth and Saturn in their orbits at the time. Thus any tilt in excess of 28° will represent a viewpoint other than from Earth.

The program

The REM statements show the general structure of the program. In the case of DRAWing the globe and rings, it is only necessary to calculate the outline of one quadrant of the ellipse and to mirror this in the remaining three quadrants by DRAWing each quadrant sequentially. Such a routine ensures that the DRAWing is executed rapidly with the minimum of calculation to slow the program down. In the case of the routine to DRAW the Cassini

Division in the ring system, the program PLOTs one complete ellipse and you will note how relatively slow, though satisfying, this proves to be.

```
10 REM Rings of Saturn
20 LET sc=1: BORDER 0: PAPER 0: INK 6: CLS
30 PRINT "Saturn ";CHR$ 127;
40 INPUT "Tilt (0\' to 90\' )",z
50 IF z>43 THEN LET sc=.66
60 INPUT "N or S tilt (n/s): "; LINE a$
70 LET ob=1.1*z/100
80 LET oe=.89+.11*ob
90 IF a$="s" THEN LET z=-z
100 PRINT TAB 21;"Tilt=";z;CHR$ 130
110 LET e=1/SIN ((.1+z)/180*PI)
115 LET p=COS ((.1+z)/180*PI): IF a$="s" THEN LET p=-p
120 LET x=255/2: LET y=88
130 LET r=54*sc: LET h=126*sc
140 GO SUB 360
160 FOR f=0 TO 1.58 STEP .01
170 LET c=INT (SIN f*h)
180 LET d=INT (COS f*h/e)
190 PLOT x+c,y+d
200 DRAW -c/3,-d/3
210 PLOT x-c,y+d
220 DRAW c/3,-d/3
230 PLOT OVER 1;x-c,y-d
240 DRAW c/3,d/3
250 DRAW OVER 1;0,1
260 PLOT OVER 1;x+c,y-d
270 DRAW -c/3,d/3
280 DRAW OVER 1;0,1
290 NEXT f: GO SUB 460
310 PRINT #0;"Press x to COPY, c to CONTINUE"
320 PRINT #1;TAB 6;"s to SAVE ""sat ";STR$ z;""""": PAUSE 0
330 IF INKEY$="x" THEN COPY : INPUT "": GO TO 310
340 IF INKEY$="s" THEN INPUT "": SAVE "sat "+STR$ zSCREEN$ : INPUT "": GO TO 310
350 RUN
370 FOR f=0 TO 1.420 STEP .019
380 LET a=INT (SIN f*r)
```

```
390 LET b=INT (COS f*r*oe)
400 PLOT x+a,y+b: DRAW 0,-b*2
410 PLOT x-a,y+b: DRAW 0,-b*2
420 NEXT f
440 FOR f=PI*.5 TO PI*1.5 STEP .1: PLOT OVER 1;x+SIN f*r,y+COS f*r/e: NEXT f:
RETURN
460 LET cd=2.1
470 FOR f=0 TO PI*2 STEP .03: PLOT OVER 1;x+SIN f*r*cd,y+COS f*r*cd/e: DRAW OVER
1;2,0: NEXT f
485 IF ABS z>26 THEN GO TO 570
490 FOR f=0 TO 1.42 STEP .01
500 LET a=INT (SIN f*r)
510 LET b=INT (COS f*r*oe)
520 IF a$="s" THEN LET b=-b
530 PLOT x+a,y: DRAW 0,b
540 PLOT x-a,y: DRAW 0,b
550 NEXT f
570 PLOT OVER 1;x,y+r*p*oe
580 RETURN
```

Saturn's Rings

This program is similar to the previous one but stripped to the minimum to produce an outline of the planet and its ring system. It indicates how a program can be brief and still prove satisfactory. Figures 6.7 and 6.8 are COPYs of the screen.

Figure 6.7
Saturn with rings wide open.

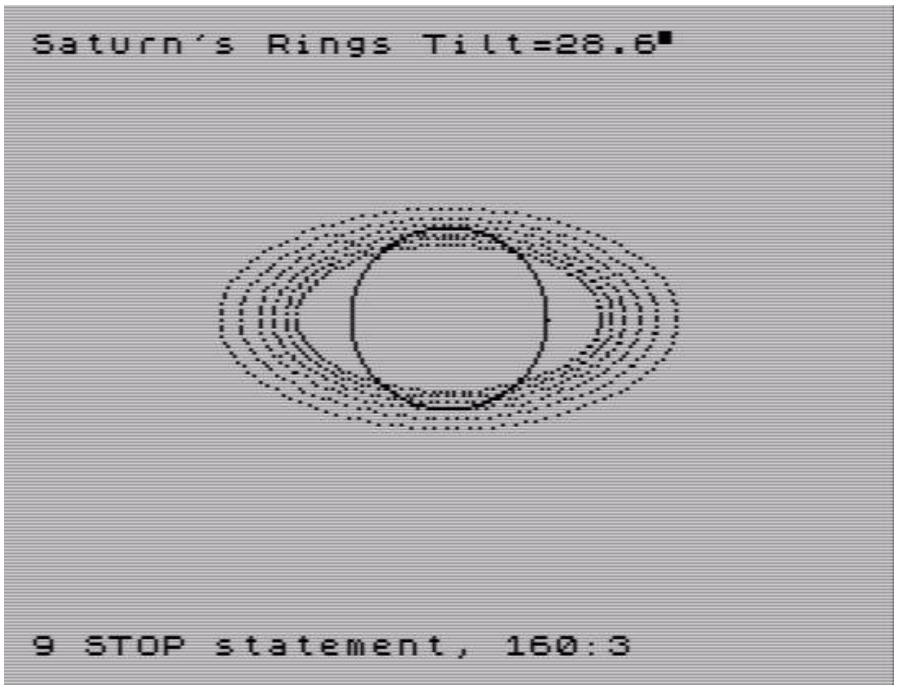


Figure 6.8
Saturn with rings edge on.

The program

With this program you can simulate the appearance of Saturn and its ring system tilted at any chosen angle from 0 to 90 degrees. If 0 is ENTERed, the planet is drawn as viewed directly over its equator with the rings shown edge-on. If 90 is ENTERed, a polar view is presented with the ring system completely encircling the planet. Any

intermediate angle will tilt the planet. As seen from Earth the appearance is limited from 0 to 26.73 degrees of tilt in both north and south directions plus or minus 0.49° depending on the relative positions of Earth and Saturn in their orbits at the time.

For simplicity, no attempt has been made to delete the ring system as it passes behind the planet and it is left to you to decide which portion of the rings is nearest to you. The Spectrum will influence your initial decision by drawing across the lower section of the globe first.

The Spectrum draws the globe and ring system with the PLOT command, using a modified Ellipse program (see Chapter 9). In the case of the globe the reduction in oblateness from 10% to zero (full circle) is applied progressively through the change of tilt from 0 to 90 degrees.

```
10 PRINT "Saturn's Rings ";
30 INPUT "Tilt (0"; CHR$ 130;"-90"; CHR$ 130;")",z
40 PRINT "Tilt=";z;CHR$ 130
50 LET ob=1.1*z/100
60 LET z=1/SIN ((.1+z)/180*PI)
70 LET x=128: LET y=88
80 LET r=30: LET h=70
90 GO SUB 180
110 FOR n=1 TO 1.6 STEP .1
120 FOR f=0 TO PI*2 STEP .05
130 LET sx=INT (SIN f*h)
140 LET cy=INT (COS f*h/z)
150 PLOT x+sx/n,y+cy/n
160 NEXT f: NEXT n: STOP
180 FOR f=0 TO PI*2 STEP PI/100
190 PLOT INT (x+SIN f*r),INT (y+COS f*r*(.89+.11*ob))
200 NEXT f: RETURN
```

Saturn Draw

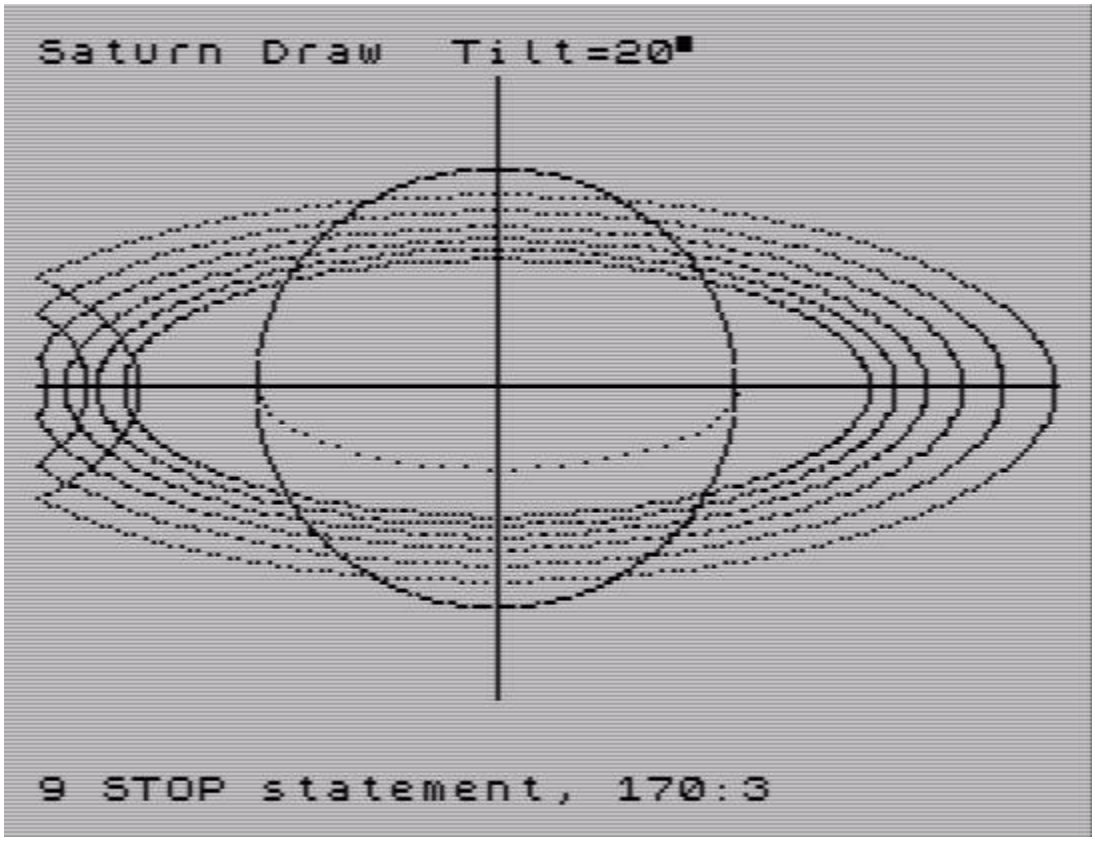
For the dedicated Saturn observer it is usual to prepare an outline of the planet and ring system for use at the telescope. The most difficult aspect in sketching the planet is getting the ring geometry correct. Much precious telescope time can be wasted in attempting to draw these subtle ellipses — a test for the most skilled draughtsman. In case those unfamiliar with planetary observation suspect that the following program will obviate the need to go to the telescope at all, this is not so. There are many subtle shadows and shadings and divisions in the rings to be recorded — the program is merely an aid, and great fun even if you are not a telescope enthusiast.

This short program, based on the Saturn Rings routine, is designed for use with the ZX printer by drawing standard outlines measuring 100 mm across the ring system major axis for subsequent tracing. Because the ZX printer cannot cope with this full width, the left hand edge of the ring system has been folded.

The planet can be tilted up to 27.22° and the program accepts 0° to 28° for INPUT. Negative values cannot be entered and in any case are unnecessary as the globe is transparent. Correction for globe oblateness according to the tilt has been carefully applied and the ring systems represent the outer edge of ring A (largest ellipse) and inner edge of ring B (smallest ellipse) respectively. Figure 6.9 is a typical COPY from the program.

Figure 6.9

Copy from the ZX printer as a standard outline for sketching Saturn through a telescope.



Computer Aided Design

If you wish to have a permanent record of the appearance of Saturn through every possible phase over a 29½-year orbital period, covering the axial tilt from 0° to 28°, make the following minor modifications to the program so that the Spectrum with printer does it automatically. You will need about 2 m (6½ ft) of ZX printer paper for the 29½ COPYs. Amend Line 170 to read:

170 NEXT f: NEXT n: COPY: CLS: LET z = z+ 1: GOTO 40

Now press

RUN ENTER 0 ENTER

and your Spectrum will make a fine and effortless job of Computer Aided Design (CAD). The computer is best left to its own devices as it takes about 5 minutes to PLOT each image before it is COPYed. The PLOT time can be reduced to about 90 seconds per image if only the outer and inner rings are drawn, with wider spacing between the PLOT positions. For this, the STEPs in the two FOR/NEXT loops in Lines 120 and 130 need modification as follows:

120… STEP .5 130… STEP .04

If you are unhappy with the left portion of the ring system being folded, then the whole image can be rescaled so that it is contained on the screen, by changing the variables in lines 80 and 90:

x = 255/2
r = 60*.9
h = 140*.9

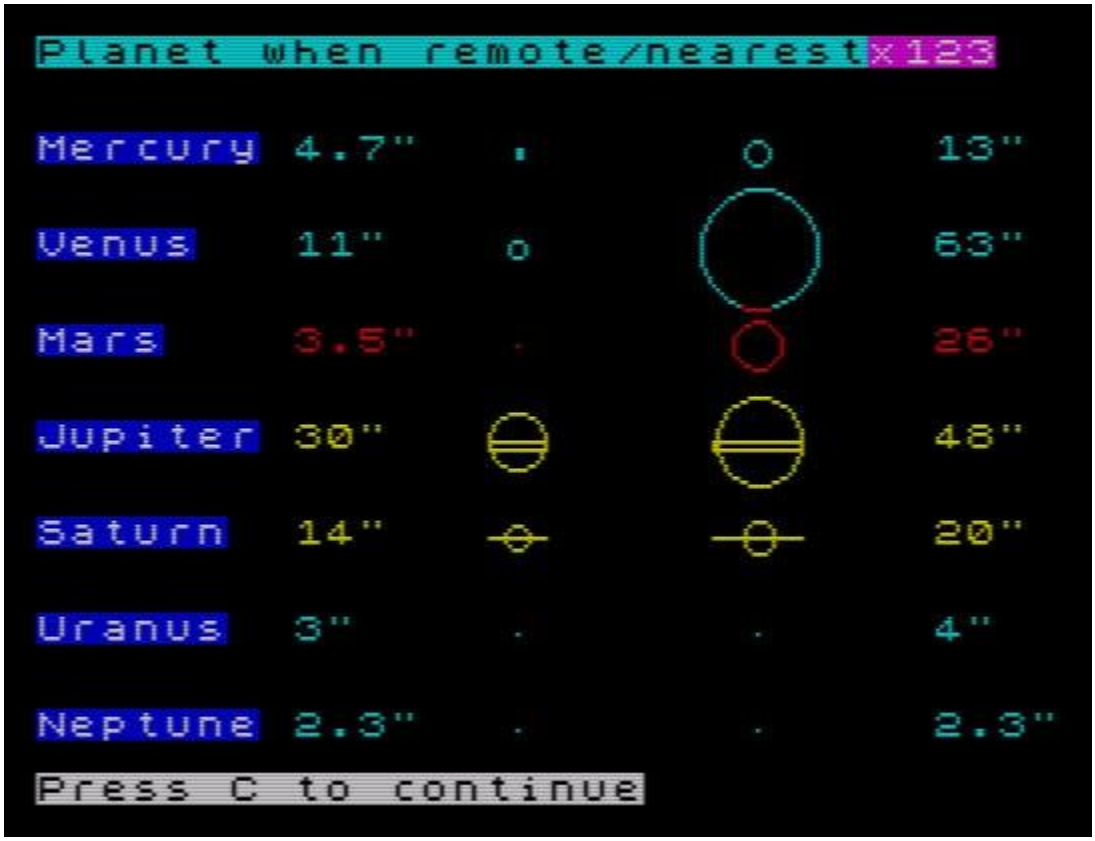
```
10 PRINT "Saturn Draw ";
30 INPUT "Tilt (0" CHR$ 130; "to 28"; CHR$ 130;")",z
40 IF z>28 THEN GO TO 30
50 LET ob=1.1*z/100
60 PRINT "Tilt=";z;CHR$ 130
70 LET e=1/SIN ((.1+z)/180*PI)
80 LET x=115: LET y=88
90 LET r=60: LET h=140
100 GO SUB 190
120 FOR n=1 TO 1.6 STEP .1
130 FOR f=0 TO PI*2 STEP .02
140 LET sx=INT (SIN f*h)
150 LET cy=INT (COS f*h/e)
160 PLOT x+sx/n,y+cy/n
170 NEXT f: NEXT n: STOP
190 PLOT 0,y: DRAW 255,0
200 PLOT x,10: DRAW 0,155
210 FOR f=0 TO PI*2 STEP .02
220 PLOT INT (x+SIN f*r),INT (y+COS f*r*(.89+.11*ob)): NEXT f
240 FOR f=PI*.5 TO PI*1.5 STEP .1: PLOT x+SIN f*r,y+COS f*r/e
250 NEXT f: RETURN
```

Planets through a Telescope

- How large does Jupiter appear in binoculars magnifying x 10?
- Are Mercury and Mars sometimes as small as Uranus in the telescope?
- What range of sizes does Venus appear to be when closest and at its most distant?

This program answers these questions and gives an impression of the relative size of all the planets as seen through a telescope for any magnification up to x 300. Included in the printout to the screen is the maximum and minimum size of each planet in seconds of arc and a scaled disc according to the magnification INPUT. Figure 6.10 is a typical screen copy.

Figure 6.10
Relative size of all the planets in a telescope magnifying x 123.



The display uses a black screen (BORDER 0: PAPER 0: INK 9) and the outline for each planet is coloured via the conditional INK command in Line 110. This command is matched reasonably well to the actual colours of the respective planets — Mars is red, Jupiter and Saturn are yellow, and so on. In the case of Jupiter, two equatorial belts are added and Saturn has an edge-on ring system.

The program

The layout of the program is both economical and straightforward, with the DATA on the planetary diameters, both maximum and minimum, stored in Line 90. These are READ in Line 110 with variable a for the minimum diameter and variable b for the maximum diameter of each planet, using the FOR/NEXT n loop for the seven planets.

The CIRCLE commands in Lines 130 and 140 produce an outline of each planet immediately the DATA is READ but modified according to the magnification as INPUT via the variable c. Lines 160 to 250 PLOT and DRAW the belts and rings of Jupiter and Saturn respectively, and correctly scaled. The planet names are PRINTed to the screen via Line 40. It is often more economical to use this simple arrangement than DATA/READ. Note the three apostrophes (”’) between each planet name — these are to give two blank PRINT lines below each name.

Interpreting the results

The results need to be interpreted if the information is to be meaningful. The program indicates the minimum and maximum size the planets can appear as seen from Earth through a telescope.

The minimum diameter applies when a planet is at its most remote from Earth in what is called the far side of the Sun or ‘superior conjunction’ — ie the Sun is between us and the planet. On these occasions the planet is unobservable due to its apparent proximity to the Sun. (This restriction does not necessarily apply to radio astronomers who could communicate with and receive data from the Viking spacecraft on the Martian surface during these periods, the radio signals being beamed within a degree or so of the Sun’s position.)

Best viewing

When the planets from Mars through to Neptune appear at their largest, they are normally due south at midnight and are seen at their best. Mars’s orbit is eccentric to the extent that the full diameter is not experienced at each ‘opposition’ as it is called, ie opposite to the Sun. In the case of Mercury and Venus, these planets are permanently nearer to the Sun than the Earth and so at times either can appear to pass in front of the Sun’s general position at what is called inferior conjunction. Again, these two planets are unobservable due to their proximity to the Sun even when they are at their closest to Earth and at their largest apparent diameter. On extremely rare occasions, the silhouette of Mercury or Venus can be seen (with special equipment only) in transit across the Sun’s disc.

In general, then, for normal viewing the planets tend to fall into an intermediate size between maximum and minimum. The greatest range of apparent sizes is for Venus, the nearest planet to Earth. The least range for Neptune — the remotest planet.

```
10 REM Planets through 'scope
20 LET d=120: LET e=180: BORDER 0: PAPER 0: INK 9: CLS
30 PRINT PAPER 5;"Planet when remote/nearest"
40 PRINT PAPER
1""Mercury""'"Venus""'"Mars""'"Jupiter""'"Saturn""'"Uranus""'"Neptune"
50 INPUT "Magnification x";c
60 IF c>300 THEN GO TO 50
70 PRINT PAPER 3;AT 0,26;"x";c
90 DATA 4.7, 13, 11, 63, 3.5, 26, 30, 48, 14, 20, 3, 4, 2.3, 2.3
100 FOR n = 1 TO 7
110 READ a,b: INK 5+ (1 AND n>3 AND N <6) - (3 AND N=3)
120 PRINT AT n*3,8; a; CHR$ 34; TAB 28;b; CHR$ 34
130 CIRCLE d, INT (170-n*24), INT (a*c/500)
140 CICRLE e, INT (170-n*24), INT (b*c/500)
150 NEXT n
155 INK 6
160 LET r=c/18: x=c/9
170 LET r1=C/11: LET x1=c/5.5
180 PLOT d,73-c/d
190 DRAW r,0: DRAW -x,0: DRAW 0,c/50: DRAW x,0
200 PLOT d,50
210 DRAW r,0: DRAW -x,0
220 PLOT e,50
230 DRAW r1,0: DRAW -x1,0
240 PLOT e,73-c/d
250 DRAW r1,0: DRAW -x1,0: DRAW 0,c/50: DRAW x1,0
260 PRINT #0; FLASH 1; "Press C to continue": PAUSE 0: RUN
```

Globe-pixel

Since the invention of the telescope and the discovery that the planets are worlds similar in shape to our Earth, astronomers have sliced up each globe into the homely lines of latitude and longitude with a north and south pole and an equator. Of the giant gas planets Jupiter, Saturn, Uranus and Neptune, only Jupiter and, to a lesser extent, Saturn have such systems of any practical interest. No positive markings (other than banding of polar and equatorial regions) have been detected on Uranus and Neptune due to their extreme remoteness. Only rarely does Saturn reveal any identifiable markings that are carried across the disc by the planet's rotation. This leaves the relatively close (typically four to six times the distance of the Sun) planet Jupiter. A wealth of detail is visible on the disc in amateur telescopes but there is a snag.

Jupiter, being a rapidly spinning gas world, has no fixed reference point — only the constantly shifting cloud top is revealed. Also the rotation of the equatorial cloud zone is faster than at higher latitudes, and the planet is distorted from a neat sphere to an oblate spheriod shape. Thus, although a grid system of latitude and longitude is possible via the Spectrum graphics which even accounts for the oblate shapes of Jupiter and Saturn, it was not quite what we had in mind. A journey closer to Earth is called for.

Mars, Mercury and the Moon

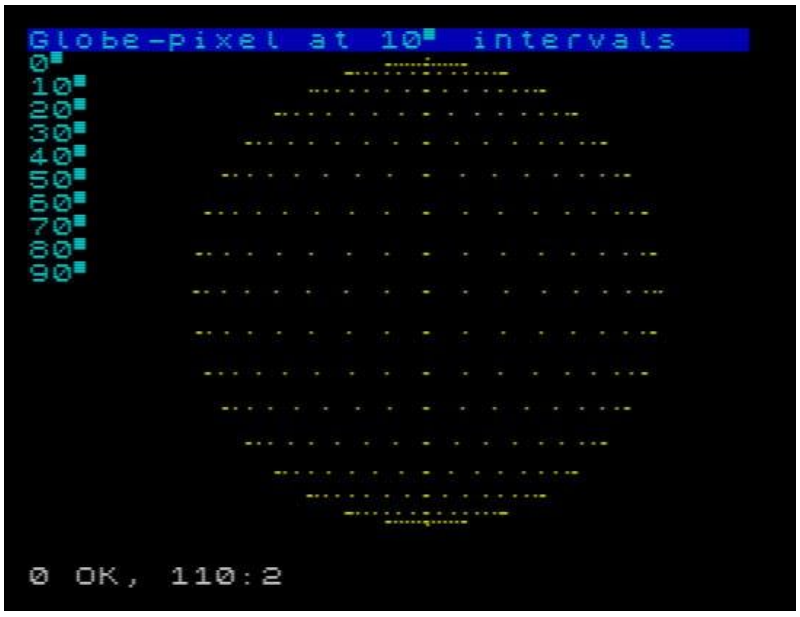
The four inner planets from Mercury to Mars (including Earth) all have rocky surfaces. Only Venus with her permanent cloak of dense cloud is unwilling to reveal all. To this foursome, the Moon should be added as Earth's twin planet. All are ripe for dissection into neat parcels of latitude and longitude as a globe projection. The two that are of practical interest to amateur astronomers are the Moon, somewhat obviously, and Mars. Mars is the only planet apart from Earth where the changing seasons and rotation-carrying surface markings across the disc can be seen using a back-yard telescope. The two programs that follow are particularly useful in converting the normal flat Mercator-type projection of, say, Mars's maps to that of a globe — making for easier recognition of features, especially when displayed in the limb regions, ie adjacent to the disc edge.

The program

The short Globe-pixel routine can be used to PLOT a globe divided at 10° intervals of latitude and longitude. Figure 6.11 is a COPY from the screen. The scale of the COPY to the ZX printer gives a 5 cm diameter disc — the diameter recognised by amateur specialist observers of Mars and Venus for sketching these planets as basic outlines.

Figure 6.11

Globe accurately plotted at 10° intervals of latitude and longitude.



The program uses two FOR/NEXT loops to PLOT each pixel — the z loop for the lines of longitude (vertical lines) and the n loop for the lines of latitude (horizontal lines). The results are precise and accurate as presented to the screen and to the ZX printer. The separation of each pixel is controlled by the STEPs in the FOR/NEXT loops as follows:

FOR z= .001 TO 91 STEP 10

where STEP 10 is for each 10° of longitude. Note that although these STEPs are at precisely 10° intervals, the actual value returned has the sequential value of:

.001, 10.001, 20.001, 30.001, 40.001, etc.

Adding .001° (1/1000 degree or 3.6″ (seconds) arc) has no visible effect on the PLOTted positions but does ensure that the program will not crash trying to evaluate the SIN of 0° – an infinite number. The n loop takes the form:

FOR n = 0 TO PI*2 STEP 1 /r* 10 (latitude lines)

where PI*2 gives a full circle and STEP 1/r*10 equals a STEP interval of 0.175 from (1/(180/PI))*10 producing the required 10° spacing in latitude.

The lines of latitude are parallel to the equator whilst the curved lines of longitude originate from the poles. The latter PLOTting is controlled by the variable c in Line 60 to produce the curvature in Line 90.

Jupiter simulation

With a very minor amendment to the program it is possible to simulate Jupiter's oblate globe — in this case, expanding the equatorial regions and leaving the polar or vertical dimensions unchanged. Amend the following line:

90 LET b = COS n*80/c* 1.08

It is the last value in the expression, ie * 1.08, that does the necessary expansion. Figure 6.12 shows Jupiter's shape.

Figure 6.12

The oblate figure of planet Jupiter shown by reducing the STEP value to 1 in the z FOR/NEXT loop.


```
3 REM Globe-pixel
10 BORDER 0: PAPER 0: INK 5: CLS
20 PRINT PAPER 1;"Globe-pixel at 10"; CHR$ 130;" intervals "
30 LET r=180/PI
40 FOR z=.001 TO 91 STEP 10
50 PRINT INT z;CHR$ 130
60 LET c=1/SIN (z/r)
70 FOR n=0 TO PI*2 STEP 1/r*10
80 LET a=SIN n*80
90 LET b=COS n*80/c
100 PLOT INK 6;INT (137+b),INT (85+a)
110 NEXT n: NEXT z
```

Globe Projection

This program DRAWs a globe divided into lines of latitude and longitude at 10° intervals rather than as a single pixel to mark each division as via the Globe-pixel program.

This program is not as accurate as the previous routine because it uses the Spectrum's DRAW command which, although executed rapidly, cannot produce the required elliptical lines but only simple arcs. The largest errors, such as they are, occur adjacent to the edge of the disc and in the polar regions. Nevertheless, the simulation is quite effective and the user has the option to show either a polar or an equatorial view. The latter may be tilted up to 12° in a north or south direction. Figures 6.13 and 6.14 are typical COPYs from the Spectrum screen. The REM statements show the general structure of the program.

Figure 6.13

The globe can be drawn for a polar or equatorial viewpoint. The latter can be tilted up to ±12°.

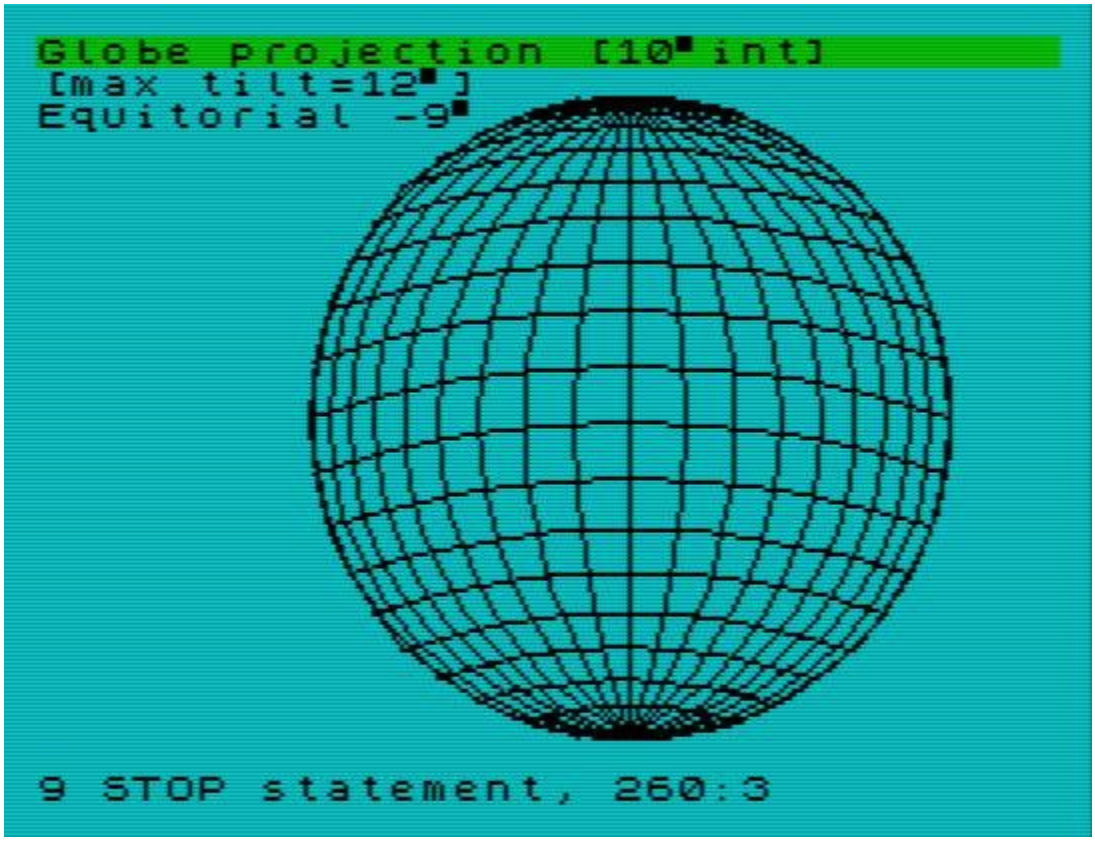
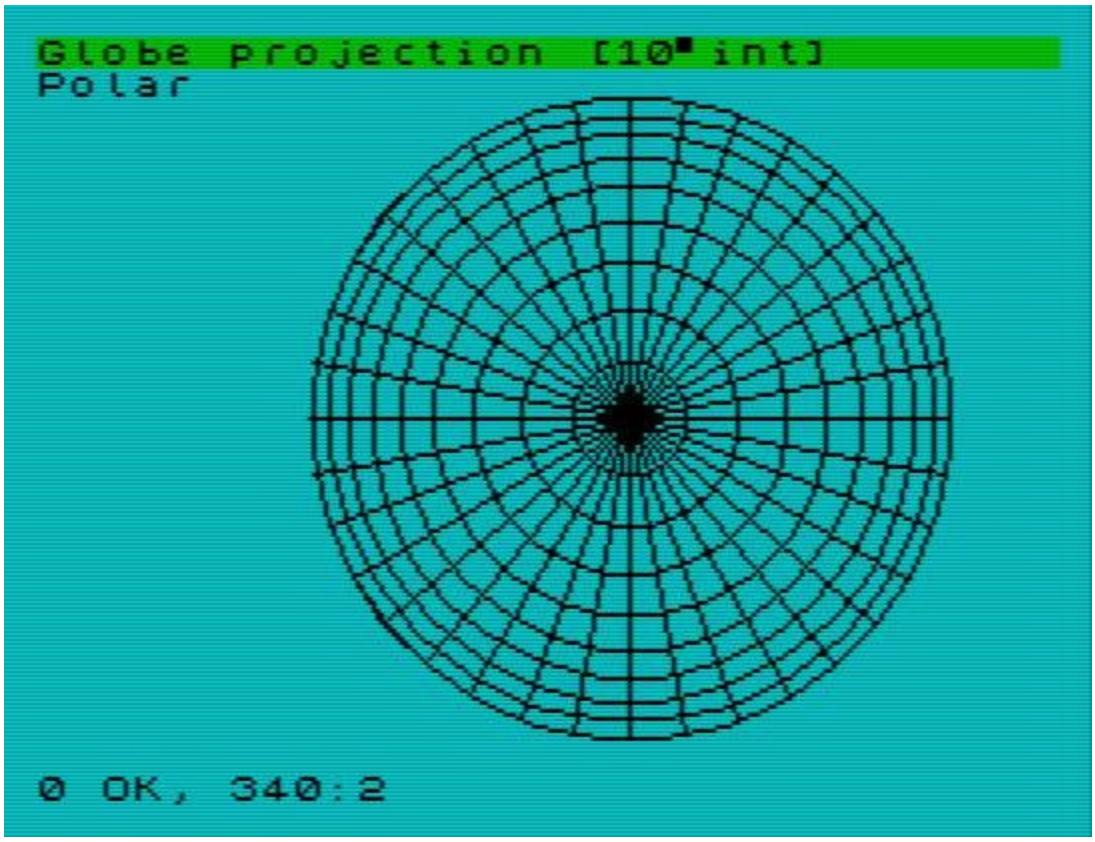


Figure 6.14

Polar viewpoint.




```
10 REM Globe Projection
30 BORDER 5: PAPER 5: CLS : PRINT PAPER 4;"Globe projection [10"; CHR$ 130;" int] "
40 LET a=148: LET b=80: CIRCLE a,b,b
60 DIM x(9): DIM y(9)
70 FOR n=1 TO 9: READ y(n)
80 LET x(n)=63*(1.24*ATN (PI/180*n*10.5)): NEXT n
90 DATA 79,75,70,64,56,47,36,27,0: RESTORE
110 INPUT "Polar or Equitorial (p or e)? "; LINE a$: GO TO
(a$="e")*120+(a$="p")*270
130 PRINT "[max tilt=12 HR$ 130;"'"Equatorial ";
135 INPUT "Tilt (-s): ";k: IF ABS k>12 THEN GO TO 135
140 PRINT k;CHR$ 130: LET k=k/14
150 FOR n=0 TO 36 STEP 2
160 LET f=n-18: LET g=2.71
170 LET h=g*ATN (PI/180*-f*7.3)
180 PLOT a,0: DRAW 0,b*2,h
190 NEXT n
210 PLOT a-b,b: DRAW b*2,0,k
220 FOR n=1 TO 9: LET c=y(n)
```

```
230 PLOT a+1-x(10-n),b+x(n)
240 DRAW c*2,0,k
250 PLOT a+1-x(10-n),-b+x(n)
260 DRAW c*2,0,k: NEXT n: STOP
280 PRINT "Polar"
290 FOR n=0 TO 72 STEP 2
300 LET d=n/36*PI: LET z=b*SIN d: LET yy=b*COS d
310 PLOT a,b: DRAW z,yy: NEXT n
330 FOR n=1 TO 9
340 CIRCLE a,b,x(n): NEXT n
350 STOP
```

Chapter 7 – Star Systems (Tri-star Orbits)

Our Sun is a pretty average and unexciting star as things go in the universe. The evolution and survival of life-forms on this planet have depended on its constant output of light and heat over hundreds of millions of years without solar fireworks. Also, the Sun is fairly small in star terms and this bodes well for its longevity. Larger stars have a habit of burning up their fuel supplies in double-quick time.

Astronomers have a scale called absolute magnitude by which to judge the true brightness of stars by assuming that they are all the same distance from Earth. This scale operates at a distance of 10 'parsecs', ie about 33 light years away.

Our sun placed at this distance would only just be visible on a dark clear night to the unaided eye with an apparent magnitude of 4.7 (see the **Stellar Magnitude program in Chapter 8** for an explanation of apparent magnitude). In contrast, the star Rigel in Orion would be magnitude – 7 at this distance, far out-shining even planet Venus at her best. At the opposite end of the scale, to show that our Sun is average, the nearest star to Earth (excluding the Sun) is Proxima Centauri, a red dwarf, and this would fade from its current brightness of 10.7 to about 15.0 at 10 parsecs.

Our Sun is also fairly conservative in not possessing a companion star to orbit around — it has a nice family of planets instead. The two are not mutually exclusive one would assume, but sharing the gravitational fields of two stars could play havoc with orbital distances and weather prospects of an inhabited planet. The following short programs enable double and triple stars to orbit each other — the sort of object that attracts the attention of astronomers rather than solitary magnitude 4.7 stars.

Tri-star Orbits

This is a modelling program to place two companion stars in orbit about a central star. The diameter of all three stars can be selected at will and each may be different. The orbit itself can also be tilted at any angle from 0° (edge-on viewpoint) to 90° (plan viewpoint). Particularly interesting orbital appearances occur with fairly large diameter stars (INPUT say 10 to 20) and a low inclination of orbit tilt (INPUT say 0 to 20). Here mutual eclipses of the stars are possible.

The program uses two FOR/NEXT loops, the f loop to compute the orbit and the n loop to CIRCLE the star's image on the orbit. The n loop uses the OVER command so that each star is first drawn then deleted to give an impression of movement in orbit. Each star leaves a trail behind via the PLOT commands in Lines 130 and 140. The central star is only drawn once via Line 90 and remains fixed for the presentation.

The program

The basis of this program is yet another version of DRAWing ellipses. If the variable z is omitted from Line 120 (together with the / sign) then each orbit would be circular. It is this variable, via Lines 30 and 40, which compresses the circle into an ellipse according to the INPUT value for tilt. The maximum radius of the orbit is set by variable h to a value of 60. This is done to avoid the program crashing where an INPUT Of 90° for tilt and an INPUT of 20 for the diameter of orbiting stars 2 or 3 meant that the CIRCLE command would exceed the screen limit at top and bottom. Figures 7.1 and 7.2 illustrate typical orbits.

Figure 7.1

Two smaller stars orbit a massive central star.

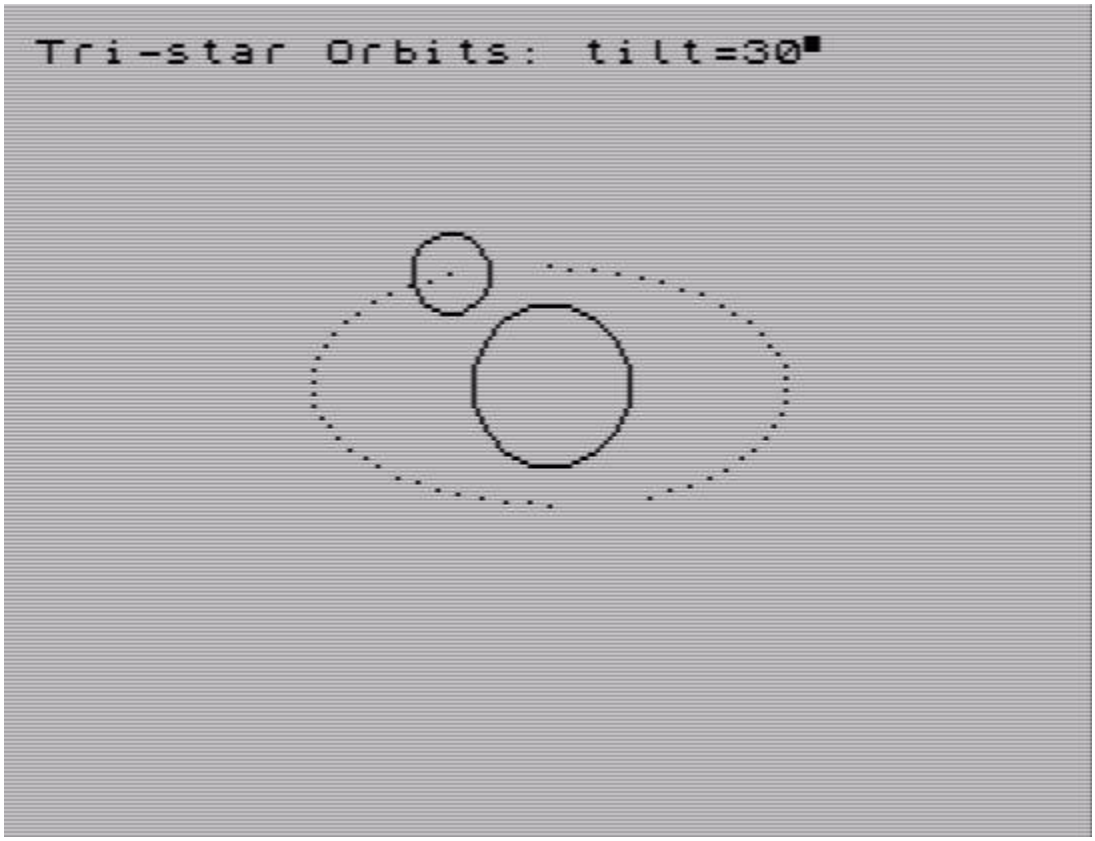
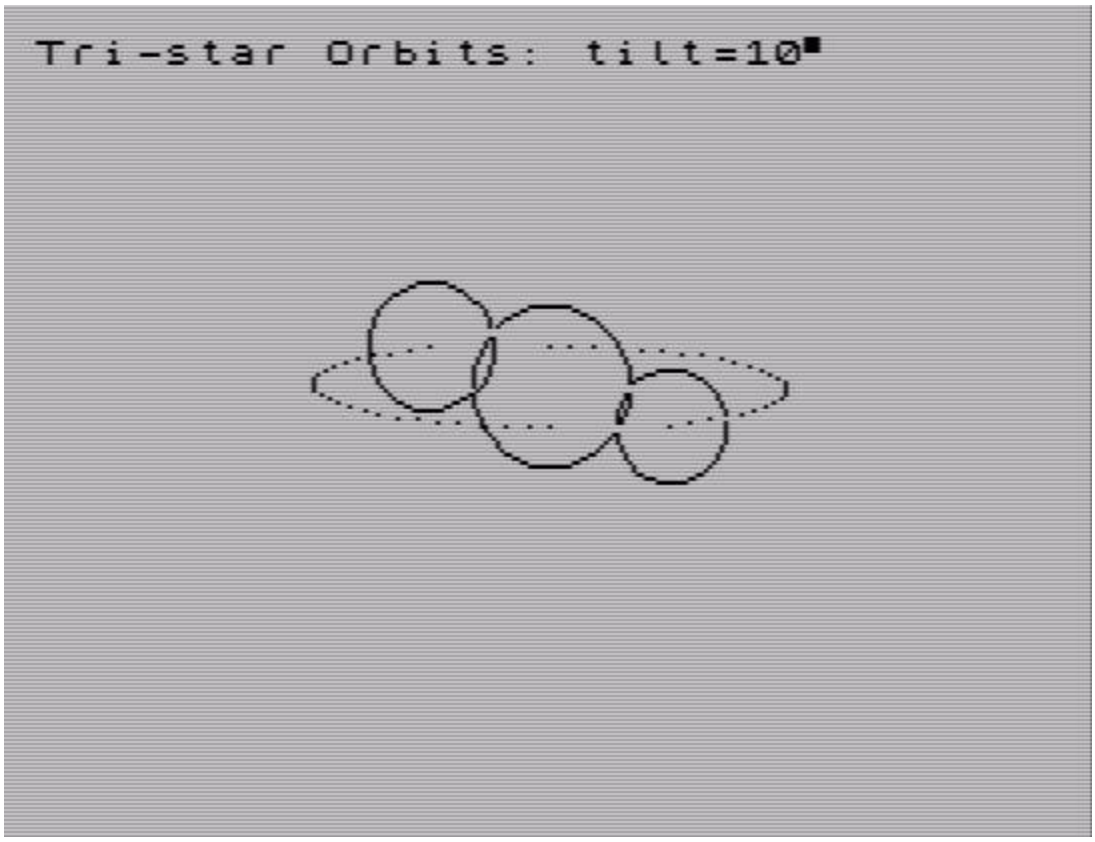


Figure 7.2

Three stars of comparable diameter with orbits near edge-on.



Lagrangian orbits

The single orbit shared by two companion stars is relatively unstable. Minor perturbations (gravitational disturbances) will cause a neat system, with two objects diametrically opposed in the same orbit, to fail. One star will advance on the other with a fair chance of collision and annihilation or absorption of the lesser of the two stars.

Having said this, there are in fact two points on an orbit where a lesser star (or planet for that matter) is reasonably safe from such a catastrophe. These are called the Lagrangian points. Located some 60° ahead or behind the other orbiting body, each forms a perfect equilateral triangle with the central star. As the program stands, only one orbital position is calculated via Lines 110 and 120. The second star's position is merely mirrored by Line 170 with the negative values. Amend the program thus to simulate two stars in Lagrangian orbits:

```
111 LET sx1=INT(SIN(f + 2)*h)
121 LET cy1=INT(COS(f + 2)*h/z)
140 PLOT x-sx1,y-cy1
170 CIRCLE x-sx1,y-cy1,d3
```

By adding a value of about 2 to f in Lines 111 and 121, the second body is advanced in its orbit so as to form an equilateral triangle between all three bodies. Obviously the orbit appears compressed for any orbit tilt less than 90° — the plan view.

It should be noted that beyond some asteroids associated with planet Jupiter and its orbit, we have no evidence of Lagrangian orbits for stars. It is reasonable to assume that if such star systems did exist, the orbit would be too large in relationship to the stars' diameters to have any reasonable stability and to avoid tidal effects on the stars' surfaces.

```
10 CLS : PRINT "Tri-star Orbits: ";
30 INPUT "Orbit tilt (0"; CHR$ 130; "to 90"; CHR$ 130;")";z
35 PRINT "tilt=";z; CHR$ 130
40 LET z=1/SIN ((.1+z)/180*PI)
50 INPUT "Star 1 diameter (1-20): ";d1
60 INPUT "Star 2 diameter (1-20): ";d2
70 INPUT "Star 3 diameter (1-20): ";d3
80 LET x=128: LET y=88
90 LET h=60: CIRCLE x,y,d1
100 FOR f=0 TO PI*2 STEP .1
110 LET sx=INT (SIN f*h)
120 LET cy=INT (COS f*h/z)
130 PLOT x+sx,y+cy
140 PLOT x-sx,y-cy
150 OVER 1: FOR n=0 TO 1
160 CIRCLE x+sx,y+cy,d2
170 CIRCLE x-sx,y-cy,d3
180 NEXT n: OVER 0: NEXT f
```

Binary-star Orbit

This program is, in comparison to the Tri-star Orbit, a little tame, but at least it is known to exist fairly commonly for star systems. In fact a fairly high percentage of stars prove to be binaries and the expression had been in use long before its adaptation to computer terminology! The sample screen COPYs show typical results by varying the INPUT of orbit tilt and diameter of the two component stars. As previously mentioned for the Tri-star Orbits, low orbital tilt and large diameter stars simulate eclipse effects. See Figures 7.3 and 7.4.

Figure 7.3
A single companion star orbits a massive primary star.

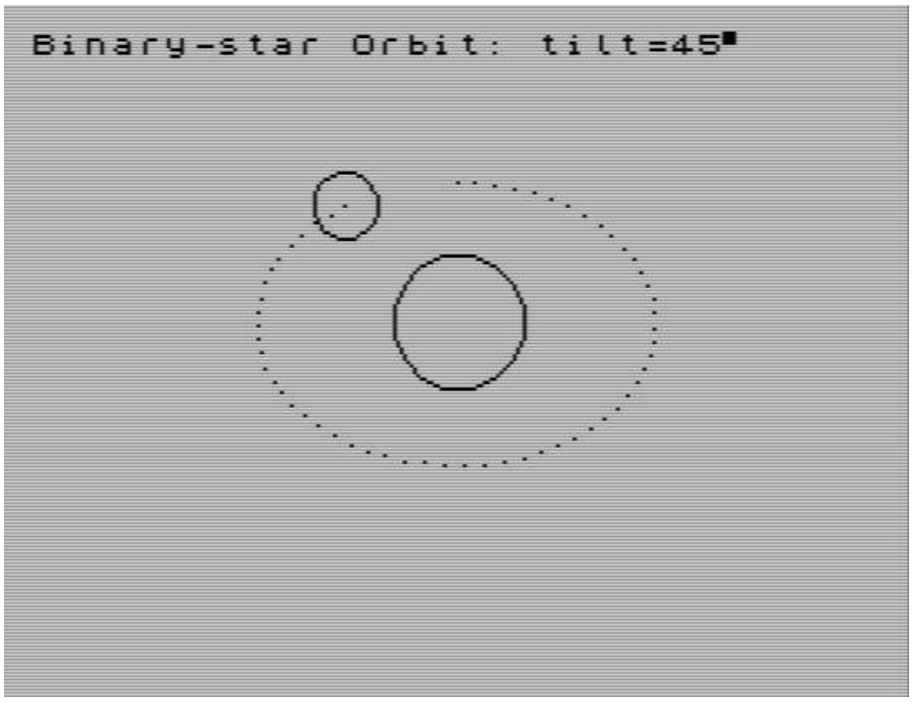
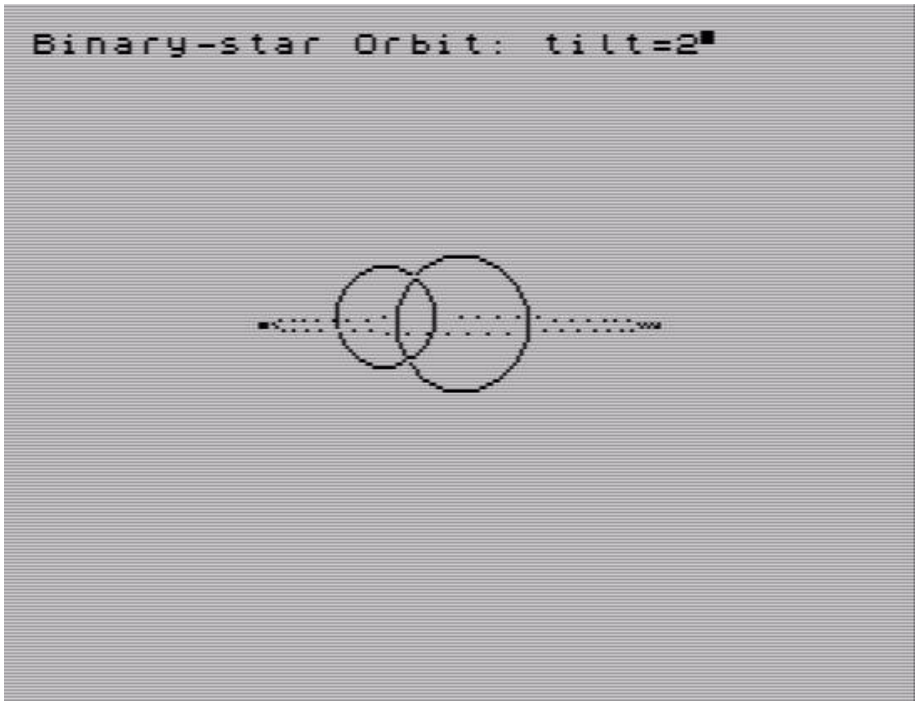


Figure 7.4
Mutual eclipses occur at low angles of orbital tilt.



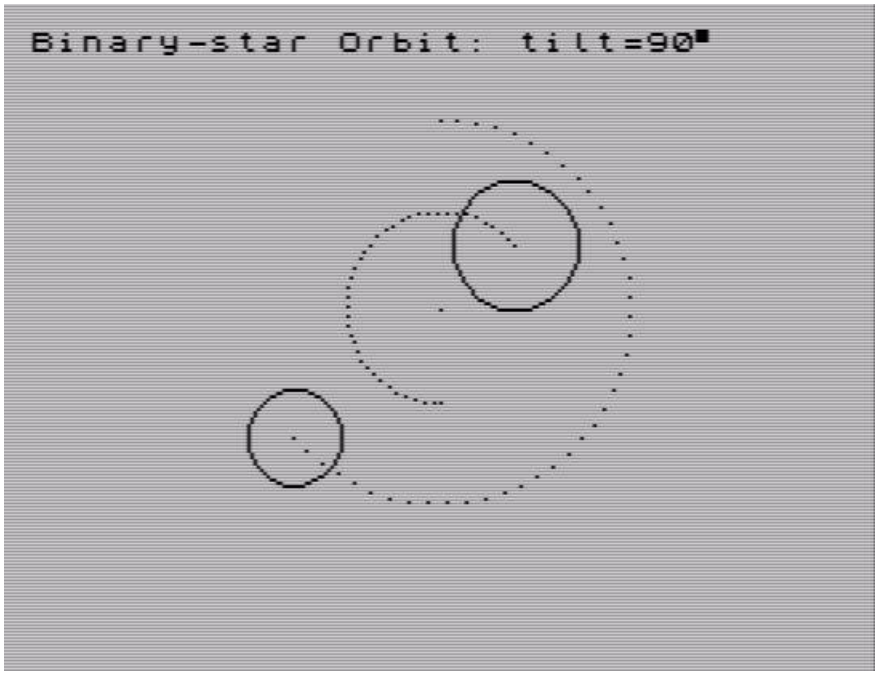
Mutual centre of gravity

You will notice that in both programs the central star remains fixed, implying that the mass of the orbiting star (or

stars) is insignificant. However, in some star systems the mass of the component stars may not be too dissimilar and so both stars orbit about the common centre of gravity for the system, as in Figure 7.5.

Figure 7.5

Two stars of comparable mass orbit around a common centre of gravity marked with a single pixel.



As such a binary bears a greater relationship to the Tri-star Orbit with the above modification for a Lagrangian orbit, it is easier to deal with this program as a further modification to the Tri-star Orbits program. However, assuming that the Binary-star Orbit has been keyed in, the modifications for orbits about a mutual centre of gravity are as follows:

```
80 LET h=60: PLOT x,y
101 LET sx1=INT (SIN f*h/2)
111 LET cy1=INT (COS f*h/z/2)
121 PLOT x - sx1, y-cy1
141 CIRCLE x-sx1, y-cy1,d1
```

The PLOT in Line 80 puts a single pixel at the centre of gravity for the system Lines 101 and 111 calculate the orbital position of the central star now shifted from a fixed point. The last value, 2, controls the orbital radius of the central star. Try other values from 1 to 5 and see the effect created. A value of less than 1 (eg 0.7) will make the former central star orbit further out than the so-called outer star. Beware that the orbits are still contained on the screen, otherwise the program will crash.

```
10 CLS : PRINT "Binary-star Orbit: ";
30 INPUT "Orbit tilt (0"; CHR$ 130; "to 90"; CHR$ 130;") ";z
35 PRINT "tilt=";z;" CHR$ 130
40 LET z=1/SIN ((.1+z)/180*PI)
50 INPUT "Star 1 diameter (1-20): ";d1
60 INPUT "Star 2 diameter (1-20): ";d2
70 LET x=128: LET y=88
80 LET h=60: CIRCLE x,y,d1
90 FOR f=0 TO PI*2 STEP .1
100 LET sx=INT (SIN f*h)
110 LET cy=INT (COS f*h/z)
120 PLOT x+sx,y+cy
140 OVER 1: FOR n=0 TO 1
150 CIRCLE x+sx,y+cy,d2
170 NEXT n: OVER 0: NEXT f
```

Spirals

I wouldn't want to suggest that the following short programs have any strict relevance to astronomy. Mainly, I want to find a use for the spiral graphics the Spectrum produces so easily! There are three areas where such shapes do have some connection with astronomy:

1. Spiral form of our Milky Way galaxy.
2. Some exotic binary stars losing matter to space as a gas trail.
3. Shadow of an eclipsing binary traversing space.

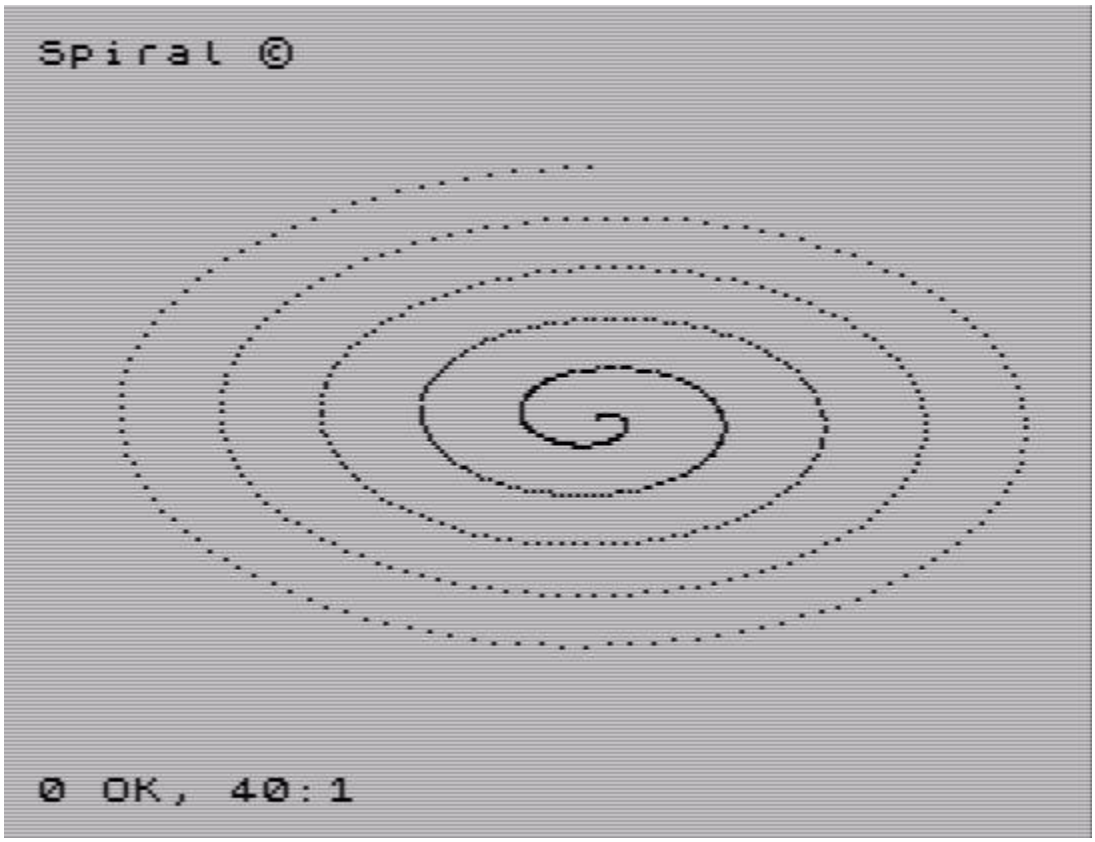
Eclipsing binary shadow

The basic Spiral program could well mimic the third option for the eclipsing star Algol (the Demon Star) in Perseus. Here a fainter companion eclipses the brighter star every 70 hours as seen from our direction in space. Algol is about 100 light years away so we do not see these events instantly but 100 years and some 12,500 eclipses later. The intervening space between Algol and Earth is separated at 70 light hours distance by this 'shadow event' in one continuous spiral as if from a giant LP record centred on Algol. The program depicts only the first five sweeps of the shadow event in the vicinity of Algol to a distance of about 15 light days at hourly intervals.

The program (in Figure 7.6) also poses an interesting paradox concerning the finite speed of light (and any physical form) at 300,000 km/s. As the orbiting of Algol's companion is constant, like the revolutions of a record, then the velocity of the shadow event on this disc, away from the centre, will soon exceed the speed of light by a

factor of thousands in the vicinity of Earth. Thus although light has a finite speed it would appear that a shadow (the absence of light) can move at infinite speed. Or can it?

Figure 7.6




```
1 PRINT "Spiral @"
10 FOR f=0 TO PI*10 STEP .05
20 PLOT 140+SIN f*40*f/10,80+COS f*40*f/20
40 NEXT f
```

Spiral galaxy

The programs Twin Spiral (Figure 7.7) and Twin Spiral 2 (Figure 7.8) could well represent the spiral form of many galaxies — the largest known objects in the universe. Only two 'arms', as they are called, are depicted and some galaxies may have many more.

The program is identical to the first of this series, Spiral, except that a second line of PLOTting (for the second arm) is included in Line 30. Note that the SIN and COS values are now negative, causing the second spiral to be PLOTted 180° from the first spiral.

The PLOTting is controlled by the FOR/NEXT f loop for five orbital sweeps (PI*10) with a STEP value of 0.05. Changing the STEP value will alter the PLOTting interval. In Twin Spiral 2 the arms are more widely spaced by halving the values applied to variable f in Lines 20 and 30 in the form:

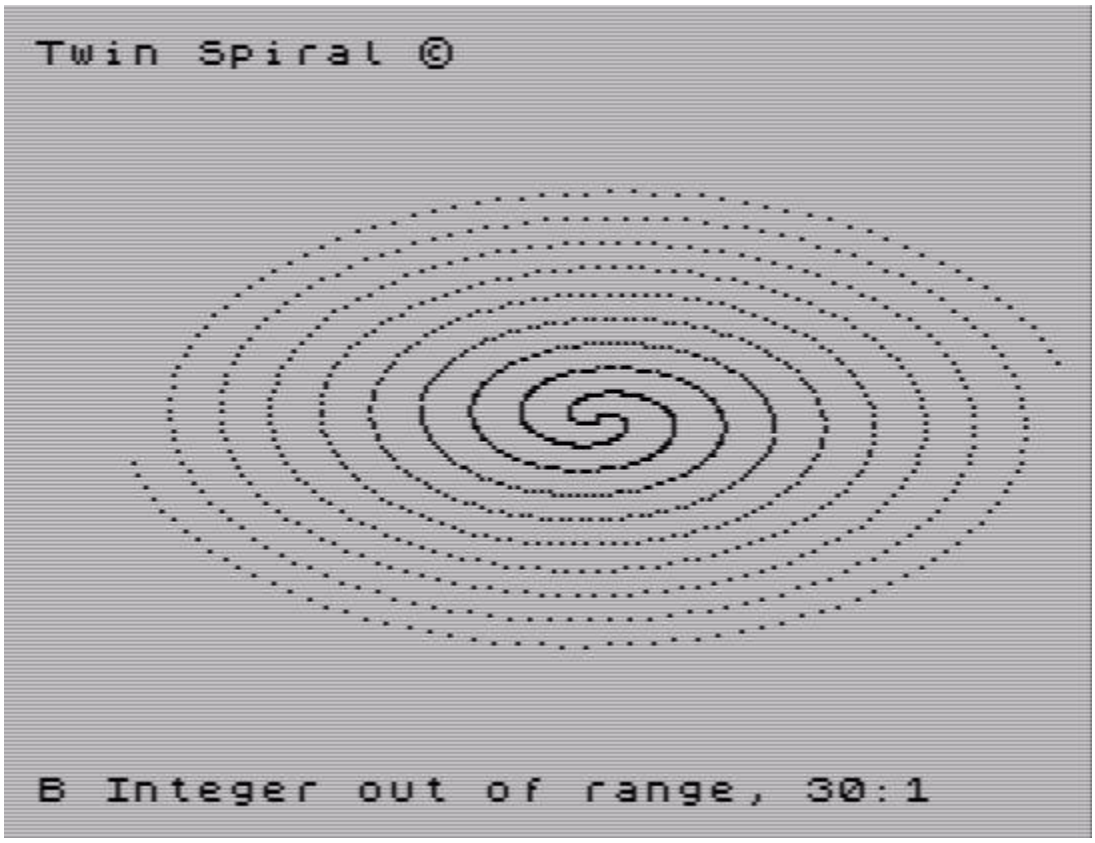
SIN … f/5 and COS … f/10

By having the second value (10) double that of the first (5), the spiral form is displayed as if inclined to the observer where the major (horizontal) axis is twice that of the minor (vertical) axis. Try amending these proportions with perhaps a larger or smaller number in the form:

COS … f/(new value)

This must be done with identical values in both Lines 20 and 30. Some permutations may cause the program to crash by attempting to PLOT beyond the screen confines.

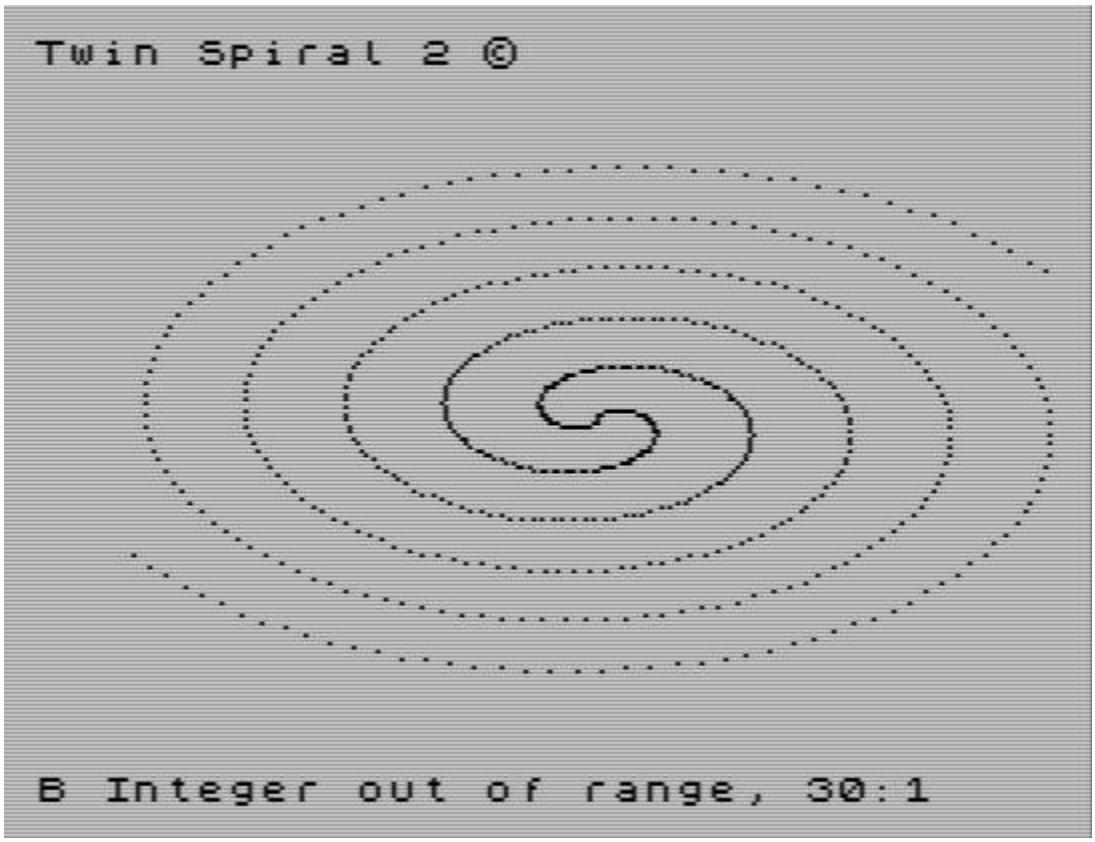
Figure 7.7




```
1 PRINT "Twin Spiral @"
10 FOR f=0 TO PI*10 STEP .05
20 PLOT 140+SIN f*40*f/10,80+COS f*40*f/20
```

```
30 PLOT 140-SIN f*40*f/10,80-COS f*40*f/20
40 NEXT f
```

Figure 7.8




```
1 PRINT "Twin Spiral 2 @"
10 FOR f=0 TO PI*10 STEP .05
20 PLOT 140+SIN f*40*f/5,80+COS f*40*f/10
30 PLOT 140-SIN f*40*f/5,80-COS f*40*f/10
40 NEXT f
```

Stargas Spiral

The last of this series (Figure 7.9) performs precisely as the Twin Spiral program but has been rewritten in a neater form. Two small CIRCLES are added to represent a close binary system spraying matter into surrounding space as the stars orbit each other in violent conflict. Our knowledge of such possible events has not been witnessed at first hand but has been deduced by analysis of the spectrum (with a small s you will note!) via the spectroscope on giant telescopes. You could give this program a little more realism by using as a direct command before RUNnng:

```
BORDER 0: PAPER 0: INK 2: CLS: RUN
```

You can also add a few coloured fireworks to the point of the gas trails origin between the two stars with an additional line:

```
12 PLOT INK RND*9; a,b
```

As this line will affect the whole character square containing the two stars, these will appear involved in the action too.

Figure 7.9


```
1 BORDER 0: PAPER 0: INK 2: CLS : PRINT "Stargas Spiral @"
3 LET a=140: LET b=80
5 CIRCLE a+3,b+3,3
6 CIRCLE a-3,b-3,3
10 FOR f=0 TO PI*10 STEP .05
12 PLOT INK RND*9;a,b
15 LET x=SIN f*40*f
16 LET y=COS f*40*f
20 PLOT a+x/5,b+y/10
30 PLOT a-x/5,b-y/10
40 NEXT f
```

Galaxy

This program enables the user to simulate the probable appearance of our Milky Way galaxy as seen from intergalactic space. It uses an inverse video type of presentation (BORDER 0: PAPER 0: INK 7) and appears almost photographic in clarity. The screen COPYs, Figures 7.10 and 7.11, are mere shadows of the screen effect.

The program

The user may tilt the galaxy at any angle from 0° (edge-on) to 90° (plan view) via the variable t. The program itself is divided into five sections as the REM statements indicate. It may be of interest to expound on these a little, starting with the galaxy DRAW routine from Line 100 to 150. Here two FOR/NEXT loops PLOT the stars to form six spiral arms in Line 130 and the galactic centre in Line 140. The constant use of RND in this PLOTting produces a clumping effect of star distribution which is known to exist. The actual length of the FOR/NEXT n loop is controlled by the variable tt (which in turn has the initial value of t — the INPUT tilt) and limits the total number of stars PLOTted in this section. Effectively, at low angles of tilt fewer stars would be seen because of the intervening dust between them, so few stars are actually PLOTted.

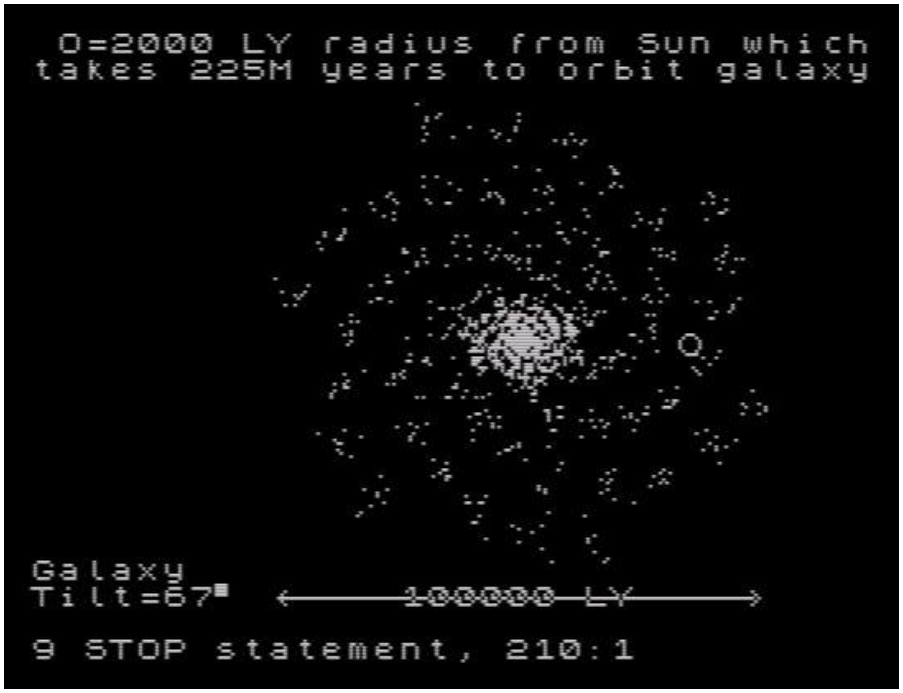
The next sequence from Line 160 PLOTs the haze of 'globular clusters' that form a spherical shell which orbits the galactic centre. Each pixel represents hundreds of thousands of stars.

Figure 7.10

Our Milky Way galaxy almost edge-on. All the stars seen from Earth (without optical aid) are within the small circle on the right.

Figure 7.11

The galaxy's spiral arms become evident as the system is tilted.



Screen comments

This section from Line 220 is used to punctuate the PLOTting with PRINT comments for each sequence. Notice how each PRINT line is the same length, 32 characters. In this way it is not necessary to delete the previous PRINT comment but simply to overPRINT it with the next. Line 230 locates the PRINT statement as a separate GOSUB routine.

The final sequence from Line 270 indicates the immensity of the universe by CIRCLEing a 2000 light year radius about the solar system (which would be totally invisible to this scale). But a telescope on the planet Earth can see up to 15,000,000,000 light years into intergalactic space with a collecting surface only 200 inches across (Hale Reflector). This is the equivalent, on a typical screen presentation of this program, of seeing over 20 miles in all directions from your TV set, from a subatomic particle in the small circle centre right. The subatomic particle would be about 0.000 000 000 000 01mm diameter!

```
10 REM Galaxy
20 BORDER 0: PAPER 0: INK 7
30 CLS : LET z=75: LET x=148: LET y=80
50 PRINT AT 20,0;"Galaxy"'"Tilt=";: INPUT "0"; CHR$ 130; "to 90; CHR$ 130;t
60 PRINT t; CHR$ 130; TAB 9;"< 100000LY >": LET tt=t
70 PLOT 76,3: DRAW OVER 1;143,0: GO SUB 230: GO SUB 240
80 LET t=1/SIN ((.1+t)/180*PI)
100 LET q=4+1/t*6
```

```
110 FOR n=1 TO tt/10+2
120 FOR f=1 TO z
130 PLOT x+f*(SIN f)+RND*q,y+f*COS f/t+RND*q-2
140 PLOT x+f*SIN f/4+RND*3,y+f*COS f/6+RND*2
150 NEXT f: NEXT n
170 GO SUB 230: GO SUB 250
180 FOR f=1 TO z
190 PLOT x-10+f*SIN f+RND*20,y-10+f*COS f+RND*20: NEXT f
200 GO SUB 230: GO SUB 260
210 STOP
230 PRINT AT 0,0;: RETURN
240 PRINT FLASH 1;" Galactic centre & spiral 'arms'": RETURN
250 PRINT FLASH 1;" surrounding globular clusters ": RETURN
260 PRINT FLASH 1;" O=2000 LY radius from Sun whichtakes 225M years to orbit
galaxy"
280 CIRCLE FLASH 1;x+52,y,3
290 RETURN
```

Chapter 8 – Starcharts (Starmaps)

Starcharts, display a simple starmap

Starcharts are the stock-in-trade of the practical astronomer and, for sheer volume of numbers of stars recorded, the Spectrum (or the sane user of any micro) could not hope to compete. For example the *Tirion Star Atlas 2000.0* (referred to as simply *Tirion* throughout this book) plots over 45,000 stars and deep-sky objects. This almost exceeds the pixel content of the Spectrum screen. Bearing in mind that a starchart is 99% empty space, the task for a Spectrum user is considerable as all the stars in *Tirion*can be seen with only 10 x 50 binoculars. A small telescope can reveal millions!

The user of starcharts in microcomputer work should concentrate on techniques of handling data or rapid paging of results for display rather than precise accuracy of plotting of huge numbers of stars. The programs that follow are aimed in this direction.

Starmaps (48K Spectrum only)

Drawing starmaps is in itself not the most fruitful occupation for your Spectrum. You would be better off consulting a good book, like *Norton* or *Tirion*. Furthermore, atlases like these have much higher positional accuracy and graduation of dot size against stellar magnitude than is possible on any micro or even minicomputer. This disregards all the other DATA of star names or codes, variable stars and deep-sky objects from the Messier and New General Catalogues which would clutter up the screen display and make it incomprehensible.

Nevertheless, the instant display of a selected, if simple, starmap can prove satisfying: as an astronomer, I feel that there is a need to challenge the abysmal quality of some published efforts in this area.

Perhaps more importantly, this program demonstrates a technique for handling and using DATA stored in CHR$ arrays for starplotting.

In its final form, you can select by number (as indicated in Figure 8.1) from one of three options to display any one of 30 named constellations as star patterns, all correctly orientated (north at the top, east on the left) on a large scale. The sky coordinates in RA and Dec for the centre of the screen are displayed together with a scale in degrees.

Figure 8.1
Starmap options selected by number.

Alternatively, you can display a starmap to a reduced scale of all the constellations visible during any one of the four seasons, with each constellation named as it is PLOTted.

The final option is a complete starmap of all 30 constellations containing all 310 stars in one panoramic strip of sky. The reduced scale of the latter highlights the limited screen resolution mentioned in the introduction to this chapter, but the results are still pleasing.

COPYing the screen

The starmaps are shown in inverse video by use of a black sky (PAPER 0) with white stars (INK 9, contrasting with the PAPER tone). An option to COPY the screen to the ZX printer is also included but of course the starmaps appear as negatives — black stars on a white ground. It is best to accept a COPY in this form because quite frankly the ZX printer cannot handle a true INVERSE presentation, which in the case of a starmap would be 99.9% solid black! With the ZX printer in mind the starmaps have been outlined and both the monochrome and colour presentation to the screen have been fully considered. Sample COPYs of the starmaps are shown adjacent, in Figures 8.2, 8.3, 8.4 and 8.5.

Figure 8.2

Aquila, the first of 30 constellations, shown full size. The ZX printer normally gives a negative image of the screen display.

Figure 8.3

A complete starmap of 30 constellations and 310 stars. Ursa Major (the Plough) is top centre and Square of Pegasus extreme left.

Figure 8.4

Starmap for winter with Orion lower centre and Leo (the lion) extreme left.

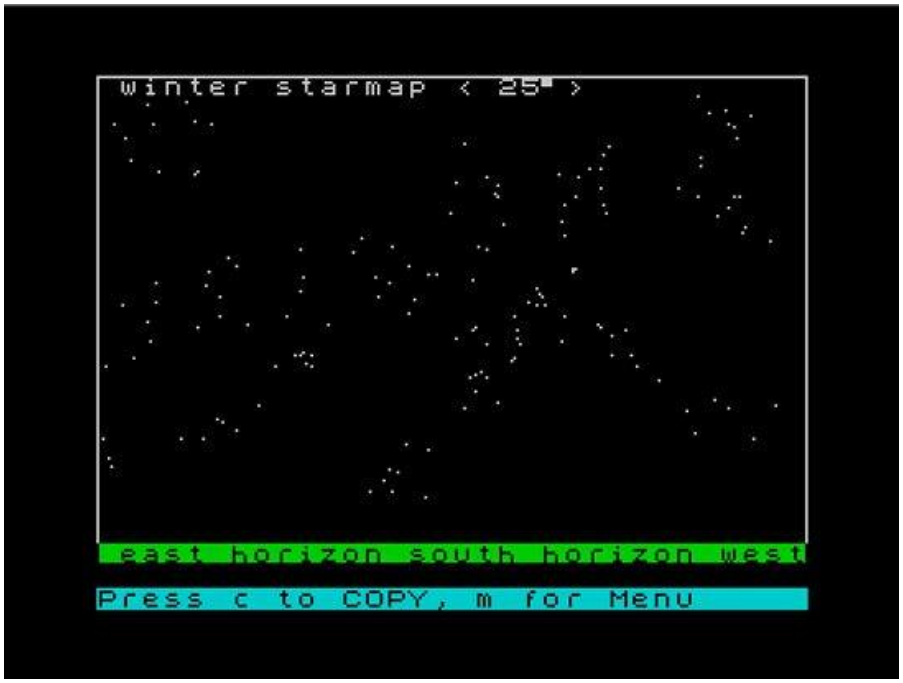
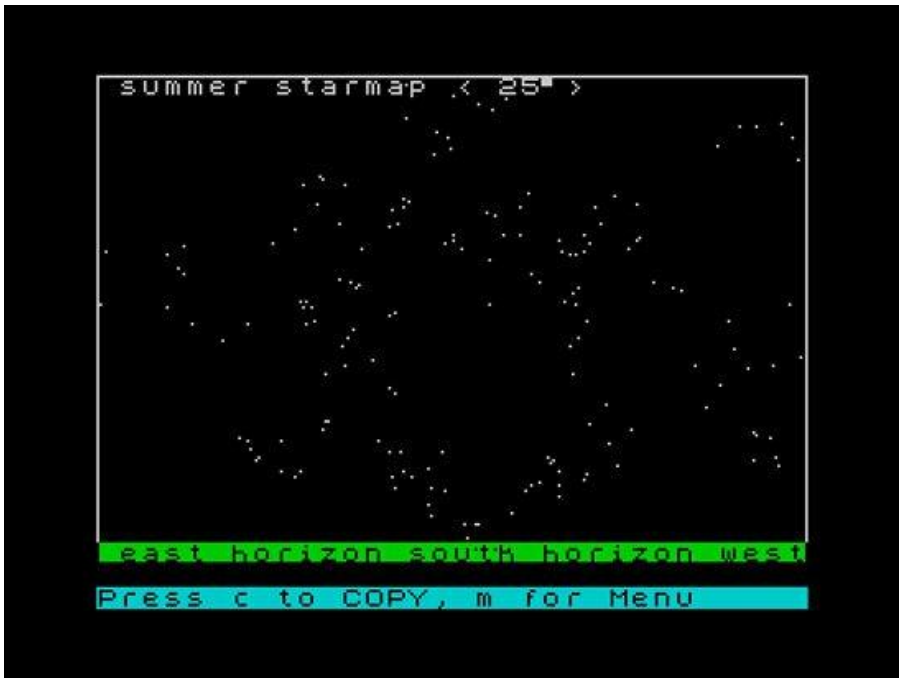


Figure 8.5

Starmap for summer with Aquila centre left and Square of Pegasus extreme left.



Constellation recognition

No attempt has been made to distinguish between stellar magnitudes with different dot sizes but a method of so doing is explained later in this chapter. The limiting stellar magnitude for these starmaps is about magnitude 3½ to 4½. To go any fainter would vastly increase the DATA and clutter the screen. The order in which each star is

PLOTted is intended to aid recognition of that particular constellation. But because the Spectrum only takes about one second to PLOT a typical constellation, this may be too quick for some eyes! To help you, a `slow' button can be pressed to slow the action down.

Starmap projection

Some slight distortion is inevitable in converting the spherical sky into a cylindrical projection for these starmaps, but it is only evident if constellations close to the north or south celestial poles are included (I have deliberately excluded these). The constellations included tend to favour visibility from the latitudes of Europe and North America and extend from Ursa Major (the Plough), Draco and Cassiopeia all centred about +60° declination north to Scorpius about —35° declination south. About half of all the constellations between these two extremes are used in the program. There are 88 recognised constellations in the sky — about two-thirds of these visible from Britain.

Preparing the program

Because the amount of DATA to be used by the full starmaps program is extensive, it has been split into two parts. The first part is called Star Loader, and handles the entry of the x and y coordinate positions of 310 stars, the constellation names and the central coordinates of each starmap so that they can be jigsawed together in the composite starmaps. The second part of the program is called Starmaps and RUNs on the DATA entered in the first part of the program.

Before proceeding with the programs themselves, some explanation of how the DATA is prepared may be enlightening. This will enable you to make modifications to the program if you wish, perhaps adding even more constellations. It involves sketching, in the simplest terms, the constellations to be portrayed. The Spectrum plays no part in this aspect of the work, but a star atlas will be necessary.

Sketching the constellations

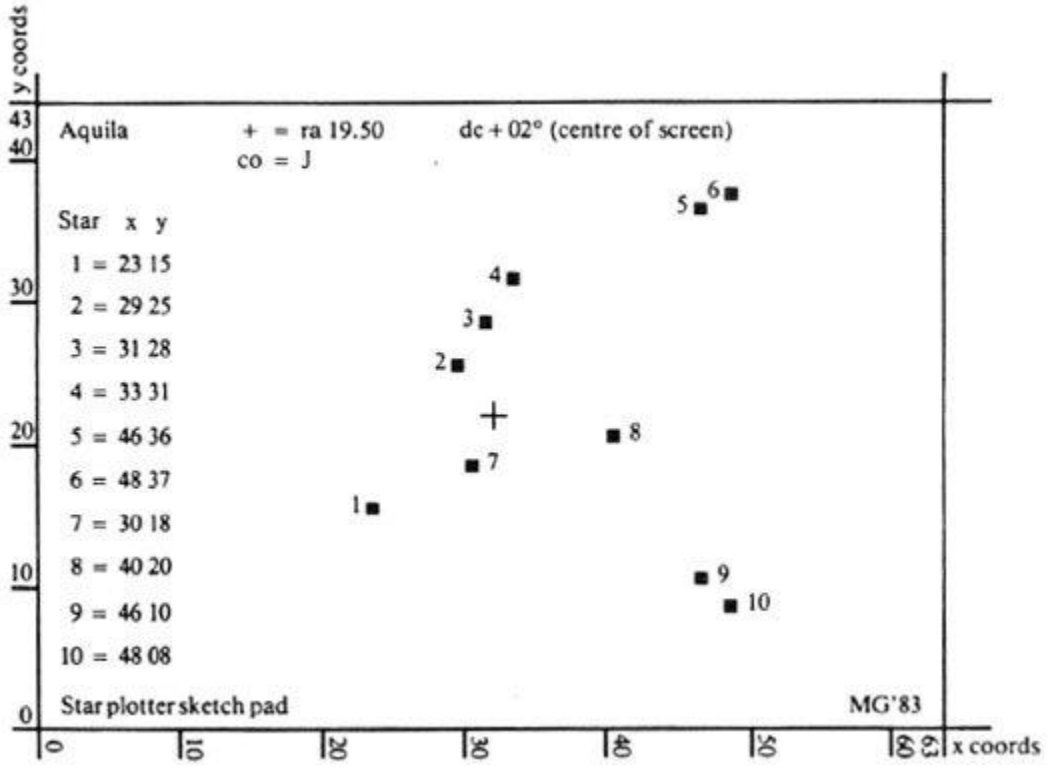
The method of preparing the DATA for each constellation is as follows:

1. Mark on a sheet of graph paper a grid measuring 0 to 63 horizontally (x coordinate) and 0 to 43 vertically (y coordinate).
2. Mark the centre of the rectangle with a '+' and note on the sheet the name and coordinates of the selected constellation.
3. Trace from a star atlas the selected constellation and transfer it to the graph paper by filling the complete square nearest to each star. Check that the grid lines on the star atlas tally with the graph paper without excessive rotation of either.
4. Number each star and list the x and y coordinates from the graph paper as a four figure number, eg 4327.
5. Enter on to the sheet a single letter to CODE the total number of stars used in each constellation — A = 1, B = 2, to Z = 26.

Star DATA

Be sure to note that the x (horizontal) coordinate comes first in item 4. Do not transpose the two halves of this number otherwise that star will be wrongly located on the starmaps. With so many stars to be PLOTted, an incorrectly located star may be difficult to find, especially if the user is unfamiliar with the star patterns of the sky. Even for the experienced astronomer caution is the byword in this initial preparation. Figure 8.6 shows a typical sketch used in this program and executed in the manner described above.

Figure 8.6



Note that the coordinate position taken from the star atlas for the centre of the screen and for each chart prepared is a five figure number, including the decimal point, for the RA (horizontal axis) and a three figure number, including a '+' or '-' prefix, for the Dec (vertical axis). The degree sign is not included in the latter. In the example shown for the constellation of Aquila (the Eagle) the RA of 19.50 indicates 19h 50m and the program in Line 1090 converts this to hours plus decimal hours and finally, in Line 1140, to degrees by multiplication by 15. Thus 19h 50m = 19.8333h = 297.5°.The RA cannot exceed 24.00 because 24.00*15 = 360°, a complete circle.

The Spectrum is of course capable of a screen resolution of 256*176 pixel points whereas the charts have only been prepared to a 64*44 resolution. I have found this to be fully adequate.

The RA are stored in the a$ array and the Dec in the b$ array for use without further computer manipulation in the titles for each large starmap. Also it is more economical on the computer memory to store DATA this way and avoids complex formatting to convert the alternative numeric array into CHR$ for display. In the initial testing of

this program both numeric and CHRS arrays were tried for DATA storage and I noticed that the program ran 10% slower using the CHRS storage. This means that a complete skymap of 310 stars took about 44 seconds to PLOT rather than about 40 seconds, which I consider acceptable, bearing in mind the advantages.

Starmaps 1: Star Loader

Key in the Star Loader program and check that it matches the listing precisely. Before entering all the DATA on the 30 constellations and 310 stars, a few test RUNs are advised to be sure all is well. It should not be necessary to test all 30 constellations, so amend the variable z in Line 2020 to read LET z = 2 and RUN the program. This will allow the first two constellations (Aquila and Capricornus) to be entered from the DATA in Figure 8.7.

The program is well error-trapped and also BEEPs when the keyboard is touched or DATA is entered so that you can concentrate on the task at hand without constant reference to the screen display. A couple of test RUNs, perhaps including few fictitious numbers, should show if all is well. You should note that the program incorporates a SAVE and VERIFY routine for those perhaps new to the Spectrum. This will ensure that all your efforts are not lost by inadvertently pressing the wrong key.

```
2000 REM Star Loader Program
2010 LET z$="": POKE 23609,100
2020 LET z=30
2030 DIM n$(z,11): DIM a$(z,5): DIM b$(z,3): DIM c$(z,17,4)
2040 FOR n=1 TO z
2050 CLS : INPUT "Name of constellation no. ";(n),t$: IF LEN t$>11 THEN GO SUB
2230: GO TO 2050
2060 LET n$(n)=t$: PRINT t$
2070 INPUT "RA=?"; LINE t$: LET e=VAL t$: IF e=25 THEN GO SUB 2230: GO TO 2070
2080 LET a$(n)=t$: PRINT "RA ";t$
2090 INPUT "Dc=?"; LINE t$: LET e=VAL t$: IF ABS e>=90 THEN GO SUB 2230: GO TO 2090
2100 LET b$(n)=t$: PRINT "Dc ";t$
2110 INPUT "co=?"; LINE h$: IF CODE h$-64>17 THEN GO SUB 2230: GO TO 2110
2120 LET z$=z$+h$: PRINT "co ";h$
2130 FOR f=1 TO CODE h$-64
2140 INPUT (f);"=?"; LINE t$
2142 LET e1=VAL t$( TO 2)
2145 LET e2=VAL t$(3 TO )
2150 IF e1<0 OR e263 OR e2>43 THEN GO SUB 2230: GO TO 2140
2160 LET c$(n,f)=t$: PRINT (" " AND f<10);f;"=";t$: BEEP .1,-10
2170 NEXT f: BEEP 1,1
2180 INPUT "Data OK (y/n or (s)top & save)?"'q$
2190 IF q$="n" THEN GO TO 2050
```

```
2195 IF q$="s" THEN GO TO 9990
2200 NEXT n: PRINT #0; FLASH 1;"List complete": GO TO 9990
2230 PRINT #0; FLASH 1;"Error": BEEP .5,30: PAUSE 100: RETURN
9990 SAVE "starmaps" LINE 2050
```

Entering the DATA

Firstly amend the variable z in Line 2020 to read LET z = 30 — this is vital — and RUN the program. Work progressively through the DATA in Figure 8.7. All 30 constellations do not have to be entered in one session but can be spread over two or three sessions at the computer if so desired.

The option to SAVE the program, with the DATA just entered, occurs on Line 2195 when key's' is pressed. Following a successful SAVE, the program then invites you to rewind and play the tape to VERIFY. A BEEP occurs on successful SAVE and VERIFY. If it does not, then enter as a direct command GOTO 9990 and repeat the SAVE and VERIFY routines until the 'OK' message appears on the screen. To LOAD the program from tape for the next session of DATA input, enter as a direct command LOAD "starmaps" and play the tape. On successful LOADing, the program automatically goes to Line 2050 ready for the next constellation to be entered.

If for any reason the program crashes, enter as a direct command GOTO 2050. On no account should you press the RUN key or give a GOTO command to a line number less than Line 2050. If this is disregarded, all the DATA not SAVEd on tape will be lost or overwritten. When all the DATA has been entered and finally SAVEd and VERIFYed, the next part of the program can be dealt with.

Starmaps 2

Key in the second part of the program — called Starmaps — and amend Line 9990 to read:

9990 SAVE "starmaps" LINE 1

This will ensure that, when the program is SAVEd and subsequently LOADed, it automatically starts by going to Line 1. The new portion of the program handles the display, selection and PLOTting of the starmaps. When you've checked that the listing is satisfactory, enter as a direct command GOTO 1 to start the program. On no account must the RUN key be pressed to labour the point.

The program

The REM statements show the general structure of the program. The actual PLOTting of the starmaps is split into two parts:

1. The selected constellations to a large scale.
2. The seasonal and annual starmaps.

———

1 Aquila

RA19.50

Dc+02

Co = J

1 = 2315

2 = 2925

3 = 3128

4 = 3331

5 = 4636

6 = 4837

7 = 3018

8 = 4.020

g = 4510

10 = 4808

———

2 Capricornus

RA21.12

Dc-18

Co = L

1 = 5128

2 = 5228

3 = 5025

4 = 4210

s = 4008

6 = 3510

7 = 2614

8 = 2715

9 = 2418

10 = 2321

11 = 2022

12 = 3521

———

3 Lyra

RA19.12

Dc+36

Co = F

1 = 4126

2 = 3924

3 = 3927

4 = 3523

5 = 3417

6 = 3718

———

4 Scorpius

RA17.10

Dc-34

Co = Q

1 = 5039

2 = 5137

3 = 5238

4 = 5434

5 = 5429

6 = 5425

7 = 4730

8 = 4429

9 = 4227

10 = 3719

11 = 3605

12 = 3106

13 = 2405

14 = 2110

15 = 2015

16 = 2415

17 = 2515

———

5 Delphinus

RA20.54

Dc+13

Co = F

1 = 3524

2 = 3724

3 = 3622

4 = 3922

5 = 4017

5 = 3716

———

6 Cygnus

RA20.46

Dc+35

Co = I

1 = 3433

2 = 2312

3 = 3117

4 = 3926

5 = 4933

6 = 4719

7 = 5610

8 = 4135

9 = 4036

———

7 Sagitta

RA20.12

Dc+19

Co = D

1 = 3623

2 = 4022

3 = 4220

4 = 4321

———

8 Hercules

RA17.30

DC +28

Co = N

1 = 3601

2 = 3617

3 = 4126

4 = 4726

5 = 3833

6 = 4736

7 = 5041

8 = 5112

9 = 5409

10 = 3534

11 = 2621

12 = 2324

13 = 2326

14 = 2023

———

9 Sagittarius

RR18.50

Dc-26

Co = M

1 = 2233

2 = 2629

3 = 2720

4 = 3029

5 = 3122

6 = 2816

7 = 3420

8 = 4023

9 = 4529

10 = 4615

11 = 4116

12 = 4010

13 = 4106

———

10 Perseus

RA03.S5

DC+40

Co = H

1 = 3108

2 = 3013

3 = 3119

4 = 3522

5 = 3629

6 = 4032

7 = 4432

8 = 4537

9 = 4740

10 = 4425

11 = 4519

12 = 4616

13 = 2930

———

11 Pegasus

RA23.05

Dc+15

Co = J

1 = 0819

2 = 1038

3 = 3237

4 = 3216

5 = 4112

6 = 5206

7 = 6112

8 = 3840

g = 3830

10 = 3532

———

12 Auriga

RA05.55

Dc+40

CD = I

1 = 4004

2 = 3016

3 = 3227

4 = 3541

5 = 4329

6 = 4726

7 = 4722

8 = 4623

9 = 4912

———

13 Bootes

RA15.05

Dc+29

Co = J

1 = 5704

2 = 5405

3 = 4707

4 = 3819

5 = 2828

6 = 2634

7 = 3338

8 = 4135

9 = 4122

10 = 4223

———

14 Taurus

RR04.45

Dc+15

Co = D

1 = 1429

2 = 1940

3 = 3421

4 = 3720

5 = 4020

6 = 3923

7 = 3824

6 = 3726

9 = 5032

10 = 5055

11 = 5133

12 = 4716

13 = 5913

14 = 6012

15 = 4607

———

15 Corona Bor

RA16.15

Dc+28

Co = G

1 = 4526

2 = 4723

3 = 4520

4 = 4219

5 = 4019

6 = 3720

7 = 3624

———

16 Draco

RA18.30

Dc+60

Co = N
1 = 3607
2 = 3715
3 = 4113
4 = 4209
5 = 2620
6 = 2430
7 = 2532
8 = 3435
9 = 3437
10 = 3835
11 = 4328
12 = 5225
13 = 5723
14 = 5227
———
17 Orion
RA05.55
RA+01
Co = Q
1 = 3229
2 = 3832
3 = 3933
4 = 4327
5 = 3715
6 = 3916
7 = 4117
8 = 4315
9 = 4706
10 = 3504
11 = 5221
12 = 5322
13 = 5429
14 = 3910
15 = 5527
16 = 5432
17 = 5337
———
18 Cassiopeia
RA01.20
Dc+59
Co = G
1 = 2729
2 = 3123
3 = 3724
4 = 3819
5 = 4018
6 = 4114
7 = 4622
———
14 Leo
RR11.00
Dc+15
Co = M
1 = 5534
2 = 5237
3 = 4532
4 = 4427
5 = 4923
5 = 4916
7 = 5913
8 = 4112
9 = 2628
10 = 2621
11 = 2314
12 = 2407
13 = 1420
———
20 Canis Major
RA07.10
Dc-24
Co = H
1 = 2512
2 = 3116
3 = 3412
4 = 4610
S = 4727
6 = 3929
7 = 3619
8 = 3320
———
21 Ursa Major
RR11.50
DC+54
Co = N
1 = 2727
2 = 0419
3 = 3122
4 = 3934
5 = 1826
6 = 4127
7 = 1226
8 = 5335
9 = 5628
10 = 6227
11 = 5710
12 = 5609
13 = 4310
14 = 3314
———
22 Corvus
RAl2.40
Dc-17
Co = F
1 = 3422
2 = 3521
3 = 3412
4 = 4213
5 = 4310
6 = 4020
———
23 Gemini
RA07.40
Dc+22
Co = L
1 = 3029
2 = 3334

3 = 3029
4 = 3820
5 = 3913
6 = 4318
7 = 5212
8 = 5007
9 = 5721
10 = 6021
11 = 5024
12 = 4431
———
24 Hydra
RR09.55
Dc-10
Co = M
1 = 6040
2 = 6036
3 = 5837
4 = 5640
5 = 5741
6 = 5440
7 = 4736
8 = 4122
9 = 3313
10 = 2816
11 = 2617
12 = 2110
13 = 1310
———
25 Virgo
RA12.52
Dc+00
Co = H
1 = 1721
2 = 2106
3 = 2614
4 = 2937
5 = 3127
6 = 3620
7 = 4521
8 = 5524
———
26 Andromeda
RA01.25
Dc+38
Co = K
1 = 5509
2 = 4512
3 = 3618
5 = 2228
6 = 4614
7 = 4021
8 = 4225
9 = 4425
10 = 3036
11 = 3039
———
27 Cancer
RA08.50
DC+17
Co = E
1 = 3437
2 = 3527
3 = 3422
4 = 2913
5 = 4410
———
28 Libra
RA15.32
Dc-19
Co = G
1 = 3921
2 = 4726
3 = 3635
4 = 3228
5 = 4213
6 = 3110
7 = 3008
———
29 Serpens
RA16.00
Dc+07
Co = H
1 = 3836
2 = 3636
3 = 3333
4 = 3732
5 = 4126
6 = 3820
7 = 3517
8 = 3607
———
30 Cetus
RA01.42
Dc-04
Co = L
1 = 0331
2 = 0338
3 = 0840
4 = 1031
5 = 1227
6 = 2022
7 = 3011
8 = 3303
9 = 4015
10 = 4512
11 = 5401
12 = 6213

In selecting a constellation (INPUT 1 to 30 inclusive) the variable in Line 170 (LET sc = 4) effectively enlarges the scale by a factor of 4 so that the constellation fills the screen. Options with an INPUT of 31 to 34 inclusive have an additional variable called x1 which has different values via the conditional GOTO in Line 120. Variable x1 selects one quarter of the complete skymap of 30 constellations for presentation. The skymap — INPUT option 35 — has a complete wrap-around effect horizontally through 360°. The variable sc is set to 1 for the seasonal starmaps and 0.52 for the annual skymap. Watch in Line 90 that you enter correctly 'm>1' and 'm<l' — the second is lower case 'L'.

```
10 REM Starmaps
20 BORDER 0: PAPER 0: INK 9: CLS : DIM e$(32): LET g=100/60
30 PRINT PAPER 5;" Constellations & Starmaps "
40 LET t$=" starmap"
50 LET j=15: LET l=35
70 FOR n=1 TO z: PRINT AT n-(j AND n>j),0+(j AND n>j);(" " AND n<10);n;" ";n$(n):
NEXT n
80 PRINT PAPER 1;AT 16,j;"31 spring";t$;AT 17,j;"32 summer";t$;AT 18,j;"33
autumn";t$;AT 19,j;"34 winter";t$;AT 20,j; PAPER 4;"35 skymap"
90 INPUT INK 6;"Select by no. ";m: IF ml THEN GO TO 90
100 PRINT #0; INK 5;"Plot fast or slow (f/s)?": PAUSE 0: CLS : LET pa=1: IF
INKEY$="s" THEN LET pa=50
120 IF m>30 THEN GO TO m*10
140 PRINT " ";n$(m);" +=RA";a$(m);" Dc";b$(m);CHR$ 130;AT 11,15; INK 4;"+"
150 PRINT INK 3;AT 1,13;"north";AT 11,1;"east";AT 11,27;"west";AT 20,13;"south": GO
SUB 1210
160 PRINT INK 4;AT 20,1;""
170 LET sc=4
175 FOR n=1 TO CODE z$(m)-64
180 LET dx=VAL c$(m,n, TO 2)
185 LET dy=VAL c$(m,n,3 TO )
190 PLOT dx*sc,dy*sc
195 DRAW BRIGHT 1;1,1: DRAW 1,-1: DRAW -1,-1
200 PAUSE pa: NEXT n
220 PRINT #0; PAPER 5; INK 9;"Press c to COPY, m for Menu "
225 BEEP .6,40: BEEP .1,30
230 PAUSE 0: IF INKEY$="c" THEN COPY : INPUT "": GO TO 220
240 GO TO 20
310 PRINT " spring";t$,: LET x1=-3: GO TO 1000
320 PRINT " summer";t$,: LET x1=5: GO TO 1000
```

```
330 PRINT " autumn";t$,: LET x1=13: GO TO 1000
340 PRINT " winter";t$,: LET x1=-11: GO TO 1000
350 PRINT " Annual skymap:due south at 9pm": LET hi=150: LET sc=.52: LET x1=5: GO
TO 1030
1010 LET sc=1: LET hi=50
1020 PRINT "": PAPER 4;AT 21,0;" east horizon south horizon west": GO TO 1060
1030 FOR n=17 TO 21: PRINT PAPER n-16;AT n,0;e$: NEXT n
1040 PRINT AT 16,0;" autumn summer spring winter "
1050 FOR n=1 TO 3: PLOT n*64,40: DRAW 0,10: NEXT n
1060 GO SUB 1210: FOR f=1 TO z
1070 PRINT #0;n$(f)
1080 LET zx=VAL a$(f)
1090 LET x=zx+(zx-INT zx)/g
1100 LET y=VAL b$(f)
1110 FOR n=1 TO CODE z$(f)-64
1120 LET dx=VAL c$(f,n, TO 2)
1130 LET dy=VAL c$(f,n,3 TO )
1140 LET xx=(x1*15-x*20+380+dx)*sc
1150 IF x1=13 AND xx>480 THEN LET xx=xx-480
1160 LET yy=(y*1.5+hi+dy)*sc
1170 IF xx255 OR yy175 THEN GO TO 1190
1180 PLOT xx,yy
1190 NEXT n: PAUSE pa: INPUT "": NEXT f: GO TO 220
1210 PLOT 0,0: DRAW 0,175: DRAW 255,0: DRAW 0,-175: DRAW -255,0: RETURN
2000 REM Star Loader Program
2010 LET z$="": POKE 23609,100
2020 LET z=30
2030 DIM n$(z,11): DIM a$(z,5): DIM b$(z,3): DIM c$(z,17,4)
2040 FOR n=1 TO z
2050 CLS : INPUT "Name of constellation no. ";(n),t$: IF LEN t$>11 THEN GO SUB
2230: GO TO 2050
2060 LET n$(n)=t$: PRINT t$
2070 INPUT "RA=?"; LINE t$: LET e=VAL t$: IF e=25 THEN GO SUB 2230: GO TO 2070
2080 LET a$(n)=t$: PRINT "RA ";t$
2090 INPUT "Dc=?"; LINE t$: LET e=VAL t$: IF ABS e>=90 THEN GO SUB 2230: GO TO 2090
2100 LET b$(n)=t$: PRINT "Dc ";t$
2110 INPUT "co=?"; LINE h$: IF CODE h$-64>17 THEN GO SUB 2230: GO TO 2110
2120 LET z$=z$+h$: PRINT "co ";h$
2130 FOR f=1 TO CODE h$-64
```

```
2140 INPUT (f);"=?"; LINE t$
2142 LET e1=VAL t$( TO 2)
2145 LET e2=VAL t$(3 TO )
2150 IF e1<0 OR e2<0 OR e1>63 OR e2>43 THEN GO SUB 2230: GO TO 2140
2160 LET c$(n,f)=t$: PRINT (" " AND f<10);f;"=";t$: BEEP .1,-10
2170 NEXT f: BEEP 1,1
2180 INPUT "Data OK (y/n or (s)top & save)?"'q$
2190 IF q$="n" THEN GO TO 2050
2195 IF q$="s" THEN GO TO 9990
2200 NEXT n: PRINT #0; FLASH 1;"List complete": GO TO 9990
2230 PRINT #0; FLASH 1;"Error": BEEP .5,30: PAUSE 100: RETURN
9990 SAVE "starmaps" LINE 1
```

Constellation Plot

The Starmaps program demonstrated a technique for entering star patterns directly into DIMensioned arrays via INPUT. This method is advisable if many constellations and stars are to be included. The Startrax program later in this chapter (using two constellations with few stars to PLOT), holds the star positions in a CHR$ as numerics for slicing.

This short program shows two additional methods for PLOTting star patterns using DATA and in CODE form in CHR$. In both these methods the information appears in the program listing and is easy to manipulate. See Figures 8.8 and 8.9.

Figure 8.8
Lyra PLOTted via DATA and READ

Figure 8.9

Lyra PLOTted via CODEd CHR$ (shown in display)

Method 1: DATA store

This is the conventional way to store DATA and is described adequately, if briefly, in the Spectrum Manual. DATA can only be called into use by the READ command and, once READ, can only be re-used with the RESTORE command. In this particular program the DATA is READ and immediately PLOTted and so the RESTORE command is not needed.

However, when the program STOPs (with the error message ‘integer out of range’ — this will be explained later) try restarting the program with GOTO 1. The program will not restart — an error message ‘out of data’ is displayed because the DATA was not RESTOREd. Enter RUN, and the program will do just that as this command automatically RESTORES the DATA. Thus the DATA/READ/RESTORE sequence must be given some thought especially for long programs where various DATA may be used in different orders.

In the Constellation Plot program, the DATA is in Line 40 and represents the summer star pattern of Lyra (the Harp) with six stars, each with an x and y coordinate position. This means 12 items of DATA. In addition the corners of the display are also PLOTted to indicate the scale. This is the sequence of DATA that appears as:

0, 0, 63, 43, 0, 43, 63, 0

This makes a total of 20 items of DATA. The final, 21st, item of DATA is the number of stars and corner positions to be PLOTted and this is held as the first figure in the DATA sequence (ie 10).

Line 90 READs the first item of DATA (10) and the variable c sets the length of the FOR/NEXT n loop in Line 100 for all the PLOTting. Line 100 also READs the x and y coordinates from the DATA and these are PLOTted in Line 110, each coordinate value multiplied by 4, the value of the variable called ‘scale’ (which I will explain later).

Method 2: CHR$ CODE store

This method is in many ways more elegant than a DATA/READ /RESTORE sequence and relies on the sophisticated string slicing ability of the Spectrum. It does however need care in the selection of a suitable range of CHR$ from the Spectrum CHR$ set and some confusion can occur in published listings as to which CHR$ is intended. You may need to refer to the Manual to be certain.

The ASCII character set

The Spectrum uses the ASCII character set, and any CODE between CODE 32 (blank space) and CODE 127 (copyright sign) can be used conveniently. The range of CHR$ could be extended to include the ‘chunky graphic’ set from CODE 128 to CODE 143. The ‘chunky graphic’ set, the copyright sign and CODE 96 (the ‘£’ sign) are not actually part of the ASCII set but can still be used in Spectrum programs in CODE form and printed by the ZX printer — although not necessarily by other printers.

In a star PLOTting program with a screen resolution limited to 64 horizontally and 44 vertically (a quarter of the Spectrum’s potential, as mentioned in the section on Starmaps) a maximum of 64 CODEs is sufficient for the requirements. CODE 34 — the quotation sign (“) — cannot be used within a string array so it is best to assume CODE 35 — the hash sign (#) — is the minimum useful CODE. It does however make life much easier if the upper and lower case letter CODEs are used — a total of 52 CODEs. As these can form the bulk of the string, they prove much easier to identify readily and to accept the balance of required CODEs both from between the upper and lower case letter set — CODE 91 to CODE 96 — and from above the lower case letter CODEs to CODE 127 as the ASCII hieroglyphics.

The selected CODEs which I usually use and which are demonstrated in the program are from CODE 64 (@) to CODE 127 (©) inclusive. These are converted into numeric x and y values in Lines 270 and 280 thus:

LET x = CODE a$(n) – 64
LET y = CODE a$(n+ 1) – 64

By deducting 64 from each CODE, a range of values from 0 to 64 is possible:

CHR$ @ = CODE 64 – 64 = 0 to
CHR$ @ = CODE 127 – 64 = 63

This represents the horizontal or x coordinate position. The range of CODEs for the vertical or y coordinates would be:

CHR$@ = CODE 64 – 64 = O to
CHR$k = CODE 107 – 64 = 43

From Line 200 the program demonstrates the PLOTting of the Lyra star pattern using the CODEs contained in the characters of the a$. The a$ contains 21 characters encoding precisely the same information as in the DATA of Line 40 used exclusively in the first part of the program. The only difference is that the first character of the a$, ie 'T', has an effective value of 20:

CHR$ T = CODE 84 – 64 = 20

and this is used in setting the length of the FOR/NEXT n loop in Line 260:

FOR n = 2 TO CODE a$(l) – 64 STEP 2
where n has a value sequentially of:

2, 4, 6, 8, 10, 12, 14, 16, 18, 20

via STEP 2 for the x coordinate value in the a$. Thus the y coordinate value is extracted by the odd numbers of the loop:

3, 5, 7, 9, 11, 13, 15, 17, 19,21

in the form… CODE a$(n+l) – 64.

Scaling the screen image

It is often useful to change the scale of the star pattern on the screen and the variable called 'scale' is used to do this. In the first part of the program using DATA, and before PLOTting the star positions, this variable is set to a value of 4 so enlarging the separation of the star images to fill the screen, particularly the corner markers. In this

second part of the program the scale is initially set to a value of 3 and this is amended, once the FOR /NEXT n loop is completed, in Line 310:

LET scale = scale – 0.5: GOTO 230

The GOTO 230 sends the program around the FOR/NEXT n loop again and again until the program crashes with the error message ‘integer out of range’, because the scale of the image is now too large to be PLOTted on the screen.

But before it crashes the Lyra star pattern is seen to shrink in size, in progressive steps, to a single pixel in the bottom left corner of the screen before enlarging again. The current value of the scale variable is displayed at the top right of the screen and it will be noted that constantly deducting 0.5 from this variable (initially set to a value of 3) will, after six repeats of the n loop, go to negative values and increase the image size again. Thus the scale variable can have negative values as well as positive values and the PLOTted results appear similar.

Preparing starcharts

For an explanation of preparing the starcharts before transfer to a Spectrum program, refer to the Starmaps program earlier in this chapter. Let it suffice here to say that, if it is intended to use characters to CODE the star positions, then the graph paper should be marked on the x and y axes with the CHR$ rather than numeric values. Use the selection of CHR$ with CODE 64 to CODE 127 for the x axis and CODE 64 to CODE 107 for the y axis as explained above.

More constellations

This program only demonstrates how to PLOT the star pattern of Lyra — one constellation. If more constellations are to be PLOTted then here is a simple way to handle and select the DATA without entering the information into DIMensional arrays. As it is much easier to select a given constellation via CHR$ in string arrays, the explanation will be confined to this method.

The CHR$ CODEs used in Method 2 above are combined with the conditional selection, used in the Startrax program later in this chapter, from a series of constellations all similarly named — in this case a$. Thus after each constellation has been sketched on to graph paper and converted into a string of CHR$ CODEs, these are entered into the program between Line 200 to 210 as an a$. Each a$ must be followed by a conditional statement so that the program may look something like this:

```
201 LET a$ = “T……..”: IF b$(1) = “A” THEN GOTO 220
202 LET a$ = “L……”: IF b$(1) = “D” THEN GOTO 220
203 LET a$ = “R……..”: IF b$(1) = “C” THEN GOTO 220
204 LET a$ = “R……..”: IF b$(2) = “u” THEN GOTO 220
```

The constellations supposedly represented are, in order, (A)quila (10 stars), (D)elphinus (6 stars), (C)ygnus (9 stars) and A(u)riga (9 stars). Note how the conditional GOTO in Line 204 selects the second letter, ie b$(2), as

lower case V in Auriga to distinguish it from b$(l) in Aquila in Line 201. The first letter in each a$ CODEs the number of characters immediately following it, which are represented by full stops. The only program amendments are the deletion of Lines 20 to 130 inclusive and Line 310 and the addition of Line 150 (INPUT b$) to select your constellation to PLOT.

```
40 DATA 10, 0, 0, 63, 43, 0, 43, 63, 0, 41, 26, 39, 24, 39, 27, 35, 23, 34, 17, 37,
18
50 BORDER 0: INK 7: PAPER 1: CLS
60 LET scale=4
70 PRINT "Lyra","scale=";scale
80 PRINT "via DATA/READ"
90 READ c
100 FOR n=1 TO c: READ x,y
110 PLOT x*scale,y*scale
120 NEXT n
130 PAUSE 100
210 LET a$="T@@@k@k@@iZgXg[cWbQeR"
220 LET scale=3
230 PAUSE 100: PAPER 2: CLS
240 PRINT "Lyra","scale=";scale
250 PRINT "via a$ codes"
260 FOR n=2 TO CODE a$(1)-64 STEP 2
270 LET x=CODE a$(n)-64
280 LET y=CODE a$(n+1)-64
290 PLOT x*scale,y*scale
300 NEXT n
310 LET scale=scale-.5: GO TO 230
```

Stellar Magnitudes

Stars that we see in the sky vary enormously in brightness — by a factor of over 100: from brightest to faintest. The programs in this book generally omit such differences for the sake of simplicity, and use a single pixel to PLOT each star on, for example, the Starmaps program. There are two basic ways to distinguish these stellar differences on the Spectrum screen and the three routines which follow can easily be incorporated into programs.

The first method relies on PLOTting additional pixels to form a larger image: the second uses progressively darker shades of INK to PLOT the pixel. Obviously the latter is only suitable for a monochrome presentation and the star positions must be sufficiently well separated so as not to occur in the same character square (as this will only

support a single INK colour.). Nevertheless the second method has the merit of realism (in monochrome) in that the star images remain small. The third method combines the first two — again this is only suitable for monochrome but has the maximum potential of brightness range.

1: Extra pixel plotting

Key in and RUN the Star Magnitudes 1 routine in Figure 8.10, and the screen is filled with star-like images that get progressively smaller down the screen as a regular grid. Each star is PLOTted initially only as a single pixel but is enlarged by the conditional DRAW routines that immediately follow. This takes the form of a FOR/NEXT n loop and stars that are PLOTted first (with the lower n values in their PLOT position) are conditionally dealt with as:

IF n<3 THEN DRAW 0,1
IF n<6 THEN DRAW 1,0
IF n<9 THEN DRAW 0,-1

Figure 8.10

```
3000 PRINT "Star Magnitudes 1 @"
3030 BORDER 0: PAPER 1: INK 9
3040 LET z=0
3050 FOR n=1 TO 12
3060 PLOT z,n*10-150
3070 IF n<3 THEN DRAW 0,1
3080 IF n<6 THEN DRAW 1,0
3090 IF n<9 THEN DRAW 0,-1
3100 NEXT n
3200 LET z=z+20: GO TO 3050
```

Figure 8.11 shows precisely how each image is PLOTted and then DRAWn. Only one extra pixel is added by each conditional line — the value 1 in each DRAW statement.

Figure 8.11

Using conditional DRAW statements to enlarge a single pixel into a brighter star. Shown here in four steps. Because the individual pixels are not easily seen at normal viewing distances from the TV, the image just appears enlarged.

PLOT/ DRAW

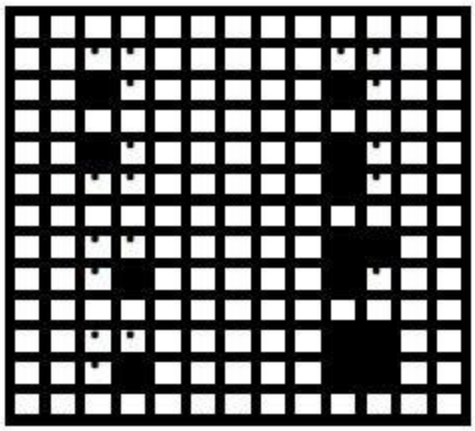
SCREEN RESULT

PLOT (single pixel)

IF n<3 THEN DRAW 0,1

IF n<6 THEN DRAW 1,0

IF n<9 THEN DRAW 0,-1



2: Colour tone (in monochrome) plus extra pixel plotting

Amend Star Magnitudes 1 to Star Magnitudes 2 (Figure 8.12) and RUN it. If a colour set is used for the display then the colour control should be adjusted to a monochrome picture. This then shows how the INK tone considerably extends the apparent range of star brightness, particularly for fainter stars. The limitations of such a scheme are mentioned earlier in this section.

Figure 8.12

```
3000 PRINT "Star Magnitudes 2 @"
3030 BORDER 0: PAPER 1: INK 9
3040 LET z=0
3050 FOR n=1 TO 12
3051 INK 9-n/2
3060 PLOT z,n*10-150
3070 IF n<3 THEN DRAW 0,1
3080 IF n<6 THEN DRAW 1,0
3090 IF n<9 THEN DRAW 0,-1
3100 NEXT n
3200 LET z=z+20: GO TO 3050
```

3: Colour tone (in monochrome)

The third version (Figure 8.13) is prepared by deleting all the conditional DRAW lines in the second routine before it is RUN. A practical version of this routine, for incorporating into a longer program (with the limitations already mentioned), would take the form:

PAPER 0: INK 9: REM dark tone most stars = INK 2
FOR n = 1 TO 12
IF n5 THEN PLOT; INK 2; x(n), y(n): REM faint stars
NEXT n

The relative brightness of the stars are controlled by the expression

INK 8-n

for the first five stars (n5) would have an INK tone value of 2 (INK pixel colour red). It is possible to extend the range to include an extremely faint star by using an INK tone value 1 (INK pixel colour blue) but on a black sky (PAPER 0) a single pixel may be undetectable.

Figure 8.13

```
3000 PRINT "Star Magnitudes 3 @"
3030 BORDER 0: PAPER 1: INK 9
3040 LET z=0
3045 BRIGHT 1
3050 FOR n=1 TO 12
3051 INK 9-n/1.5
3060 PLOT z,n*10-150
3100 NEXT n
3200 LET z=z+20: GO TO 3050
```

Starcharts with magnitude control

To incorporate any of the above routines into a longer program, you will need to take care in the order in which you PLOT each star. The brightest stars should be listed first and so on in decreasing order of brightness. If a particular constellation has no bright stars it is necessary to repeatedly PLOT the same star before PLOTting the next star, until the correct INK tone is used. This means that more apparent star positions will be listed than finally appear on the screen.

To satisfy the above condition with the first method (extra pixel plotting) then the PLOTting may have to include more conditional statements so that a particular constellation skips some of the extra pixels that otherwise would be DRAWn. This could take the form:

FOR n = 1 TO 30: REM 30 constellations
FOR f = 1 TO (no of star positions) STEP 2
PLOT x(n,f), y(n,f+1): REM x,y = star coordinates
IF n=5 AND f=1 THEN GOTO XXXX : REM skip DRAW routine for constellation no 5, star 1
IF f<3 THEN DRAW 0,1

IF f…
NEXT f: NEXT n (NB line XXXX)

Star Graphics

This short demonstration program shows how more complex shapes can be constructed for incorporating into starmaps. These images would represent the brightest stars to be displayed and as such should have geometric forms.

The program

The program is divided into three parts as indicated by the REM statements. The first part to Line 7 copes with the selection of images numbered 1 to 4 via the INKEYS command in Line 3. Line 8 contains two GOSUB commands — the first is conditional on the selection:

GOSUB 100 or 200 or 300 or 400

in the form GOSUB a x 100 where a equals the VALUE of INKEY$ from Line 4. At the appropriate GOSUB line, the selected star form is PLOTted and DRAWn according to its shape. The program then RETURNS to Line 8 for the second GOSUB (GOSUB 1000).

The third and final section of the program from Line 1000 uses the POINT command to scan the star image just PLOTted and enlarge it for easy viewing. Two FOR/NEXT loops are used (n and f) to cover the tiny area of the screen adjacent to the x and y coordinate positions of the small star image. Lines 1010 and 1020 take the form:

1010 IF POINT (x + n, y + f) = 1 THEN PRINT … “CHR$143″
1020 IF POINT (x + n, y + f) = 0 THEN PRINT … “CHR$58″

If POINT equals a value of 1 then PRINT a solid INK square character (ie a pixel is PLOTted at this location). Conversely, if POINT equals a value of 0 then PRINT a colon (ie a pixel is not PLOTted at this location).

Line 1020 is not really necessary to define the star shape but it does help to locate its relative position by marking the unPLOTted points. The program then RETURNS for the next selection.

Further star shapes can be included in the program, starting at Line 600 in steps of one hundred (700, 800, 900, and so on). This is to satisfy the conditional GOSUB in Line 8. The value 4 currently set in Line 3 and Line 5 should be amended for each additional shape included in the program. The final statement in each new PLOT and DRAW section must conclude with RETURN.

It is useful to sketch the new star shapes on squared paper before committing them to the program but it is not essential. You can simply start with a PLOT x,y condition and then add DRAW commands (preferably with quite

small values — certainly no greater than a value of 6) and RUN the program to test the new design. If it is unsatisfactory then change it. The aim should be to contain all the new design within the portion of the screen scanned by the POINT command and displayed in enlarged form.

Incorporating results in a starmap program

A subroutine virtually identical to this program could be incorporated in a starmap program to represent some of the brighter stars. Only Lines 100 to 499 (plus any extra lines for your own designs) are required in such a subroutine together with some control lines to make the subroutine work. These would be in the main program area dealing with PLOTting the stars and could take the form:

```
star PLOTting routine where x and y are star coordinates.
IF n = 1 THEN GOSUB 400 : REM mag 0 star = brightest
IF n = 2 THEN GOSUB 100 : REM mag 1 star
IF n = 3 THEN GOSUB 300 : REM mag 2 star
IF n = 4 THEN GOSUB 200 : REM mag 3 star
If n > 4 THEN PLOT x,y: REM mag 4 are PLOTted as a single pixel and represent the faintest stars.
```

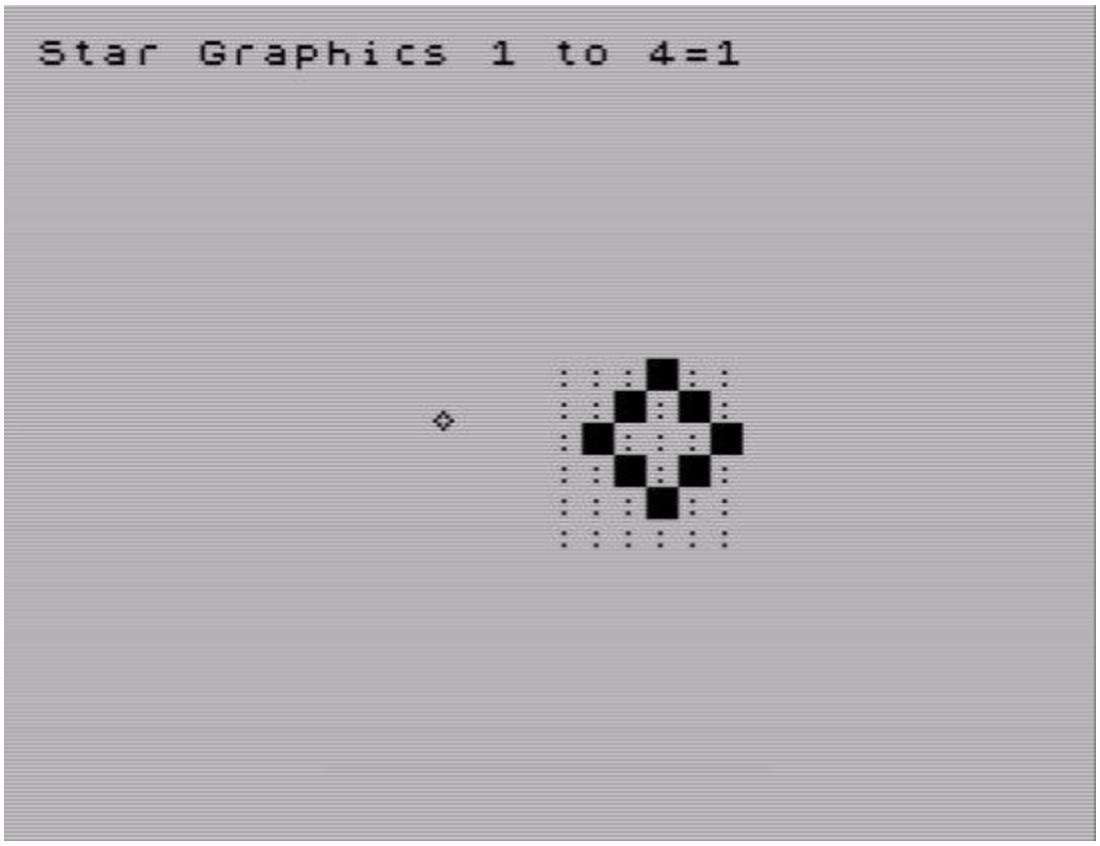
These conditional sample lines shown above are only one example of what could be used in the program. The permutations are endless. Suppose for example that the first three stars are to be magnitude 0 (the brightest) in each constellation of the starmap and the next two stars of magnitude 1, then the routine would read:

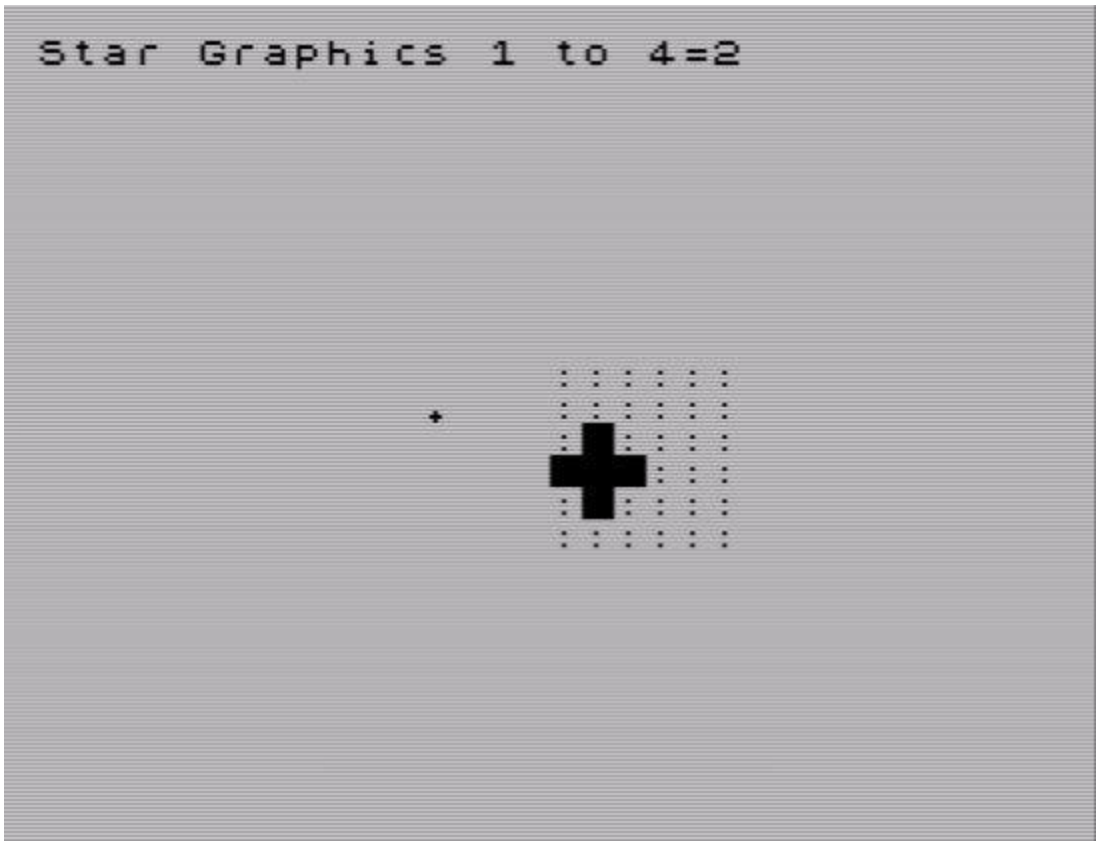
```
IF n<4 THEN GOSUB 400: NEXT n: REM mag 0 stars 1 to 3
IF n<6 THEN GOSUB 100: NEXT n: REM mag 1 stars 4 and 5
If n=6 THEN GOSUB 300: REM mag 2 star 6
```

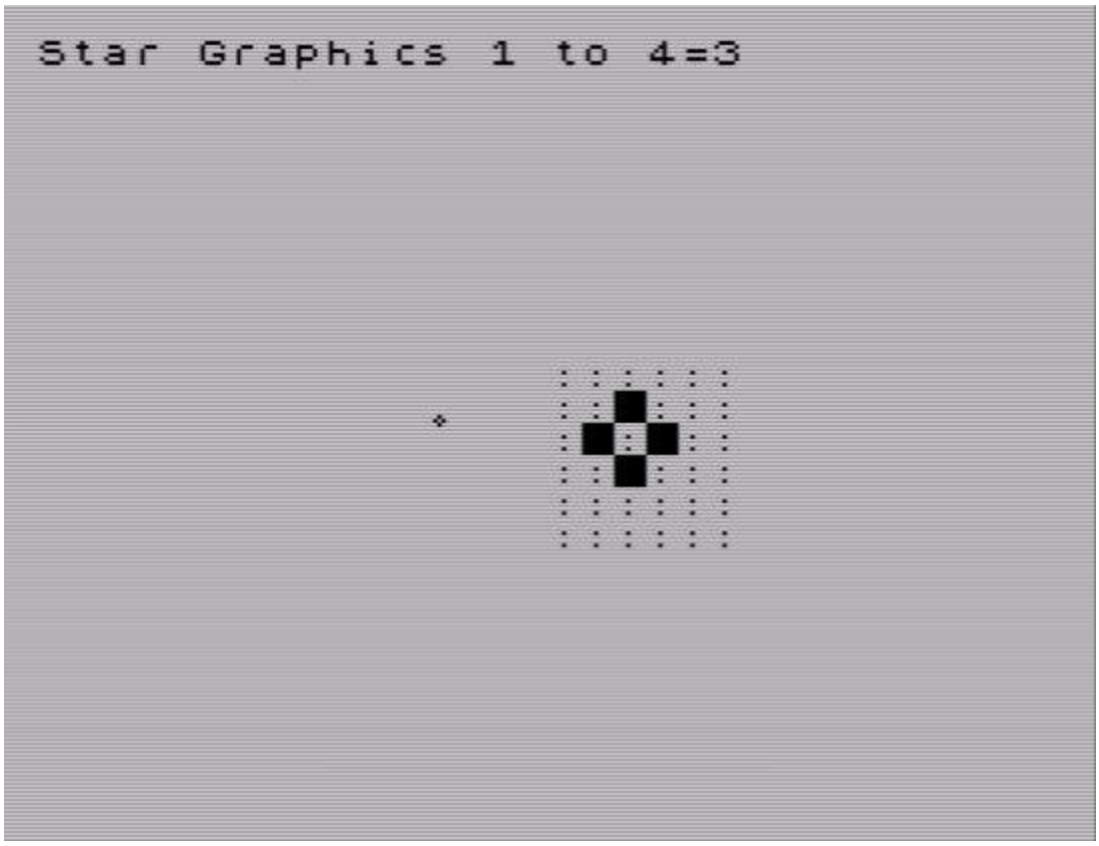
and so on through the sequence as required. Note that the first two conditional lines must include the statement NEXT n so that the program does not proceed beyond that line until the condition is satisfied.

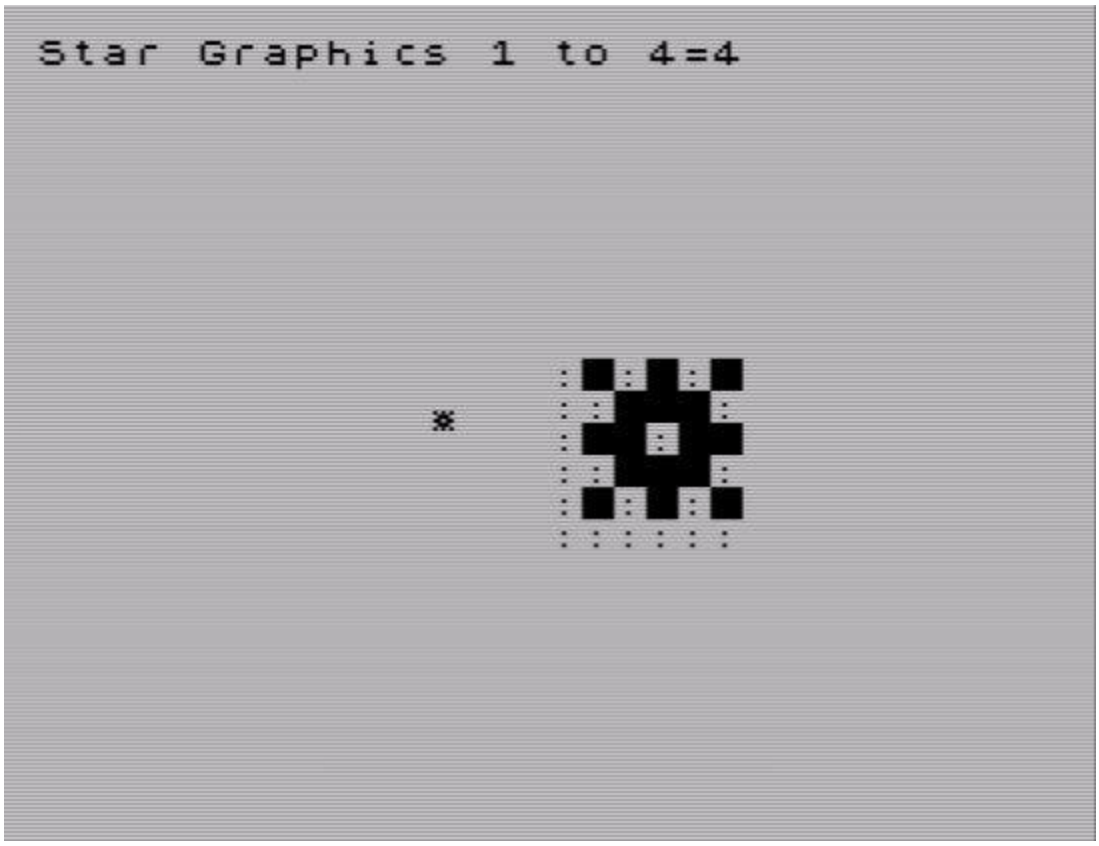
Figure 8.14

Typical shapes to represent brighter stars on computer starmaps. Shown enlarged here for clarity. The POINT command, via two FOR/NEXT loops, is used to scan the actual (tiny) images to the left of each sample which are used in the final starmaps.


Star Graphics 1 to 4=1


Star Graphics 1 to 4=2


Star Graphics 1 to 4=3


Star Graphics 1 to 4=4

```
2 PRINT "Star Graphics ";
3 PRINT "1 to 4=";INKEY$
4 PAUSE 0: LET a=VAL INKEY$
5 CLS: IF a>4 THEN RUN
6 LET x=99: LET y=80
7 PRINT AT 0,21;a
8 GO SUB a*100: GO SUB 1000
9 PRINT AT 0,0;: GO TO 1
100 PLOT x,y: DRAW 2,2
110 DRAW 2,-2: DRAW -2,-2
120 DRAW -2,2
199 RETURN
200 PLOT x,y: DRAW 0,2
210 PLOT x+1,y+1: DRAW -2,0
299 RETURN
300 PLOT x,y: DRAW 1,1
310 DRAW 1,-1: DRAW -1,-1
399 RETURN
400 PLOT x,y: DRAW 4,0
410 PLOT x+2,y-2: DRAW 0,4
420 PLOT x,y-2: DRAW 4,4
430 PLOT x,y+2: DRAW 4,-4
440 PLOT OVER 1;x+2,y
499 RETURN
1000 FOR n=0 TO 5: FOR f=0 TO 5
1010 IF POINT (x+n-1,y+f-2)=1 THEN PRINT AT 10+f,16+n;CHR$ 143
1011 IF POINT (x+n-1,y+f-2)=0 THEN PRINT AT 10+f,16+n;CHR$ 58
1020 NEXT f: NEXT n: RETURN
```

Flashing Stars

There are two basic ways to make a star (a single pixel) flash ‘on’ and ‘off’ using the OVER command, and this can be useful in highlighting a particular point in your programs involving starmaps. You have the option of the star remaining visible or disappearing at the end of the flashing sequence.

Routine using two FOR/NEXT loops to flash a star The following one-line routine demonstrates the principles using two FOR/NEXT loops. The T loop controls the number of flashes and the ‘n’ loop acts to switch the OVER command on or off (1 or 0). It is necessary to include a brief PAUSE (PAUSE 10) to make the flashing obvious

otherwise one is left with no more than a rapid flicker. It will be noted that the n loop finishes with a value of 0: in this way, the star remains visibly PLOTted at the end of the routine. If the values in the FOR/NEXT n loop are reversed thus:

…: FOR n = 0 TO 1 : …

(omitting the STEP routine as it is now superfluous), the star will disappear at the end of the routine with PLOT OVER 1 (ie unPLOT position of pixel).

```
1000 PRINT "flash star": FOR f=0 TO 10: FOR n=1 TO 0 STEP -1: PLOT OVER m;200,100:
PAUSE 10: NEXT n: NEXT f
```

Routine using a single FOR/NEXT loop to flash a star

This routine is probably easier to use and demonstrates the rather odd way in which the OVER command works. The OVER command is set to a value of 1 throughout — it unPLOTs (makes invisible) the pixel position.

However to unPLOT an unPLOTted pixel position via the next value in the FOR/NEXT loop, PLOTs the pixel back into visibility! Thus throughout the FOR/NEXT loop the OVER command becomes:

OVER 1… OVER 0… OVER 1… OVER 0… OVER 1… OVER 0…
and the star will flash in the required manner

If the FOR/NEXT loop has an effective number of steps that are uneven (2 TO 10 = 9 STEPs) then the OVER command will finish with a value of 0 and the star will be PLOTted and visible at the end of the routine — see Line 1000. If the FOR/NEXT loop has an even number of steps (1 TO 10 = 10 STEPs in Line 1001) the star will disappear with a final PLOT OVER 1 condition.

```
1000 PRINT "flash star-on": FOR f=2 TO 10: PLOT OVER 1;200,100: PAUSE 10: NEXT f
1001 PRINT "flash star-off": FOR f=1 TO 10: PLOT OVER 1;200,100: PAUSE 10: NEXT f"
```

Startrax (Stellar Proper Motion)

Astronomy, the study of the heavens, is the oldest science known to man: it dates back to 2000 BC, to the Chaldeans of Asia Minor. The star groupings or constellations familiar today were named by them. A remarkable feature of the constellations is that after 4000 years they appear virtually identical to us today as they did to those early astronomers — despite the fact that we know most stars are moving through space at tens of kilometres per second.

Why are the constellations not distorted beyond recognition by this random motion? Quite simply, the stars are so remote — distances measured in tens or hundreds of light years — the individual movements are undetectable to

the unaided eye. Only one star, Arcturus at 36 light years distance, has moved appreciably since early Greek star maps were prepared. And even this amount is very little — about the diameter of the full moon, ie½°.

To see the constellations change shape, it is necessary to have a time scale of hundreds of thousands of years and this program simulates this effect on two selected star groups — the Plough and Orion. The DATA is taken from Hutchinson's *Splendour of the Heavens* (1923). You have the option of allowing the individual stars to 'trail' their image across the screen or to PLOT as a moving pixel. The screen is presented in inverse video — BORDER 0: PAPER 0: INK 9.

In order that the PLOTting of the constellation selected is as smooth as possible, the individual star positions in the a$ are entered into DIMensional arrays x and y, the coordinate positions of each star. These arrays incorporate the 'proper motion values', as they are called, from the m$. How these values in the m$ are constructed will be described in detail later.

Economising memory

The method of handling the DATA is economical on the Spectrum memory (RAM). Although there is no real necessity in this case to save memory, the technique may be of interest and perhaps of use in your programming. As noted, the a$ contains the star positions and these are held as two 2-bit numbers:

Line 90
09 (x coordinate star 1)
17 (y coordinate star 1)
18 (x coordinate star 2)
22 (y coordinate star 2)

and so on to the end of the string array (star 16).

The m$ contains two 1-bit numbers per star to indicate the star's xx,yy proper motion, so the LENgth of the respective m$ is precisely half that of the associated a$. The a$ and m$ are repeated to the end of the DATA: each pair is used for a separate constellation. In order that the correct DATA is used on RUNning the program a conditional GOTO must be incorporated immediately after each pair of string arrays, ie:

110 IF b$(I) = "P" THEN GOTO 150

then the program will jump out of the DATA, carrying the last values of a$ and m$ which were read. The last pair of string arrays does not need a conditional GOTO as it is assumed to be the selected pair by default. Between lines 110 and 120 there is room for three more constellations via your DATA without renumbering or using multiline statements.

Remember that the first letter of the b$ (used to INPUT the name of the constellations in line 30) must be included in the associated conditional GOTO. The example in Line 110 is 'P' for the Plough. By only requiring one letter

(the first) to be correct, the program will ignore user misspellings (a necessary precaution as some of the constellation names can be very difficult).

Selecting DATA

The LENgth of the m$ is now used to DIMension to x and y arrays and the routine to Line 270 is used to enter the DATA into these arrays. In the case of Orion, with 20 stars and 20 positions for each star to be computed for subsequent PLOTting, this makes 400 x and 400 y positions. While the Spectrum does the necessary computation it displays a neat little 'I am computing' routine. Of course you could place every constellation (whether called upon for immediate display or not) into a DIMensional array but at 800 bytes per constellation it seems a little unnecessary for this program at least. In this program, the arrays are only reDIMensioned if the alternative constellation is called for display. Lines 50 and 60 recognise this condition and jump straight to the display if the same constellation is selected be it for a trailed or 'point image'.

Proper motion values

The point of this program lies in the proper motion values in the m$, which permits stars to move in any direction and velocity across the screen. Figure 8.15 explains the principle.

Figure 8.15

Typical grid layout for estimating the relative motion of stars in a Proper Motion program. + in the centre of the grid marks the point of rest (ie no motion required). The xx and yy coordinate positions are measured from the bottom right corner. Note that the east (e) and west (w) positions appear reversed from the normal convention. It is usual in astronomy to represent the skies above our heads with east to the left of south, which is the case here.

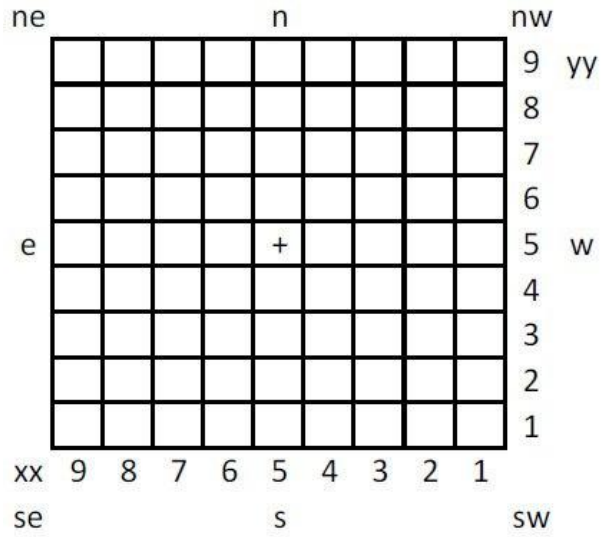
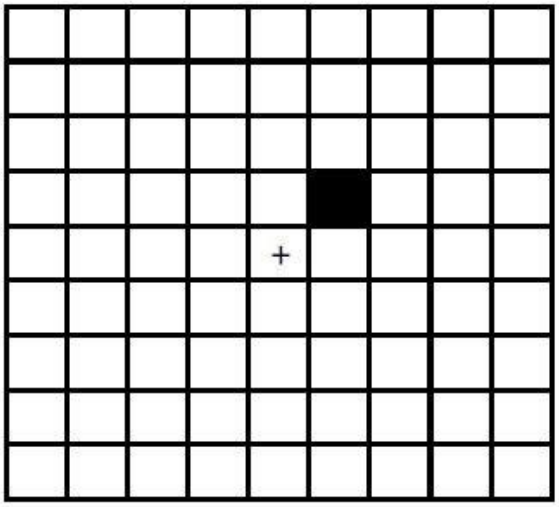
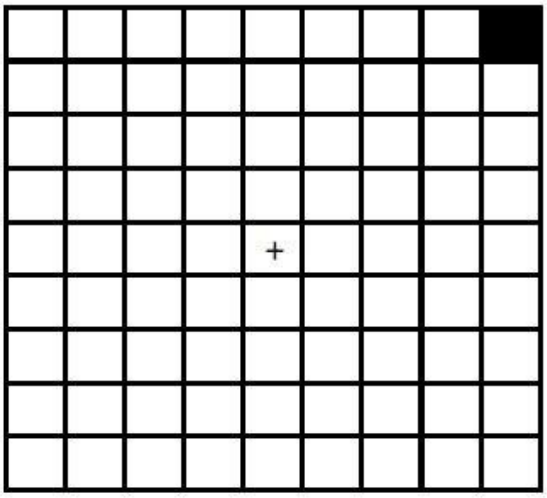


Figure 8.16

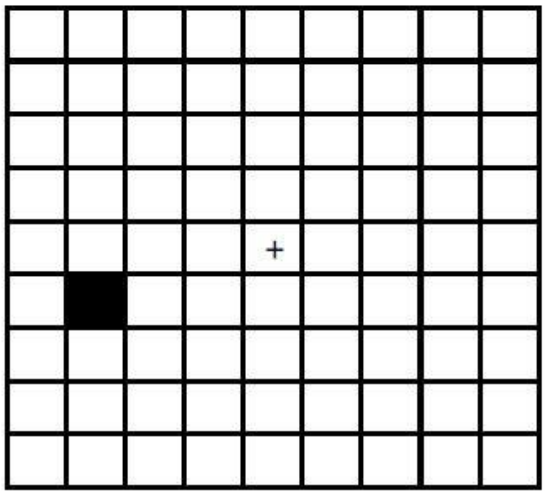
Examples of typical values which could be used in the program. The further the selected coordinates are from the centre of the grid the faster a star will move in that direction. These values are incorporated into the m$ until all the stars in the constellation are included.



m$="46"



m$="19"



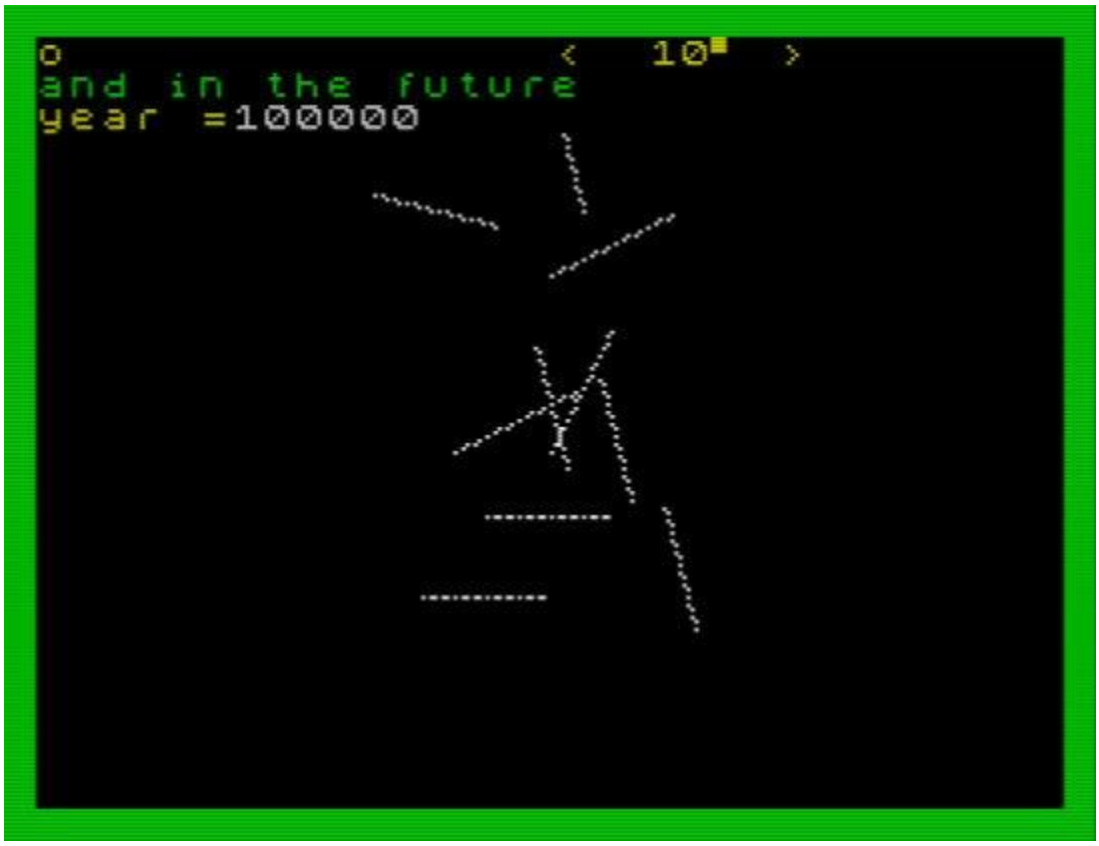
m$="84"

Figure 8.17

The familiar star pattern of Orion as seen today.

Figure 8.18

The apparently random motion of the stars in Orion over 200,000 years.



The grid is divided into nine horizontal (xx coordinate) and nine vertical (yy coordinate) boxes, a total of 81 boxes. To be different from the Spectrum PRINT and PLOT systems these coordinates start from the bottom right position. A star at 'rest' is placed at the centre of the grid and has an xx, yy coordinate value of 55. As a star is placed further from the centre of the grid, the velocity increases proportionally and in the direction indicated. For example a value of 46 indicates a small movement towards the NW (the top right corner). A value of 19 indicates the largest movement to the NW direction and so on for all points of the compass. Some typical sample values are shown in Figure 8.16.

Before the m$ VALue is entered into the x and y arrays, a value of – 5 is deducted from each number. Thus our m$ VALue of 55 becomes 00, ie no relative motion, and 46 becomes -1 +1. In this way only positive numbers need to be contained in the m$ although the required VALues are sometimes negative. The conversion is done in Lines 180 and 190. The FOR /NEXT f loop from Line 220 allocates the values to the x and y arrays which are incremented by the m$ values in Lines 230 and 250. In the final display the FOR/NEXT loop contains 20 STEPs and the middle STEP 10 represents the form of the selected constellation as we know it today. Thus a brief but powerful program results.

Screen display

The method of preparing starmaps on the Spectrum is described more fully in the **Starmaps program**. Let it suffice here to say that this example uses a screen resolution of 64*44 pixel positions in the DATA of the a$ and that this is expanded to the full 256*176 of the Spectrum by multiplying each PLOT position in Line 360 by 4. One

point should be observed if you extend the program to include your own selected constellations. Avoid the edge of the screen with your basic starmaps. The proper motion in a star's movement will cause the program to crash if it attempts to PLOT beyond the maximum coordinates (255 for x; 175 for y). Figures 8.17 and 8.18 are typical screen COPYs.

```
10 REM Startrax
20 LET c$=""
30 INPUT FLASH 1;"Plough or Orion ";b$
40 INPUT "Trailed or point images (t/p)? "; LINE t$
50 IF c$=b$(1) THEN GO TO 290
60 LET c$=b$(1)
70 BORDER 0: PAPER 0: CLS
80 INK 9: FOR n=1 TO 9: PRINT PAPER n; FLASH 1;AT 5+n,11;"COMPUTING": NEXT n: GO
SUB 500
90 LET a$="09171822252133203614472648195321"
100 LET m$="3594938484268484"
110 IF b$(1)="p" OR b$(1)="P" THEN GO TO 150
120 LET a$="25333337363130203221342232142809401136l9"
130 LET m$="94499797497995954949"
150 DIM x(LEN m$,20)
160 DIM y(LEN m$,20)
170 FOR n=1 TO LEN m$ STEP 2
175 LET h=n*2
180 LET d=VAL m$(n)-5
190 LET e=VAL m$(n+1)-5
200 LET a=VAL a$(h-1 TO h)+d
210 LET b=VAL a$(h+1 TO h+2)+e
220 FOR f=1 TO 20
230 LET a=a-d/10
240 LET x(n,f)=a
250 LET b=b-e/10
260 LET y(n,f)=b
270 NEXT f: NEXT n
290 CLS : PRINT INK 6; b$; "10chr$ 130" ","in the distant past","year ="
300 FOR n=1 TO 20
310 LET y=-1e5+1e4*n
320 PRINT AT 2,6;y;" "
330 IF t$="t" THEN FOR f=1 TO 0 STEP -1
340 IF t$"t" THEN FOR f=0 TO 1
```

```
350 FOR p=1 TO LEN m$ STEP 2
360 PLOT BRIGHT 1; OVER f;x(p,n)*4,y(p,n)*4
370 NEXT p
380 IF y=0 AND f=0 THEN GO SUB 430
390 IF y=1e5 AND f=0 THEN GO SUB 420
400 PAUSE 1+10*(1-f)
410 NEXT f
420 NEXT n: GO SUB 500
425 GO TO 30
440 PRINT INK 5;AT 1,0;"Today's ";b$;" ": GO SUB 500
450 PRINT INK 4;AT 1,0;"and in the future ": RETURN
500 FOR v=0 TO 5: FOR k=7 TO 0 STEP -1: BORDER k: BEEP .01,40-k: NEXT k: NEXT v:
RETURN
```

Stellar Magnitude

Astronomers have a scale to measure the apparent brightness of the stars called magnitude. This was first classified by Hipparchus in 127 bc: he described 'the brightest stars as magnitude 1′ and 'the faintest stars as magnitude 6′ with four steps in between. This basic system has been retained ever since, but of course refined and explained mathematically, principally by Pogson in the last century. Pogson ascribed a value of 2.512 to 1 as a ratio between one whole magnitude and the next. This equates to a ratio of 100:1 for a star of first magnitude against a star of sixth magnitude just visible to the naked eye. Pogson also noted that the common logarithm of 2.512 is precisely 0.4, much simplifying computation.

This short program uses a simple FOR/NEXT loop to PRINT to the screen a magnitude range from -26 (the Sun) to +24 (faintest star detected with the 200-inch Hale telescope at Mt Palomar, California). The ratio to the standard star Vega at magnitude 0.0 is also displayed for each whole magnitude step. Our Sun proves to be 1:2.51E10 (25,000,000,000) times brighter than Vega: a magnitude +24 star proves to be about a similar ratio fainter than Vega.

Using a series of conditional PRINT statements the list is punctuated by some mainly familiar objects that match particular magnitudes. The standard star Vega is made to FLASH to identify itself readily.

The Spectrum does not use common logs but natural logs (LN function) to a base of 2.71828 … so Pogson's neat common log relationship has to be fudged in this program, via Line 80, to give a ratio of 100:1 over five magnitudes.

```
10 REM Stellar Magnitude
15 PRINT "Mag Name","1:ratio"
```

```
20 FOR n=-26 TO 24 STEP 1
30 PRINT PAPER 6;(" " AND ABS n0);n;" ";
37 PAPER 5
40 PRINT "Sun" AND n=-26;
41 PRINT "Full Moon" AND n=-13;
42 PRINT "Venus" AND n=-4;
43 PRINT "Sirius" AND n=-1;
50 PRINT FLASH 1;"Vega-standard" AND n=0;
51 PRINT "Uranus" AND n=5;
52 PRINT "Neptune" AND n=8;
53 PRINT "Barnard's Star" AND n=10;
60 PRINT "Pluto" AND n=14;
61 PRINT "200"" eye limit" AND n=19;
62 PRINT "200""photolimit" AND n=24;
70 PAPER 7
80 PRINT TAB 18;EXP (LN 2.51193*-n): NEXT n
```

Chapter 9 – Further Programs (The Messier List)

The following programs do not fit conveniently into previous chapters and have been gathered here. Although they are all diverse, I hope that you will find them interesting.

The Messier List

There is great kudos in having one's name permanently applied to a heavenly body, but chance discoveries are exceptionally rare and an intimate knowledge of the night sky is really required if you are to have any hope of success.

Curiously, the French astronomer Charles Messier is renowned no longer for the many comets he discovered, but for his 'Catalogue of 104 Non-stellar Objects' (published in 1784) which forewarned the unwary against false claims: each object on the 'Messier List' is a remote deep-sky nebula or galaxy which appears comet-like in a small telescope or binoculars.

The program

The following program, designed around the 'Messier List', indicates how DATA can be stored economically in CODE form in string arrays. The program has two options:

1. To list all 104 obj ects in numerical sequence with host constellation and type of object.
2. To quiz the user as to where a random selection of these objects are to be located.

Good use is made of Spectrum colour to enliven the displays and to mark your score in the quiz mode. In the latter your initial title of 'ASTRONOMER' is slowly cut down in size for each incorrect answer!

Data store

The DATA is stored in the following manner:

a$ = 104 characters in the range 0 to 9 & A to Y (total = 35) for the 35 constellations hosting the 104 objects.

c$ = 35 constellation names to correct IAU abbreviations.

k$ = 104 numeric CODEs in the range 0 to 5 (total = 6 CODEs) for the class of object.

Unlike the non-standard ZX-81 CHR$ set (on which this program was originally designed), CHR$ 0 to 9 and A to Y are not consecutive in the Spectrum ASCII set (used for this book). Thus the a$ is handled in two ways to extract the DATA used in the variable z as follows:

```
270 IF CODE a$(n) < 58 THEN LET z = 1 + VAL a$(n): GOTO 290 280 LET z = CODE a$(n)
- 54
```

Lines 410 and 420 work in a similar way. Consult your Spectrum Manual to see the values returned by various CODEs. The k$ is used to return a VALue used in the conditional GOSUB eg:

`320 GOSUB 100 + VAL k$(n)* 10 ie GOSUB 100, 110, 120` etc.

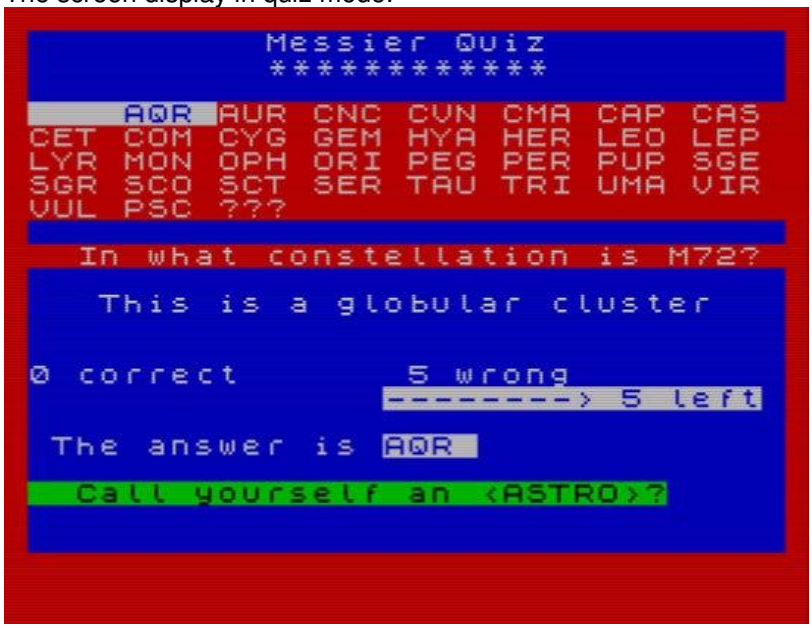
and then PRINT the object type (eg 'an open cluster') and RETURN. Line 480 works in a similar way.

Figure 9.1

First page of the Messier List.

Figure 9.2

The screen display in quiz mode.



RUNning the program

Once the program has been keyed in and RUNs satisfactorily as checked against the sample screen displays in Figures 9.1 and 9.2, it is worth remembering to set the CAPS LOCK. In the quiz mode lower case answers are unacceptable. If you need a little help and have aZX printer, the 'Messier List' can be COPYed a 'page' at a time with:

BREAK COPY ENTER

When the COPY is complete, press CONTINUE ENTER for the next page.

```
10 REM Messier List & Quiz
20 DIM b$(20): INK 9
30 LET y$="ASTRONOMER"
40 LET a$="S14PRPPOIIQIDIKROOIOOOOOOQWOA600TL
B222AU5JJ35MMCVH479OOGGVVVVP49EE3COON11XOLBJF
PUUCV9VV9VVYDM4EEU999UY7V"
50 LET c$="AND AQR AUR CNC CVN CMA CAP CAS CET COM CYG GEM HYA HER LEO LEP LYR MON
OPH ORI PEG PER PUP SGE SGR SCO SCT SER TAU TRI UMA VIR VUL PSC ??? "
60 LET K$="42222114221222214124121111321255511
11111014411111515122223555525555122222152354 2
25555555555021555355550 15"
70 GO TO 200
100 PRINT "a Messier mistake!"
105 RETURN
110 PRINT "an open cluster "
115 RETURN
120 PRINT "a globular cluster"
125 RETURN
130 PRINT "a planetary nebula"
135 RETURN
140 PRINT "a diffuse nebula "
145 RETURN
150 PRINT "an external galaxy"
155 RETURN
200 INPUT "Messier List or Quiz (l or q)?", LINE l$: CLS : PAUSE 1
210 IF l$"l" AND l$"L" THEN GO TO 360
230 PRINT TAB 10;"Messier List"
240 PRINT TAB 10;"**********": GO SUB 600
250 PRINT "No Con Code Type"
260 PRINT : FOR n=1 TO 104
```

```
270 IF CODE a$(n)<58 THEN LET z=1+VAL a$(n): GO TO 290
280 LET z=CODE a$(n)-54
290 LET j=VAL k$(n)
300 BEEP .01,j*10: PAPER j
310 LET x$=c$(z*4-3 TO z*4)
320 PRINT "M";n;TAB 5;x$+a$(n)+""+k$(n);" ";
330 GO SUB 100+VAL k$(n)*10
340 IF n/5=INT (n/5) THEN PRINT
350 NEXT n: PAPER 7: GO TO 590
365 POKE 23658,8
370 LET c=0: LET w=0
380 FOR n=1 TO 10: CLS
390 LET r=1+INT (RND*103)
400 IF r=40 OR r=91 OR r=102 THEN GO TO 390
410 LET k=VAL k$(r): BORDER k: LET d=w
420 IF CODE a$(r)<58 THEN LET z=1+VAL a$(r): GO TO 440
430 LET z=CODE a$(r)-54
440 LET x$=c$(z*4-3 TO z*4)
450 PRINT TAB 10;"Messier Quiz"
460 PRINT TAB 10;"************"
470 PRINT PAPER k'c$+b$"" In what constellation is M";r;"?"+(" " AND r<100)+(" "
AND r<10)
480 PRINT '" This is ";
490 GO SUB 100+VAL k$(r)*10
500 INPUT t$: LET t$=t$+" "
510 IF t$=x$ THEN LET c=c+1: GO SUB 600
520 IF t$x$ THEN LET w=w+1
530 PRINT "c;" correct",w;" wrong"
540 PRINT FLASH 1;AT 15,(d-w)*15;"———> ";10-n;" left"
550 IF t$x$ THEN BEEP .2,-16: BEEP .5,-30: PRINT '" The answer is "; FLASH 1;x$;AT
3+INT (z/9),0;TAB (z-1)*4;x$
560 PRINT PAPER n-1;AT 19,0;" Call yourself an ?"
570 PAUSE 300: NEXT n
580 PRINT PAPER 1; PAPER 4'" Time is up "+y$+b$( TO 8)
585 POKE 23658,0
590 GO SUB 600: PRINT #0;" Press any key to continue.": PAUSE 0: GO TO 200
600 FOR a=-30 TO 30 STEP 5: BEEP .1,a: NEXT a: RETURN
```

Telescope

This program sets out to answer the following questions:

1. What is the faintest star visible in a 6 cm or a 6 m telescope?
2. What is the maximum and minimum useful power eyepiece to use?
3. Why is the Hale 5 m telescope not suitable visually?
4. What sort of area in degrees will a 35 mm camera lens cover?
5. What is the photographic and visual resolution of a 15 cm telescope?
6. What astronomical performance can I expect from my telescope/camera?

The last question summarizes most of the previous questions.

The answers

There is nothing particularly special about this program other than the fact that it brings together all the facts and figures pertinent to telescopes, binoculars and cameras, when used for astronomy. Only two items have to be INPUT — the aperture of the instrument in millimetres and the f/ratio as it is called. The latter is the ratio of the diameter of the lens (of the camera, for example) against the focal length and this in turn is the distance from the lens to the image the lens forms. From these two INPUTs a lot of data can be computed in just one second that would need a library of books to reference. Just key in the program and RUN it.

The program

The program is based on my own many years of practical experience at the telescope. Only one item — the visual or Dawes resolving power of a telescope — assumes a perfect instrument, perfectly aligned on a perfect night free from atmospheric turbulence (twinkling stars) and an experienced eye. The balance of the items is either reasonably indisputable (like the power of a given eyepiece) or not too contentious (like the limiting visual magnitude attainable on a clear night away from town lights).

Figure 9.3 shows a screen COPY for an f/4 120 mm focal length camera lens and Figure 9.4 a screen COPY for my own telescope. It may be useful to go through each sample and the relevant program lines to explain the significance of the data.

Figure 9.3

Astro performance of f/4: 120 mm fl lens.

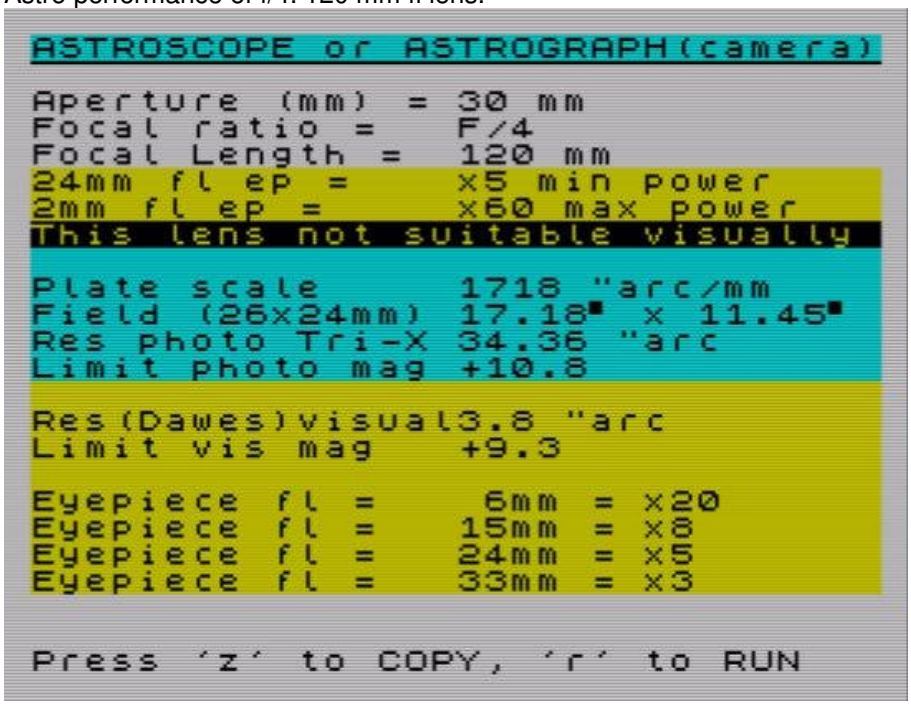
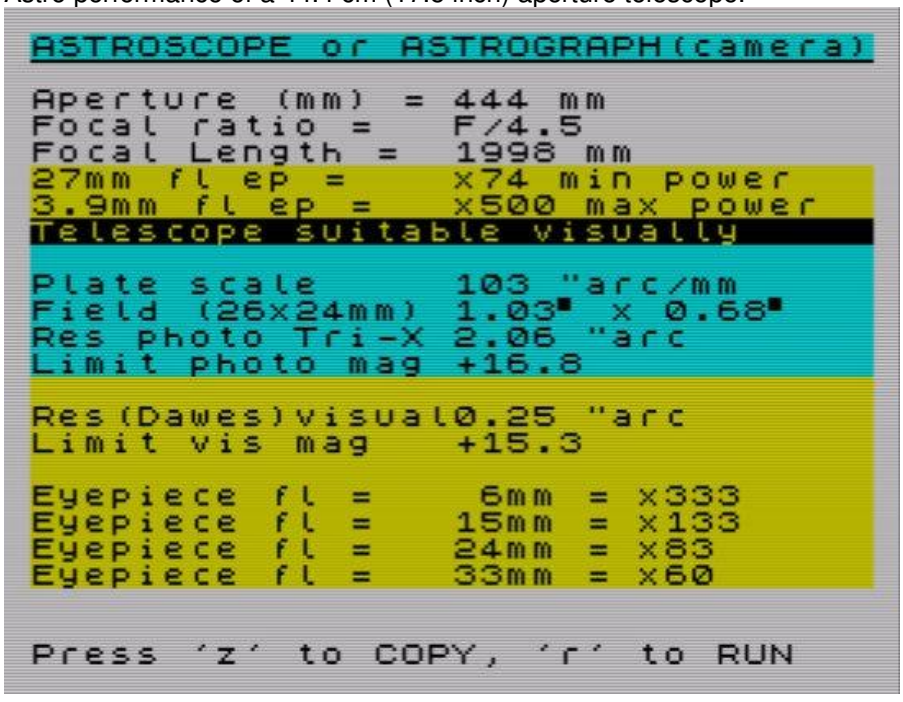


Figure 9.4

Astro performance of a 44.4 cm (17.5 inch) aperture telescope.



The first two lines of the printout are the INPUT of aperture and f/ratio and these, multiplied together, produce the third line — focal length. The next two lines are the maximum and minimum power eyepieces for the system and

their respective focal lengths. It will be noted that the next line in Figure 9.3 for the camera lens comments: “This lens not suitable visually”

This is worked out by slicing the A$ and T$ in Line 20 according to the conditional Lines 170 and 180. In this case the focal length is less than 300 mm and therefore not identified as a telescope.

The next group of four items specifically refers to the photographic performance of the instrument. The items are, in order, the ‘plate scale’ or number of seconds of arc per mm of film, the area in degrees for a 35 mm format camera, the resolving power of TRI-X film in seconds of arc and, finally, the faintest star normally recorded in a guided exposure before skyfogging (stray light) sets in. The next group of two items refers to the visual resolving power (to Dawes formula) in seconds of arc and the limiting visual magnitude the instrument will show. Notice how a given lens or telescope will record stars on film to 1.5 magnitude less than the eye will record at the same instrument. This is because the eye (when fully adapted to dark conditions) sees no more after about quarter of a second, however much you peer through the eyepiece, whereas the photographic film slowly builds up an image over seconds, minutes and even hours in perfect conditions. The human eye’s sensitivity is much superior to the fastest 1000 ASA film but tires easily. Also the eye’s resolving power — pixel for pixel — is again superior to all but the slowest film, say 20 ASA.

The final section of the printout gives some typical focal length eyepieces and notes the effective powers with the lens concerned. This is done by the FOR/NEXT loop in Lines 250 to 270. An option to COPY the screen is included on Line 290.

Interpreting results

Practical experience would indicate that any telescope magnifying more than x 500 is pointless. The Earth’s atmosphere is never steady enough to support such a magnification and remain sharp and clear. Line 100 sets this upper limit: otherwise Line 90 assumes that the aperture (in mm) x 2 is the upper limit. The lowest useful power eyepiece is set by the maximum diameter of the human pupil, 7 mm when adapted to dark. I have adopted a diameter of 6 mm from practical experience of numerous tests. Effectively, if the eyepiece power is too low, not all the light entering the telescope reaches the retina because of the physical limit to the diameter of the eye’s iris.

In the case of the Hale 200-inch telescope at Mt Palomar with a clear aperture exceeding 5000 mm, a minimum power eyepiece exceeding x 750 would be necessary for all the light to enter the eye for the reason mentioned above. This giant telescope is not for looking through but is a huge camera for peering into the most remote regions of the cosmos.

The following variables carry the various formulae used in the program:

A = the clear aperture in mm
F = the focal ratio of system
FL = the focal length of system in mm
L = lowest useful power eyepiece

H = highest useful power eyepiece

V = limiting visual stellar magnitude P = plate scale in ” arc/mm

The two last variables are also used for the following:

P/100 = field width in degrees

P/50 = TRI-X film resolving power in ” arc/mm

V + 1.5 = limiting photo magnitude 114/A = Dawes visual resolution

```
5 REM Telescope Performance
10 CLS : PRINT PAPER 5;”ASTROSCOPE or ASTROGRAPH(camera)””
15 PLOT 0,167: DRAW 255,0
20 LET a$=”not suitable visually”: LET t$=”Telescope This lens “
30 PRINT “Aperture (mm) =”,
40 INPUT a: IF a500 THEN LET h=500
110 LET v=1.9+INT (LN (a*a)*11)/10
120 LET p=INT (206264/fl)
130 PRINT “Focal Length =”,fl;” mm”
139 PAPER 6
140 PRINT INT (fl/l+.5);”mm fl ep =”,”x”;l;” min power”,
150 PRINT INT (fl/h*10)/10;”mm fl ep =”,”x”;h;” max power”,
160 IF l>=h THEN PRINT FLASH 1;t$( TO 10);a$,
170 IF l=25 AND fl>=300 THEN PRINT FLASH 1;t$( TO 10);a$(5 TO ),
180 IF l<h AND (a<25 OR fl<300) THEN PRINT FLASH 1;t$(11 TO );a$,
190 PAPER 5: PRINT ,,”Plate scale”,p;” “”arc/mm”,
200 PRINT “Field (26x24mm)”,INT p/100;”[G]2 x “;INT (p*2/3)/100;”[G]2 “,
210 PRINT “Res photo Tri-X”,INT (p*2)/100;” “”arc”,
220 PRINT “Limit photo mag”,”+”;v+1.5,
229 PAPER 6
230 PRINT ,,”Res(Dawes)visual”;INT (11400/a)/100;” “”arc”,
240 PRINT “Limit vis mag”,”+”;v,,,
250 FOR n=1 TO 6 STEP 1.5
260 PRINT PAPER 6;”Eyepiece fl = “,(” ” AND n=1);6*n;”mm = x”;INT (fl/6/n),
270 NEXT n: PAPER 7
280 PRINT #0;”Press ‘z’ to COPY, ‘r’ to RUN”: PAUSE 0
290 IF INKEY$=”z” THEN COPY
300 RUN
```

Star Tester

This is a short demonstration program to show how a simple quiz, using questions and answers, can be set up. Obviously, the quiz does not have to relate to astronomy, but it does for the purposes of this book. The program as it stands contains only ten star names, grouped in pairs in a$ and this is far too few to be really effective. In the example, 'Antares' = 'Alpha Scorpii', 'Regulus' = 'Alpha Leonis' — these are the alternative names for each star.' Thus the program is ripe for expansion by the user perhaps with entirely different paired names.

The program

The program works in the following way. A DIMensional array is set up in line 30 to accept (in this case) 10 names with a maximum length of 13 characters to cope with the longest a$ – a$(2), a$(6) and a$(10) all have a total of 13 letters and spaces. A DIM b$(13), a single DIMensioned array, is also set up to receive the answers to be INPUT in Line 230.

The computer asks the user up to ten questions via the FOR/NEXT f loop in Line 160. The variable z in Line 170 selects a RANDOM number between 1 and 10 and PRINTs to the screen the a$ with this number. The computer also decides that the answer will be:

a$(z + t)

where variable t acts as a switch eg

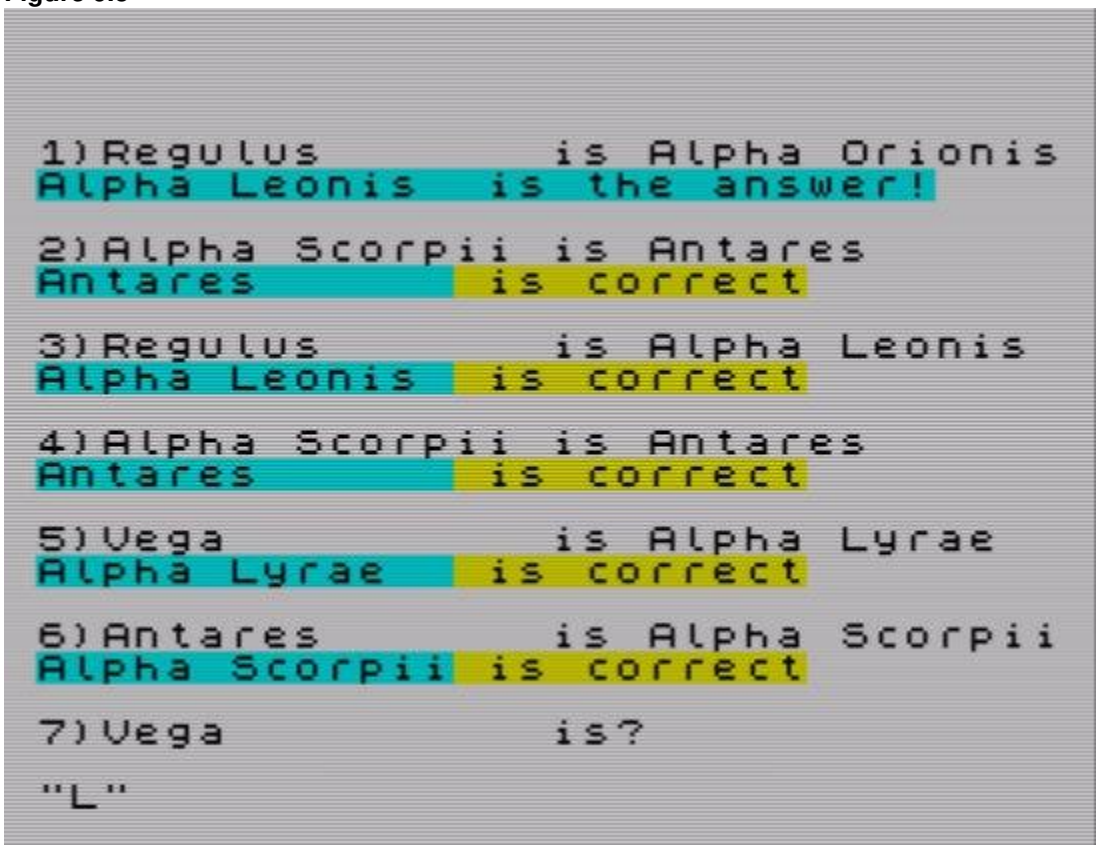
if z = 1 then t = 1 if z = 2 then t = – 1

and so on for each odd or even number that is thrown up for z so that t has a value of 1 or – 1 as the case may be. If a$(l) to a$(10) in Lines 40 to 130 inclusive are checked, then it can be seen that the computer has identified each answer correctly by grouping the a$ in pairs.

Line 230 INPUTs and PRINTs the answer, and the '?' is overprinted by using the expression PRINT CHR$ 8; b$. Lines 310 and 320 mark the answers and tot up the correct results with the variables score which is PRINTed at the end of the f loop.

The program uses the automatic scroll conditions by POKEing (POKE 23692, 255) and PRINTing below screen line number 21, so that the questions and answers flow up from the bottom of the screen next to the INPUT line 22. This is effected by program Line 350. Figure 9.5 is a typical COPY of the screen display.

Figure 9.5



Although the computer is looking for an answer which is precisely correct in upper and lower case lettering, there are insufficient permutations in a range of numbers from 1 to 10 without frequent repetition. Ideally the minimum should be about 30 questions, ie 15 paired questions/answers — preferably more to avoid repetition. Amending the program to receive extra questions involves minimal changes and can be done in two ways.

1) Additional listing in the a$

This is fairly straightforward and simply means squeezing more a$s into the listing, starting at a$(ll) — assuming the first 10 questions/answers in the listing are to be retained. For neatness, Lines 40 to 130 should be renumbered (perhaps as Lines 40 to 50 inclusive) and the extra new items for the a$ started from Line 51:

51 LETa$(11) = "…star name 6a…"
52 LETa$(12) = "…star name 6b…"
53 LETa$(13) = "…star name 7a…"
54 LETa$(14) = "…star name 7b…"
55 LETa$(15) = "…

and so on with all the extra data. Finally, amend the DIMensional array in Line 30 to account for the extra questions included:

30 DIM a$ (total number in a$, 13): DIM b$(13)

and amend the last value in the Line 170:

170 LET z = 1 + INT (RND*total number of questions in a$)

The program can now be RUN (RUN because all the program is contained in the LISTing).

2) INPUT subroutine

There is no particular need for the questions and answers to appear in the program and they can be INPUT via a subroutine. The procedure is as follows:

Delete Lines 20 to 130 inclusive

Amend Line 280 (final command) to GOTO 1 (in lieu of RUN)
Add the subroutine 400 DIM a$(50,13): DIM b$(13)

This allows for 50 questions/answers — 25 pairs — a maximum of 13 characters long, including spaces.)

```
410 FOR n = 1 TO 50
420 PRINT n; " "; LET c$ = ""
430 INPUT c$ : PRINT c$
440 IFc$ = "stop" THEN STOP
450 LET a$(n) = c$
460 NEXT n: STOP
```

The subroutine is started with GOTO 400 and the paired questions/answers are INPUT sequentially. It is not necessary to use up all the DIMension a$ array if you run out of questions as long as the last INPUT was an even numbered item. Simply INPUT “stop” and Line 440 will STOP the program. The program should now be SAVEd with GOTO 9990 as a precaution and VERIFYed. Now amend Line 170 to read:

170 LET z = 1 +INT(RND*(n-l))

The variable n is the last value + 1 used in the INPUT loop in Line 440. The program should be SAVEd and VERIFYed as version no 3 by amending Line 9990 to read:

9990 SAVE “startest3″ LINE 1

and then, as a direct command, GOTO 9990. The amendment will ensure that the program starts automatically when LOADed with LOAD “”or LOAD “startest3″ (which is preferred) and so avoid the pressing of the RUN key which will erase all the DATA from the computer. All would not be lost of course — simple reLOAD but this time enter GOTO 1.

Now that the program is safely on tape, it is time to test the program started with GOTO 1.

Adding more questions

If less than the full quota of 50 questions/answers was initially INPUT, then more can be added by BREAKing the program and entering as a direct command:

CLS: GOTO 420

which will pick up the INPUT routine where it left off and start the n loop again. Again the INPUT can be terminated before all 50 items have been entered with:

INPUT “stop”

and as a direct command to restart the quiz with GOTO 1. The current value of n will again update the maximum RANDOM number for the variable z. It is a wise precaution to SAVE each version where more DATA is INPUT before the quiz is tested. Better safe than sorry.

```
10 REM Star Tester
30 DIM a$(10,13): DIM b$(13)
40 LET a$(1)=”Antares”
50 LET a$(2)=”Alpha Scorpii”
60 LET a$(3)=”Regulus”
70 LET a$(4)=”Alpha Leonis”
80 LET a$(5)=”Betelgeuse”
90 LET a$(6)=”Alpha Orionis”
100 LET a$(7)=”Vega”
110 LET a$(8)=”Alpha Lyrae”
120 LET a$(9)=”Rigel”
130 LET a$(10)=”Beta Orionis”
140 LET score=0
160 FOR f=1 TO 10: LET t=1
170 LET z=1+INT (RND*10)
180 LET q=z/2
190 IF INT q=q THEN LET t=-1
200 GO SUB 340
210 PRINT f;”)”;a$(z);” is? “;
220 LET x$=a$(z+t)
230 INPUT b$: PRINT CHR$ 8;CHR$ 8;” “;b$
240 GO SUB 300: LET x=z
250 GO SUB 340: PRINT : NEXT f
270 GO SUB 340: PRINT “Your score: “;score;” correct answers”
280 PRINT FLASH 1;” Try again? – press any key “: PAUSE 0: RUN
```

```
300 GO SUB 340
310 PRINT PAPER 5; FLASH 1 AND b$x$;x$;
320 IF b$=x$ THEN LET score=score+1: PRINT PAPER 6; FLASH 1;" is correct": GO TO
340
330 PRINT PAPER 5;" is the answer!"
350 POKE 23692,255: PRINT AT 20,0: PRINT : RETURN
```

Ellipses

For astronomical presentations an 'ellipse' command is sadly missed from the Spectrum (and most other micros for that matter) but the following short routines will suffice, using the machine's trigonometry functions of SIN and COS. These routines prove much less cumbersome than some published examples using complex algebraic formulae. Also keeping the routine brief ensures that the results are executed quickly, so speeding up the PLOTting process.

Effectively an ellipse is a circle seen at an angle so that its form is compressed. With minor changes, each routine can be made to PLOT in a clockwise or anticlockwise direction with the long or major axis either vertical or horizontal.

The Ellipse routines

Key in and RUN the first routine called Ellipse (see Figure 9.6). You will notice that in most cases the variable names are sufficiently long to describe their function in the program:

xaxis = x (horizontal) coordinate position
yaxis = y (vertical) coordinate position
rad = maximum radius of ellipse called semimajor axis
tilt = INPUT angle of tilt to a circle
angle = compression factor to produce an ellipse

The formula in the variable 'angle' contains a small make-weight value of 0.1 added to a variable 'tilt'. This is done to stop the program crashing if an INPUT of 0° is made (a circle shown edge-on). The value returned by 0° would be a truly infinite number exceeding the Spectrum's numeric capacity which is limited to a range from $3*10^{-39}$ to $7*10^{38}$.

The short routine called Ellipse Angle Values (Figure 9.7) shows what values are returned by the formula called angle between 0° and 90° in whole degree steps.

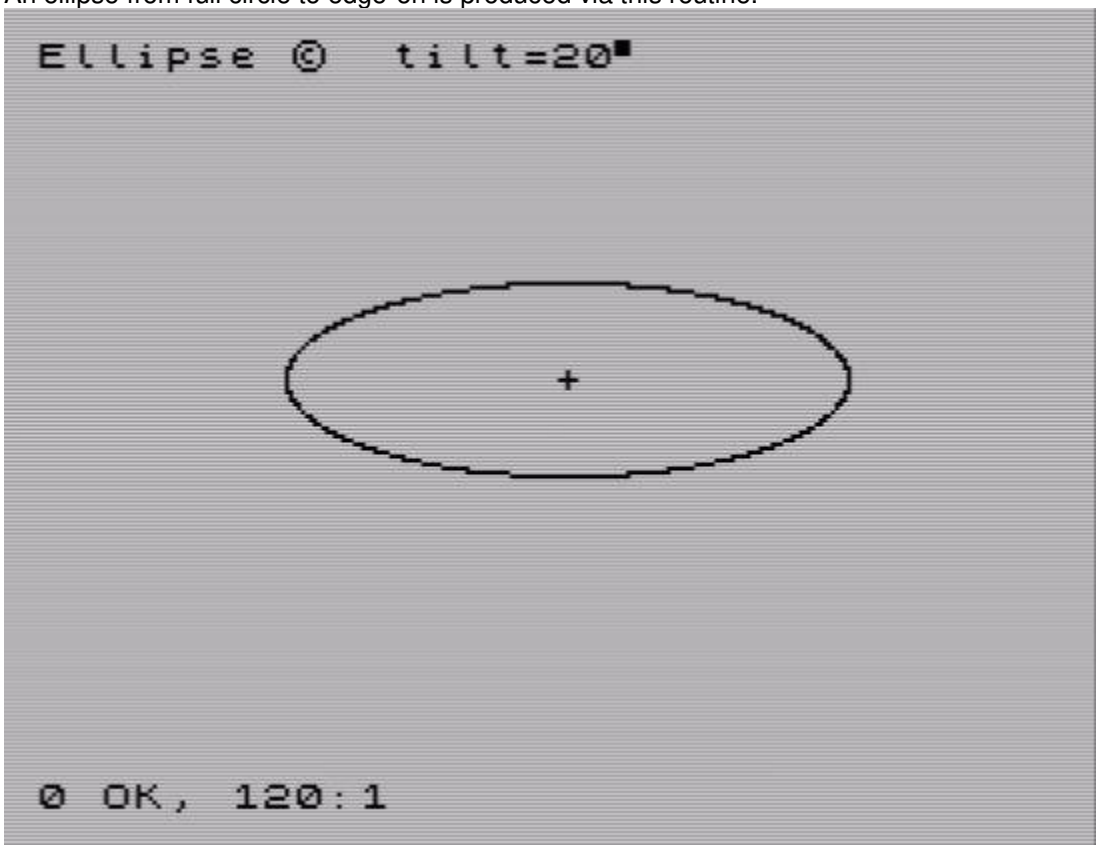
In reality, 0° is made to return a value as if 0.10, ie 1/10° or 6′ arc, and 0.1 is added throughout the sequence. The Spectrum's PLOTting accuracy is well below this small angle and the results (as PLOTted) are indistinguishable from whole degrees.

It should now be evident that if the somewhat improbable INPUT angle for tilt of -0.1° is made, the Ellipse program will crash because:

- 0.1 + 0.1 = 0°

Figure 9.6

An ellipse from full circle to edge-on is produced via this routine.




```
10 PRINT "Ellipse @ tilt=";
30 LET xaxis=132: LET yaxis=91
40 LET radius=70
50 INPUT "0CHR$130 to 90CHR$130 ",tilt
60 PRINT tilt;"CHR$130 "
70 PRINT AT 10,16;"+"
80 LET angle=1/SIN ((.1+tilt)/180*PI)
90 INPUT "step (1 to 10)",s
100 FOR f=0 TO PI*2 STEP 1/s/10
110 PLOT xaxis+SIN f*radius,yaxis+COS f*radius/angle
120 NEXT f
```

Figure 9.7

```
1 REM Ellipse angle values
```

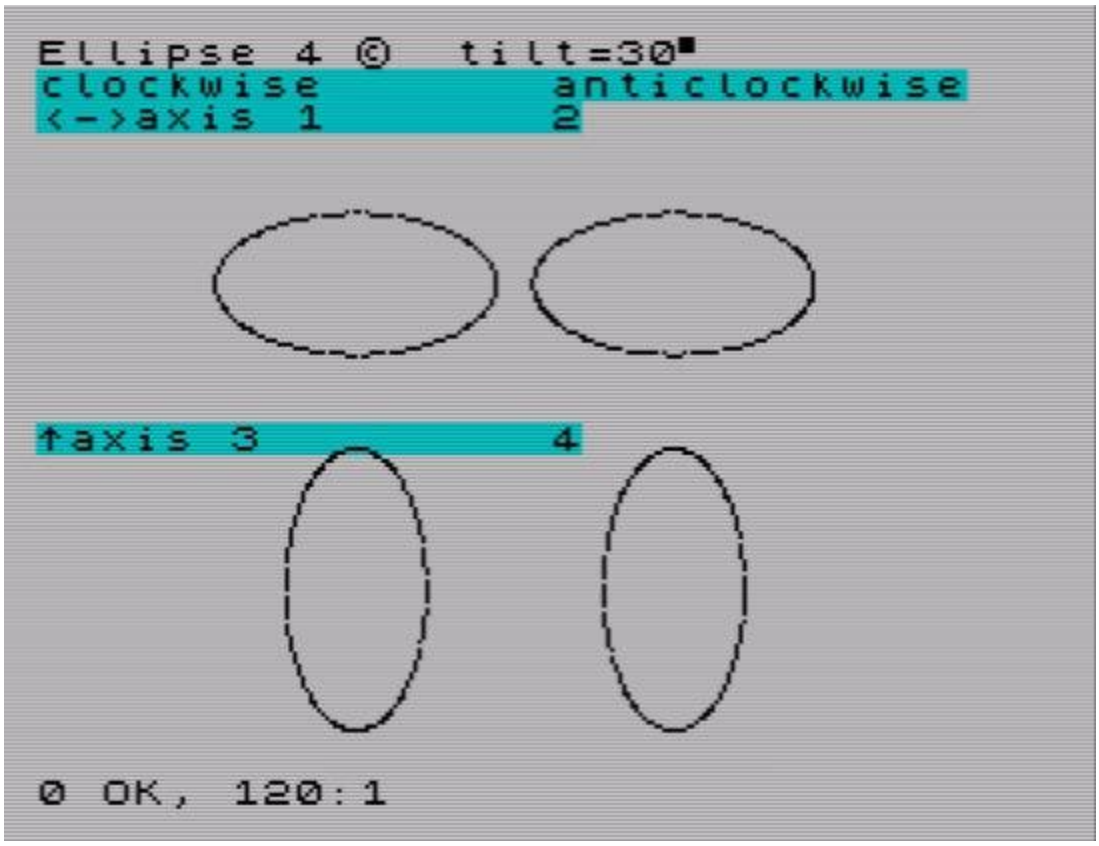
```
5 FOR x=0 TO 90
7 LET tilt=x
10 LET angle=1/SIN ((.1+tilt)/180*PI)
20 PRINT angle,tilt;CHR$ 130
30 NEXT x
```

Ellipse 4

This routine is precisely the same as the above routine but is used to demonstrate the simultaneous PLOTting of four ellipses. Againthe tu.o all four ellipses and the PLOTting frequency via variables can be selected by he two INPUT commands. Figure 9.8 shows two example COPYs from the screen.

Figure 9.8

Samples of ellipses PLOTted in different directions and long (major) axes.



The ellipses are made to PLOT in a clockwise or anticlockwise direction by the exchange of the SIN and COS expressions in each half of the PLOT routine. If the routine commences with the SIN function, PLOTting starts at the 12 o'clock position: if it commences with COS, PLOTting starts at the 3 o'clock position. If the variable angle is in the first half of the PLOT routine, ellipse major axis is vertical and, if it is in the second half, the major axis is horizontal.

Line 100 contains the FOR/NEXT f loop used for PLOTting the ellipse in the form:

FOR f = 0 TO PI*2 STEP I/s*3

where 0 to PI*2 produces a full ellipse. If the expression were to read:

FOR f = PI TO PI*3 STEP…

then a full ellipse would still be PLOTted, but commencing at 180°, or, on the opposite side, at the 6 o'clock or 9 o'clock locations respectively.

Try changing the values in the FOR/NEXT f loop to test this.

```
10 PRINT "Ellipse 4 @ tilt=";
30 LET x=132: LET y=76
31 LET x1=x*.6: LET y1=y*1.5
```

```
32 LET x2=x*1.2: LET y2=y*1.5
33 LET x3=x*.6: LET y3=y*.5
34 LET x4=x*1.2: LET y4=y*.5
40 LET r=35
50 INPUT "0CHR$130 to 90CHR$130 ",tilt
60 PRINT tilt;"CHR$130 ": PAPER 5
70 PRINT "clockwise","anticlockwise"
75 PRINT "axis 1","2"
77 PRINT AT 12,0;"^axis 3","4"
79 PAPER 7
80 LET angle=1/SIN ((.1+tilt)/180*PI)
90 INPUT "step (1 to 10)",s
100 FOR f=0 TO PI*2 STEP 1/s/3
111 PLOT x1+SIN f*r,y1+COS f*r/angle
112 PLOT x2+COS f*r,y2+SIN f*r/angle
113 PLOT x3+SIN f*r/angle,y3+COS f*r
114 PLOT x4+COS f*r/angle,y4+SIN f*r
120 NEXT f
```

Solid Ellipse

This routine uses the DRAW command to produce solid (filled in) ellipses. It actually executes the results about four times quicker than the previous ellipse routines, where the STEP intervals are sufficiently frequent to produce a continuous unbroken outline. This is because it is only necessary to compute one quadrant of an ellipse and to mirror the results in the remaining three quadrants sequentially. Different variable names have been assigned to this routine but the principles are precisely as before. You will notice that, as it is only necessary to compute one quadrant, the FOR/NEXT loop reads:

FOR f = 0 TO PI/2 STEP .012

only one quarter the length of… 0 TO PI*2 … used previously. The STEP interval is fixed at 0.012 as this was found, with this routine, to give the solid form required. If the ellipse is not to fill the Spectrum screen then a larger value can be tested.

The variables a and b are the x and y coordinate positions for the ellipse, where variable e provides the necessary compression to the vertical or minor axis. Line 130 PLOTs the left hand upper outline of the ellipse and Line 140 does the same for the right hand side. The DRAW routines that follow the PLOT commands in the form:

DRAW 0, – b*2

DRAW a vertical line equal to twice the value of b in a downwards direction, so completing the lower portion of the ellipse. Line 200 completes the routine by marking the major (horizontal) and minor (vertical) axes of the ellipse using the OVER command.

The ellipses are made to PLOT in a clockwise or anticlockwise direction by the exchange of the SIN and COS expressions in each half of the PLOT routine. If the routine commences with the SIN function, PLOTting starts at the 12 o'clock position: if it commences with COS, PLOTting starts at the 3 o'clock position. If the variable angle is in the first half of the PLOT routine, ellipse major axis is vertical and, if it is in the second half, the major axis is horizontal.

Line 100 contains the FOR/NEXT f loop used for PLOTting the ellipse in the form:

FOR f = 0 TO PI*2 STEP l/s*3

where 0 to PI*2 produces a full ellipse. If the expression were to read:

FOR f = PI TO PI*3 STEP…

then a full ellipse would still be PLOTted, but commencing at 180°, or, on the opposite side, at the 6 o'clock or 9 o'clock locations respectively. Try changing the values in the FOR/NEXT f loop to test this.

```
10 PRINT "Solid Ellipse"
20 INPUT "tilt ";z: PRINT "tilt=";z;"CHR$130 "
30 LET e=SIN ((.1+z)/180*PI)
40 LET x=255/2: LET y=175/2
100 FOR f=0 TO PI/2 STEP .012
110 LET a=INT (SIN f*y)
120 LET b=INT (COS f*y*e)
130 PLOT x-a,y+b: DRAW 0,-b*2
140 PLOT x+a,y+b: DRAW 0,-b*2
150 NEXT f
200 OVER 1: PLOT 0,y: DRAW 255,0: PLOT x,0: DRAW 0,175: OVER 0
```

Inclined Ellipse

There are times when an ellipse is required with the major axis, neither vertical nor horizontal but in an intermediate position. This final routine 'indicates a method of achieving this aim to a modest extent. Figure 9.9 shows a screen COPY from the program and typical INPUTs and ellipses as PLOTted. The INPUT values for tilt and inclination (rotation of the image) may be from -90° to 90° in each case. Line 90 does the actual work of computing both these factors in the final part of the expression:

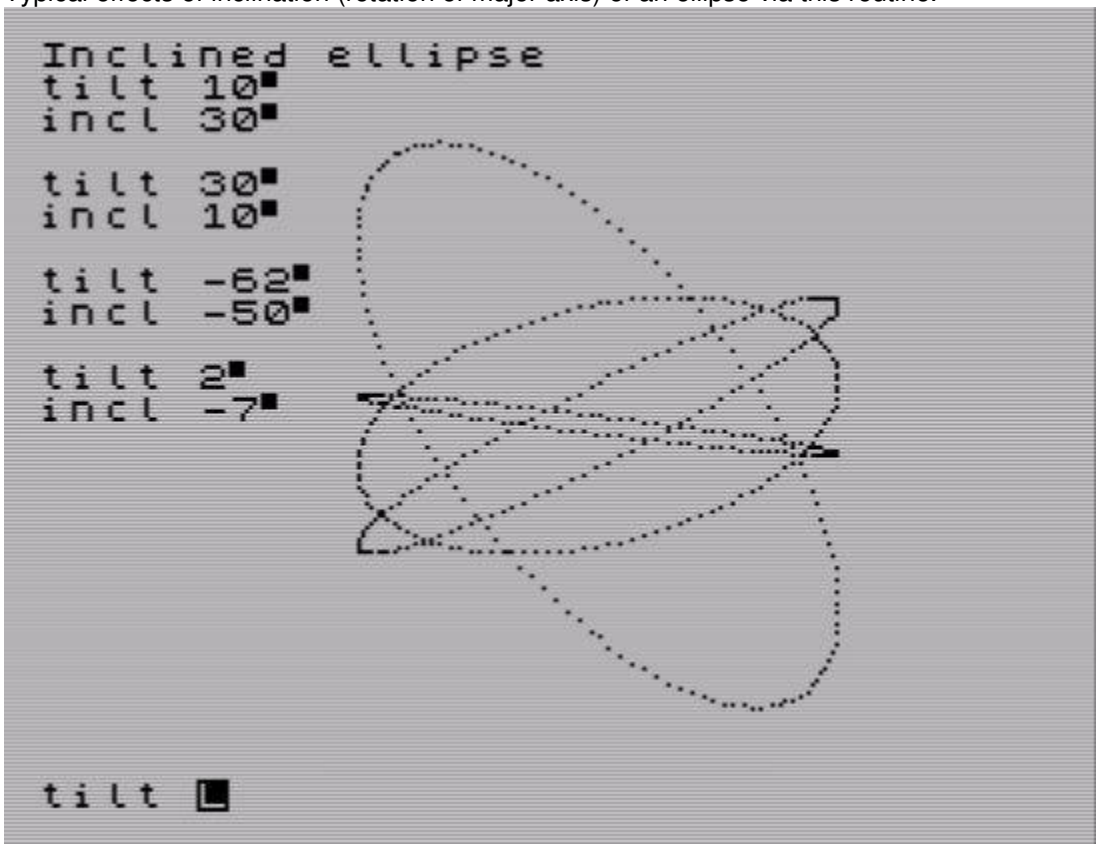
…/t + a/z

where variables t (for tilt) and z (for rotation) are the reciprocal SIN values as INPUT. The PLOTting interval (STEP 0.05) and maximum horizontal displacement (value 60 in Lines 80 and 90) are fixed but can be amended to suit the requirements.

The Ellipse routine is used frequently, sometimes in a modified form, for various programs in this book, including Saturn's Rings, Galaxy, Binary-star Orbits. The expression 'ellipse' is in common use to describe the shape of planetary and particular cometary orbits and the shape of optical (lens and mirror) surfaces. Ellipse is one of a family of forms in the conic section which includes the circle, parabola and hyperbola.

Figure 9.9

Typical effects of inclination (rotation of major axis) of an ellipse via this routine.




```
10 PRINT "Inclined ellipse"
20 INPUT "tilt ";t,"incline ";z
30 PRINT "tilt ";t;"CHR$130 "
40 PRINT "incl ";z;"CHR$130 ""
50 LET z=1/SIN ((.1+z)/180*PI)
60 LET t=1/SIN ((.1+t)/180*PI)
70 FOR n=0 TO PI*2 STEP .05
75 PLOT OVER 1;140,80
80 LET a=SIN n*60
```

```
90 LET b=COS n*60/t+a/z
100 PLOT INT (140+a),INT (80+b)
110 NEXT n: GO TO 20
```

Ellipses and Kepler's Orbits

One cautionary note should be observed in using the Ellipse routines to simulate a planet in true elliptical orbit about the Sun, perhaps — where the Sun cannot be in the centre of the ellipse but at one of the foci. Here Kepler's 2nd Law of Planetary Motion applies and the planet will 'speed-up' as it approaches the Sun and 'slow-down' as it recedes. The Ellipse program will not in this case give a perfect account of itself but on many occasions the effect can be reasonably convincing. For a correct simulation of Kepler's 2nd Law, see the Kepler's Orbits program in Chapter 5, and the sample screen COPYs.

Spectrum World Map

One of the problems of producing a picture on the TV screen from a program listing is trying to comprehend what graphics the author intended to be used. The Spectrum is capable of over 44,000 pixel positions (256*175) and it would be quite unreasonable to create a full screen image defining the x and y coordinates of all the pixels to be displayed. It is however quite reasonable to create a simple 'lo-res' screen image using the Spectrum chunky graphic set from CHR$ 128 to CHR$ 143 inclusive. This effectively gives a screen resolution of 64*44 pixels (c for chunky!) or 2816 in total.

Fortunately, by selecting the correct graphic characters for each four adjacent pixels a complete screen can be defined in 32*22 character spaces, 704 in total. Now it becomes manageable.

Coding the characters

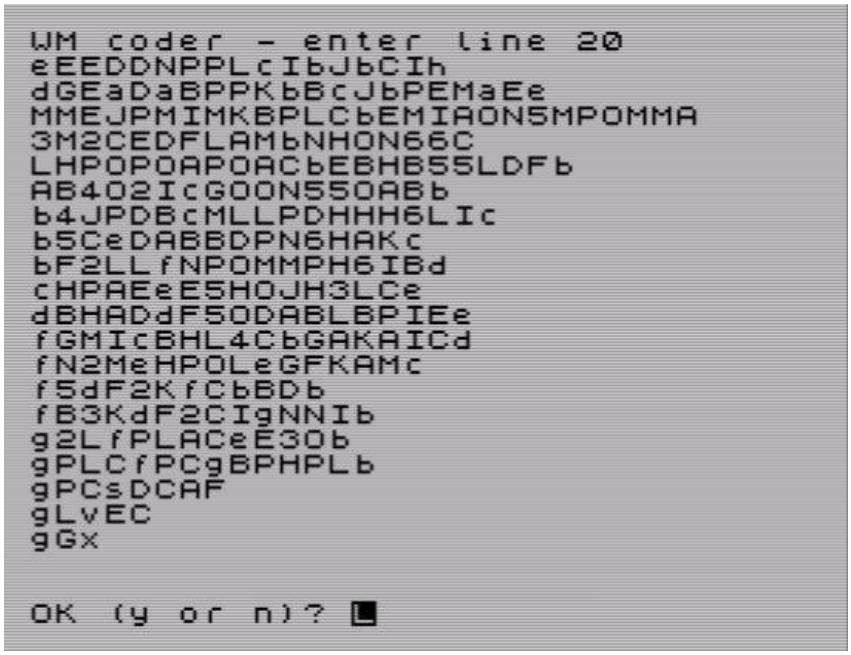
The simplest way to produce such a screen image is to list 22 consecutive PRINT statements, each containing up to 32 chunky graphic characters, and RUN them. Easy for the programmer but almost impossible for subsequent users to understand. The intended characters must be coded in a legible form that can be keyed in with certainty. The following program does just that and uses a world map as a demonstration screen display. Once the data has been entered, in a mixture of numbers and upper and lower case letters, the program automatically produces the finished picture via an intermediate coded form. This proves quite interesting to watch.

The program

Key in the program and RUN it. Then enter the lines of coded data line by line from Figure 9.10a. The program is reasonably error-trapped, giving a full opportunity to correct any entry, and is well prompted. Once all 20 lines are completed (two lines are omitted for titling, etc) the program converts the codes into the world map and displays the result recalled by PRINT T$

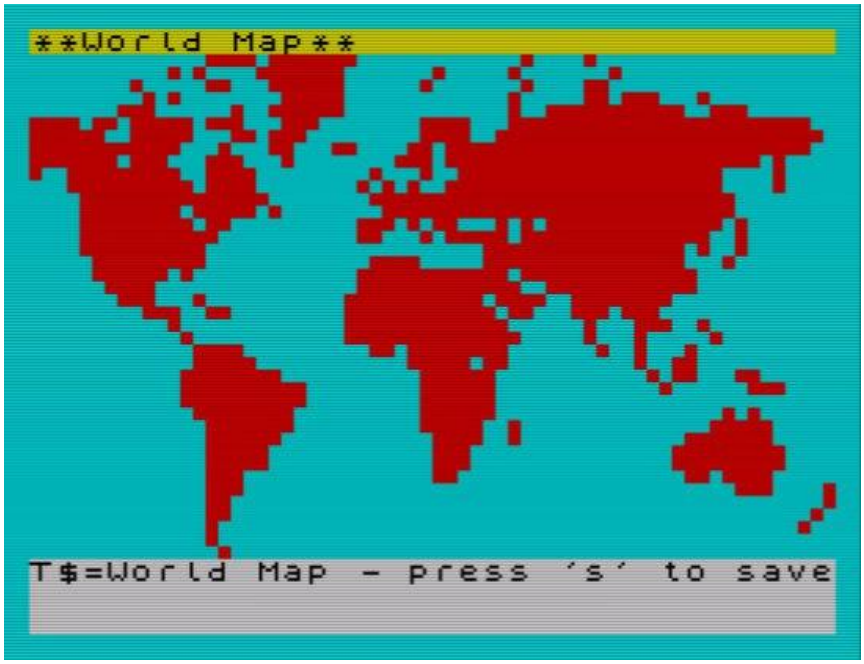
Figure 9.10: Picture Drawing in Code Form

a) These codes are entered into the program as prompted. The numerics refer to the number of consecutive solid squares — CHR$ 143. The lower case letters refer to the number of blank squares — CHR$ 128.



b) The computer converts the numeric and lower case codes from a) to upper case letters, Tilling each character square.

c) The computer converts the upper case letters in b) to the Spectrum chunky graphic set (CHR$ 128 to CHR$ 143 inclusive). The whole picture is stored in T$ for instant recall.



Saving the DATA

The actual picture is now stored in the DIMensioned T$ array ready to be SAVEd as prompted by the program, the DATA as SAVEd can be MERGEd into other programs as follows:

LOAD “world” DATA T$() ENTER

The picture is still called T$ so beware that this array name is not repeated in the new program: and, of course, the program must be started with GOTO (line number) and not RUN.

This particular program has now done its job and can in theory be discarded. But, before doing so, SAVE the complete program on to tape with GOTO 9990 for use with your own coded pictures.

Making your own coded pictures

The procedure to produce these coded pictures is tedious and is only worthwhile if they are to appear in published form. Page 92 of the Spectrum Manual lists the 16 chunky graphic set characters and these should be marked A through to P, starting at CHR$ 128 as A and finishing at CHR$ 143 as P. The artwork is now prepared on squared paper using page 102 of the Manual for guidance by overlaying tracing paper using the best shapes from the chunky graphic set. These are then converted into letters A to P inclusive as appropriate.

As the bulk of a simple picture is usually either blank or solid black (or any other INK colour) the long strings of AAAAAAAA or PPPPPPP can be compressed in each horizontal line to lower case letters or numerals respectively. Use Figure 9.10a for guidance. For example, the final line:

gGx means AAAAAAAGAAAAAAAAAAAAAAAAAAAAAA

where g equals the 7th letter of the alphabet and x the 24th.

Those with programming experience can get the Spectrum to do the conversion of chunky graphics to alphanumeric codes and so ease the task.

```
50 DIM e$(20,32)
60 DIM z$(20,32)
70 PRINT "WM coder – enter line"
80 FOR n=1 TO 20
90 PRINT AT 0,22;n
100 INPUT "Codes"' LINE e$(n)
110 PRINT AT n,0;e$(n)
120 INPUT "OK (y or n)? "; LINE q$: IF q$="n" THEN GO TO 100
130 NEXT n
140 PRINT PAPER 5;AT 0,0;"*World Map recoded* "
150 FOR n=1 TO 20: LET h$=""
160 FOR f=1 TO 32: LET x$=""
170 IF e$(n,f)>CHR$ 80 THEN GO TO 220
180 IF e$(n,f)<CHR$ 65 THEN GO TO 250
190 LET h$=h$+e$(n,f)
200 NEXT f: PRINT PAPER 6;h$: LET z$(n)=h$: NEXT n: GO TO 310
220 FOR x=1 TO CODE e$(n,f)-96
230 LET x$=x$+"A": GO TO 280
250 FOR x=1 TO CODE e$(n,f)-48
260 LET x$=x$+"P": GO TO 280
280 NEXT x: LET h$=h$+x$
290 GO TO 200
310 PRINT PAPER 6;AT 0,0;"**World Map** "
320 BORDER 5: LET w$=""
330 FOR n=1 TO 20
340 FOR f=1 TO 32
350 LET t=CODE z$(n,f)+63
360 PRINT INK 2; PAPER 5;CHR$ t;: LET w$=w$+CHR$ t
370 NEXT f: NEXT n: DIM t$(640): LET t$=w$
```

```
380 PRINT "T$=World Map - press 's' to save": PAUSE 0
400 SAVE "world" DATA t$()
405 BEEP 1,1
410 PRINT #0;"Rewind/play to verify": VERIFY "world" DATA t$()
430 BEEP 1,1: PRINT FLASH 1;"data saved OK": STOP
```

Chapter 10 – Spectrum Hints and Tips (Error-trapping of entries)

Every programmer acquires some working knowledge of the strengths and foibles of the computer system he uses and I have done this with the Spectrum. Because of computing's very nature, some of these discoveries tend to be repeated for many different people, so apologies to all who have read some of mine elsewhere or have found them out for themselves.

Error-trapping of entries

Generally any extensive error-trapping is omitted from this book for the sake of brevity. The following routines are offered as guidance — I leave it up to you to decide which part of your program(s) to place them in. Don't forget to renumber the lines.

1) Protecting numeric iNPUTs within upper and lower limits for DATE and TIME

```
100 INPUT "month no", month
110 IF INT month month OR month 12 THEN
GOTO 100
120 INPUT "day no", day
130 IF INT day day OR day 31 THEN GOTO 120
140 INPUT "hour (0 to 23)", hour
150 IF INT hour hour OR hour 23 THEN GOTO 140
160 INPUT "minute (0 to 59)", min
170 IF INT min min OR min 59 THEN GOTO 160
180 REM program continues INPUTs complete
```

The following routine is much superior in rejecting any INPUTs that are not composed entirely in numbers (Line 520) and within limits (Line 550).

2) Protecting INPUT to numeric only within upper and lower limits for DATE and TIME

```
200 LET a$ = "Year": LET a = 2000: LET b = 1834: GOSUB 500 : LET y = c
210 LET a$= "Month": LET a = 12: LET b= 1: GOSUB 500 : LET m = c
220 LET a$ = "Day": LET a = 31: GOSUB 500 : LET d = c
230 LET a$ = "Hour": LET a = 23: LET b = 0: GOSUB 500 : LET h = c
240 LET a$ = "Minute": LET a = 59: GOSUB 500 : LET mi = c
250 LET a$ = "Second": GOSUB 500 : LET s = c
260 REM program continues INPUTs complete 270 .....
500 INPUT (a$;"(";b;" to ";a;")";), LINE b$ 510 FOR x = 1 to LEN b$
```

```
520 IF CODE b$(x) 57 THEN GOTO 600
530 NEXT x
540 LET c = VAL b$
550 IF c> a OR c< b THEN GOTO 600
560 RETURN : REM INPUT checked OK
600 PRINT # 0 ; FLASH 1 ; "Entry error": PAUSE 100: GOTO 500
```

Note: The values of a and b in Line 200 may be for any selected years. Line 500 appears on the screen (using the Year INPUT as an example) as:

Year (1834 to 2000)

Line renumbering programs

If you go beyond the shortest routine (and even short routines have a habit of growing like Topsy!) a renumbering program can be a great time saver and will make your final listings more presentable. Various versions — preferably in machine code and stored above RAM — are available commercially on tape. It will be worthwhile, too, even if you choose to key in a program published in a magazine, to have GOTOs and GOSUBs automatically renumbered.

Conditional GOTOs or GOSUBs, like .. .GOTO sky… or.. .GOSUB k*10… may not be renumbered and must be done manually and in a methodical manner. For example, if your program contains:

… LET sky =1000

… GOTO sky

1000 PRINT "starchart"

then amend Line 1000 to read '1000 PRINT "starchart" : REM GOTO sky'. Once the program has been automatically renumbered, amend the variable 'sky' to the new line number:

… LET sky = (line number with "REM GOTO sky" at end of it)

In the absence of these REM statements as ' flags' the process can be a long chore.

ZX printer notes

Whilst developing a long program it can be exceptionally tedious to have to search for and compare ‘pages’ of program listing. Hardcopy via the ZX printer will help you retain your sanity! A complete LLISTing is often unnecessary — just a few lines printed for comparison with the screen page of interest with … LLIST n (for line number) and BREAK when sufficient, is printed.

Storing program listing from the ZX printer

The metalised ZX printer paper is very sensitive to finger marks and should always be handled by the edges. If you are preparing a listing for submission to a magazine, I advise you to take the precaution of washing your hands immediately before touching the surface — accidents will happen.

Two widths of ZX printer will fit side by side on an A4 sheet of paper. A minimum of 20 mm margins at the top and bottom of the page is preferable — the listing trimmed with a Stanley knife and straight edge. Be sure to cut precisely between the lines of print. Sellotape the listings, very lightly, face down on old newspaper and spray the backs with aerosol adhesive. Now carefully apply the listing to the A4 sheet using a light pressure. Get the listing photocopied for your own reference and store the original listing — unfolded of course — in a strong envelope until needed.

A cheap A4 photograph album with cellophane leaf overlays is an ideal place for storing short programs, routines and graphic printouts. This also keeps them safe from prying (and dangerous!) fingers. If you do not possess (or your household objects to the use of) picture rails to suspend long listings, then rolling listings in the discarded cores of toilet paper may suffice — the dimensions are just right to give the necessary protection.

If a long listing is sent through the postal system in rolled form it must be supported on a firm core — like the one at the centre of the ZX printer rolls.

Appendix – Magazines and Clubs

Magazines and Clubs

General Astronomical Magazines

The following publications deal exclusively with astronomy. They occasionally include articles on astrocomputing for the home micro.

Sky & Telescope
49 Bay State Road, Cambridge, Massachusetts 02238 – 1290 — monthly $2.00 plus postage, subscription only

Journal of the British Astronomical Association
Dept. SA, Burlington House, Piccadilly, London W1V ONL — bimonthly £2.75, free to members

Astronomical Clubs and Associations

British Astronomical Association
Dept. SA, Burlington House, Piccadilly, London W1V ONL

Junior Astronomical Society
Dept. SA, Mr V. L. Tibbott, 58 Vaughan Gardens, Ilford, Essex IG1 3PD

The Federation of Astronomical Societies
Dept. SA, Mrs R. Naylor, 24 Julia Crescent, Stonebroom, Derby DE5 6LS

American Association of Variable Star Observers
187 Concord Avenue, Cambridge, Massachusetts 02138

Association of Lunar and Planetary Observers
PO Box 3AZ, University Park, Les Cruces, New Mexico 88001

Any of the above organisations should be able to advise on affiliated local clubs.