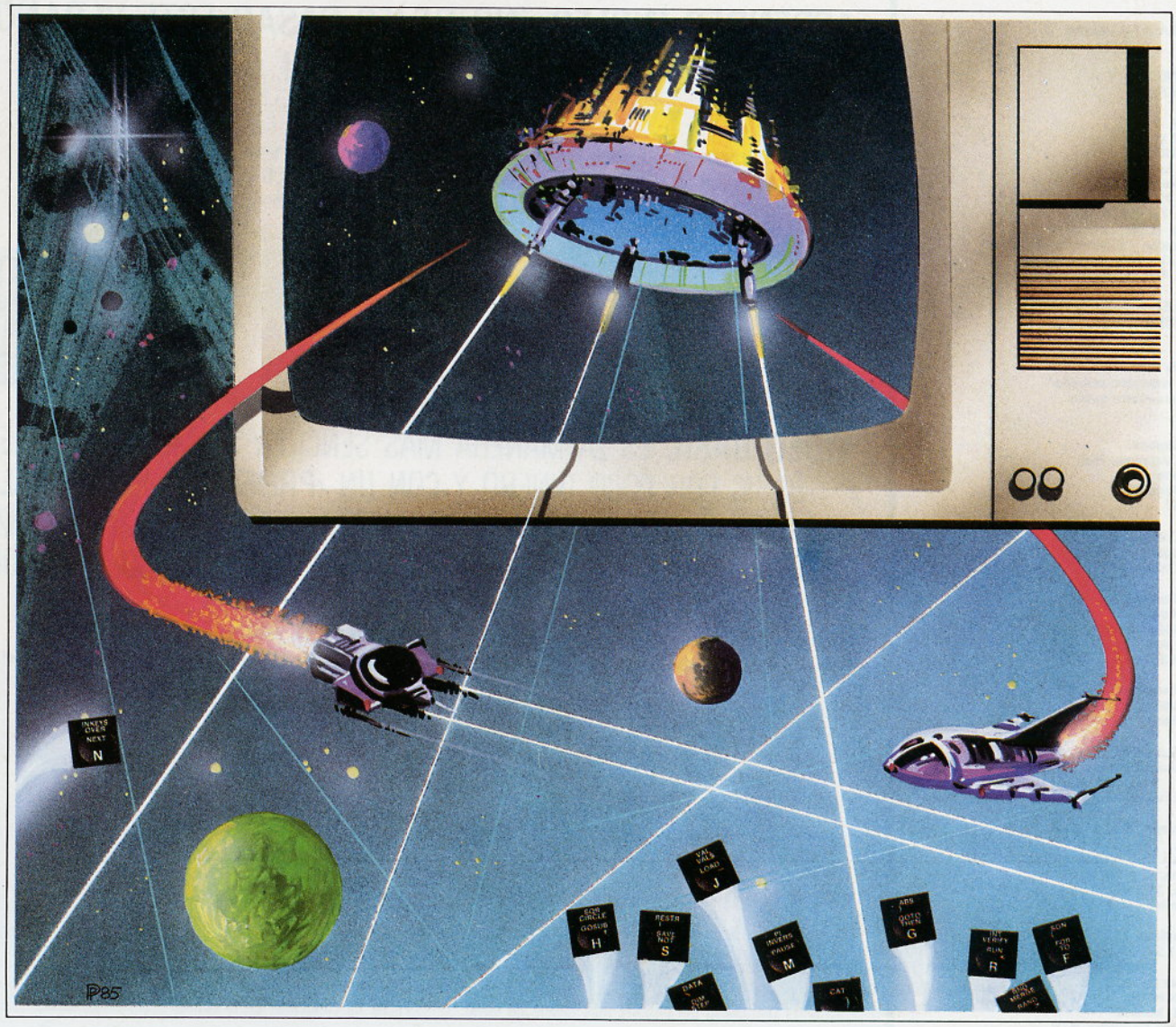
13
150pts.

RUN

Enciclopedia Práctica del Spectrum



Nueva Lente/Ingelek

FUNCIONES MATEMATICAS

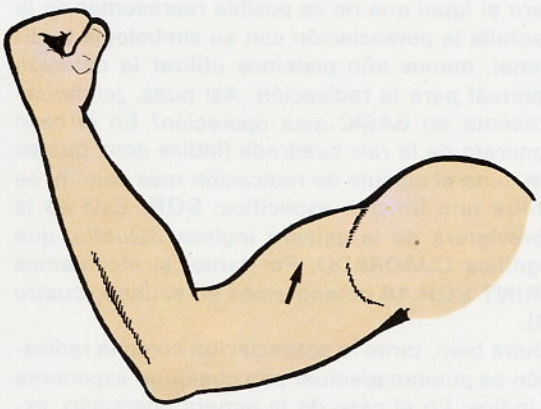
BAJO esta denominación general, trataremos las funciones de potenciación, exponenciación, logarítmicas y trigonométricas. La estructura de estas funciones es muy similar a las de tratamiento numérico que vimos anteriormente; no obstante, su uso es bastante menos frecuente, al restringirse su aplicación al campo científico o de cálculo matemático.

Por ello, no debemos asustarnos si en algún momento no llegamos a comprender exactamente las explicaciones sobre la utilización de estas funciones. Tengamos en cuenta que, si realmente no conocemos su uso, nunca las utilizaremos en nuestros propios programas. Por tanto, nos bastará con saber que existen, así como su sintaxis, para facilitarnos la comprensión e introducción de listados de programas, confeccionados por otras personas que sí las utilicen.

Así pues, los poco introducidos en el campo de las matemáticas, no deben preocuparse por no comprender completamente las explicaciones que a continuación se brindan. En todo caso, se ha procurado simplificar estas en lo posible, lo que puede dar lugar a algunas imprecisiones que no son dignas de ser tenidas en consideración.

POTENCIACION

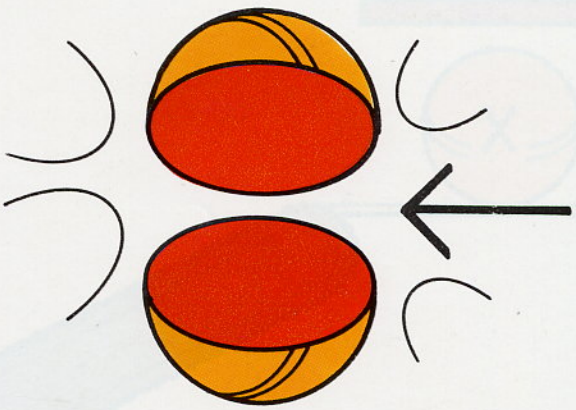
Es de todos conocido que la potenciación consiste en multiplicar un número, denominado base, por sí mismo, tantas veces como indica el superíndice que le sucede, que se denomina exponente. Así por ejemplo, el término 4^2, que se lee "cuatro elevado a dos" (o "cuatro al cuadrado"), tiene como resultado 4×4=16. Lógicamente, debido a la restricción de no poder utilizar la notación habitual de superíndices, en BASIC se adopta el criterio de representar esta operación mediante la base, seguida de un signo "flecha hacia arriba" (↑) y, a continuación, el exponente. Con lo cual, la operación del ejemplo anterior se escribiría en BASIC de la siguiente manera: 4 ↑ 2.



Para la obtención de la función de potenciación, en el Spectrum se utiliza un símbolo «flecha hacia arriba» (↑).

Dentro de las funciones matemáticas, probablemente sea esta la de uso más frecuente, sobre todo con el exponente dos. Daremos un breve repaso a nuestros conocimientos básicos sobre esta operación.

La potenciación, al igual que el resto de las operaciones matemáticas que hemos visto hasta el momento (suma, multiplicación, etc.), tiene una operación inversa la cual nos permite conocer qué base elevada a un exponente proporciona determinado resultado; esta operación se denomina radicación, y se representa por una uve (**V**) en cuya parte superior se expresa el índice de la raíz, unida a una línea de super-rayado debajo de la cual se escribe el radicando (argumento de la función).



La potenciación admite exponentes fraccionarios, mediante los cuales conseguir la función de radicación.

¡!

La función exponencial y logarítmica son inversas.

*

Para la obtención en el Spectrum de logaritmos en una base distinta de **e**, tenemos que utilizar una propiedad de los logaritmos, por la cual, el logaritmo en base **B** de **N**, es igual al cociente del logaritmo natural de **N**, partido por el logaritmo natural de la base **B**.

¡!

Los argumentos de las funciones trigonométricas deben expresarse obligatoriamente en radianes.

*

La potenciación consiste en multiplicar un número (base) por sí mismo, tantas veces como indica el superíndice (exponente) que le sigue.

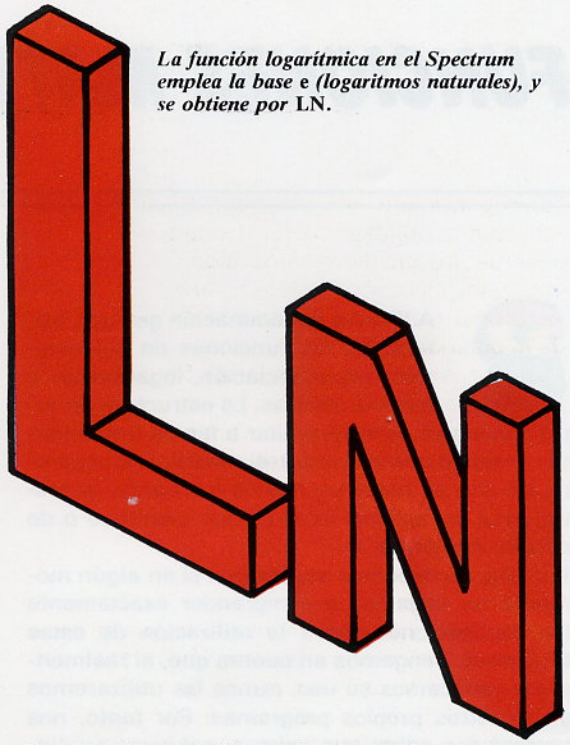
En el caso concreto del ejemplo anterior, la operación inversa al cuadrado de cuatro, es la raíz cuadrada de dieciséis, representada $\sqrt{16}$, cuyo resultado es 4. Así pues, se cumple que dada una base cualquiera **B**, y un exponente cualquiera E...

$$B^E = X \text{ y } B = \sqrt[E]{X}$$

Pero al igual que no es posible representar en la pantalla la potenciación con su simbología tradicional, menos aún podemos utilizar la notación habitual para la radicación. Así pues, ¿cómo representa en BASIC esta operación? En el caso concreto de la raíz cuadrada (índice dos), que es sin duda el cálculo de radicación más común, se utiliza una función específica: **SQR.** Esta es la abreviatura de la palabra inglesa *SQuaRe,* que significa CUADRADO. Por tanto, si efectuamos **PRINT SQR 16** obtendremos el resultado cuatro **(4).**

Ahora bien, tanto la potenciación como la radicación se pueden efectuar con cualquier exponente o índice. En el caso de la pimera operación, expresar un exponente distinto a dos, es bien sencillo: no hay más que cambiar el número escrito tras la flecha por el exponente que deseemos; sin embargo, en el caso de la radicación, la función BASIC **SQR** implica una raíz cuadrada, ¿cómo expresar entonces un índice distinto de dos? Para responder a esta pregunta es preciso disponer de una cierta base matemática.

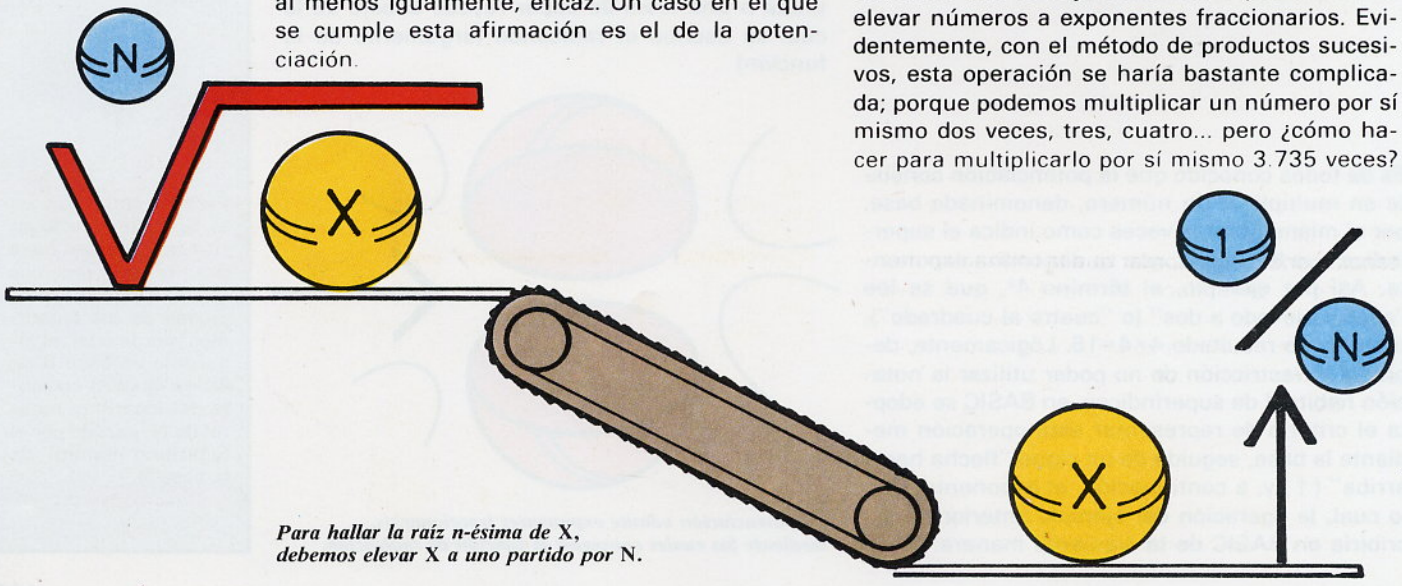
Los conocimientos de cálculo matemático de nuestro Spectrum son considerables, y prácticamente es seguro que, sin ánimo de menospreciarnos, superen los que nosotros tenemos. Por ello, la forma en que el ordenador realiza determinadas operaciones internamente, no se corresponde exactamente con la manera tradicional por la que las llevaríamos a cabo los profanos en la materia, aunque el sistema empleado sea más, o al menos igualmente, eficaz. Un caso en el que se cumple esta afirmación es el de la potenciación.



La función logarítmica en el Spectrum emplea la base e *(logaritmos naturales), y se obtiene por* **LN.**

Como ya hemos dicho, elevar una base a un exponente equivale a multiplicar dicha base por sí misma, tantas veces como indique el exponente; sin embargo, esta no es la forma en la que el Spectrum lleva a cabo dicha operación. Nuestro ordenador se basa en su capacidad para realizar la operación matemática denominada LOGARITMO, que dentro de muy poco pasaremos a estudiar. Más concretamente, el sistema de potenciación se apoya en una propiedad de los logaritmos, por la cual la base **B** elevada al exponente **E,** es equivalente al antilogaritmo del exponente **E** multiplicado por el logaritmo de la base. Esto se expresa en BASIC de la siguiente manera: **B ↑ E=EXP (E*LN B).**

Seguir este sistema para el cálculo de la potencia, tiene una ventaja clarísima: la posibilidad de elevar números a exponentes fraccionarios. Evidentemente, con el método de productos sucesivos, esta operación se haría bastante complicada; porque podemos multiplicar un número por sí mismo dos veces, tres, cuatro... pero ¿cómo hacer para multiplicarlo por sí mismo 3.735 veces?



Para hallar la raíz n-ésima de X, *debemos elevar* X *a uno partido por* N.

Una buena regla mnemotécnica para recordar el valor de **e**, *la proporciona el hecho de que su aproximación coincida con 2.7, seguido del año de la muerte de Goya (1.828).*

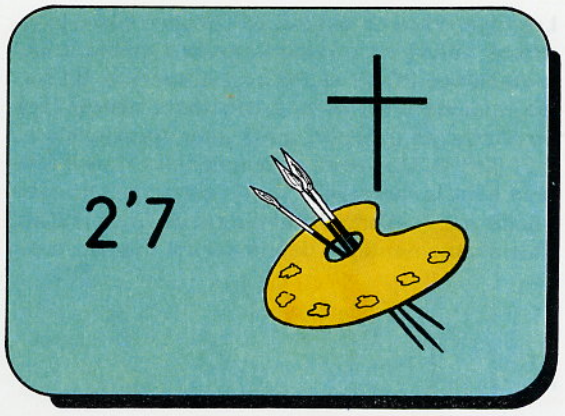
La pregunta que se plantea inmediatamente, se refiere a la utilidad de elevar una base a un exponente fraccionario; pues bien, es mucha, y pronto lo comprobaremos, puesto que nos permite obtener las raíces de cualquier índice a través de la potenciación.
El hecho antes mencionado, se fundamenta en una propiedad de las potencias, por la cual una base cualquiera **B**, elevada a un exponente fraccionario **N/D**, es igual a la raíz de índice **D** de la base **B** elevada a **N**, o lo que es lo mismo:

$$B^{N/D} = \sqrt[D]{B^N}$$

Por tanto, puesto que cualquier base elevada a exponente 1 tiene por resultado la misma base, podemos particularizar la expresión anterior, asegurando que la raíz n-esima de cualquier número es igual al número elevado a uno partido por el índice. Esto quiere decir, que la obtención de la raíz cuadrada se puede llevar a cabo de dos formas diferentes: una por medio de la función BASIC específica **(SQR)** y otra, elevando la base a un medio **(1/2)**. Gracias a este sistema, podemos obtener cualquier raíz; así por ejemplo, la raíz de índice tres de 343 se puede averiguar mediante la siguiente instrucción: **PRINT 343↑(1/3)**.
Para finalizar con el tema de las potencias, y puesto que hemos visto como obtener las radicaciones mediante esta operación, vamos a enumerar algunas propiedades interesantes, que no están directamente relacionadas con el BASIC, pero que nos pueden ser de utilidad a la hora de realizar algún programa que precise esta operación:

B↑0 = 1
B↑(-E) = (1/B)↑E
B↑(E+F) = B↑E * B↑F
B↑(E*F) = B↑E↑F

En cuanto a la prioridad en el orden de ejecución, la potenciación tiene la más alta entre las operaciones matemáticas elementales. Otra precaución a tomar para la correcta ejecución de las potenciaciones, es recordar que la utilización de bases negativas produce un error del tipo **A Invalid argument**, al igual que ocurre con la función **SQR**, como tuvimos oportunidad de comprobar en el capítulo anterior cuando estudiamos la función **SGN**.



FUNCIONES LOGARITMICAS

Comenzaremos este epígrafe definiendo la función que trata: el LOGARITMO en cualquier base **B** de un número **N**, es el exponente al que se ha de elevar **B** para obtener **N**. Se trata pues de una de las operaciones inversas a la potenciación; por tanto, hemos de tener cuidado de no intentar efectuar el logaritmo de un número negativo, para no obtener de nuevo el desagradable mensaje de **A Invalid argument** (argumento erróneo). Los logaritmos, como hemos visto, se pueden expresar en cualquier base, pero sin duda son dos las más ampliamente utilizadas en el cálculo matemático: la base 10 y la base **e**. Los de la base 10, son también conocidos bajo el nombre de logaritmos decimales, vulgares o de BRIGGS, por ser este matemático el que confeccionó la primera tabla de logaritmos de este tipo.

La inversa de la función logarítmica es la función exponencial, obtenida mediante **EXP**.

¡!

La función de potenciación (↑), permite utilizar exponentes fraccionarios, gracias a lo cual facilita la obtención de raíces enésimas.

*

Algunas de las propiedades de las potencias, que conviene recordar son: **A↑0=1**, **A↑(-B)=(1/A)↑B**, **A↑(B+C)=A↑B*↑A↑C**, **A↑(B*C)=A↑B↑C**.

*

La fórmula general para la obtención de la raíz n-ésima de **B** es: **B↑(1/N)**.

¡!

Los logaritmos obtenidos directamente por el Spectrum son los del tipo neperiano o natural (base **e**).

*

El Spectrum incorpora la constante **PI** internamente, con un valor aproximado 3.1415927.

*

La función exponencial (antilogaritmo) se obtiene en el Spectrum mediante **EXP**.

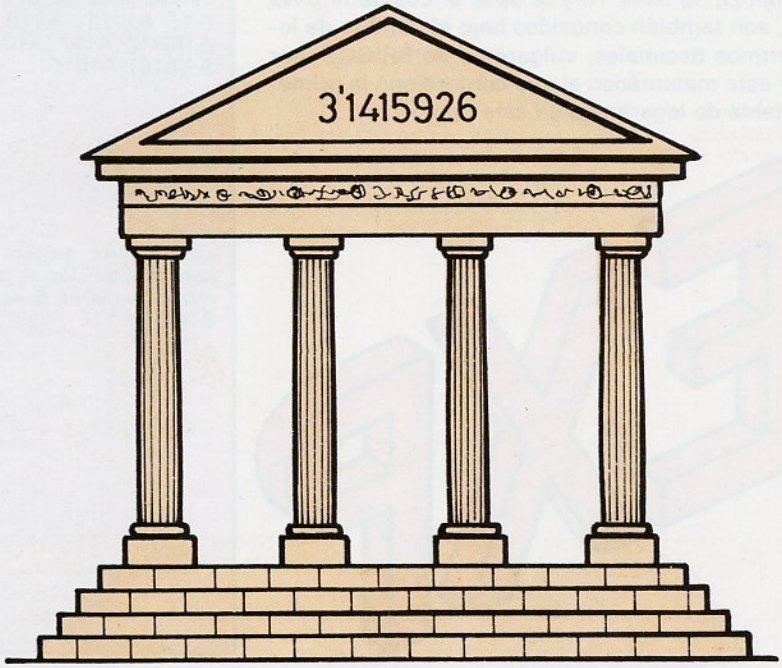
Los logaritmos en base **e**, se conocen como logaritmos naturales o neperianos, en memoria del matemático JOHN NEPER. El número **e**, de forma análoga a **PI**, es una constante matemática ampliamente utilizada, cuyo valor aproximado es 2.7182818. Una regla mnemotécnica muy difundida para la memorización de esta constante, es añadir a 2.7, el año de la muerte de Goya (1828). Más exactamente, la definición del número **e** es:

$$e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$$

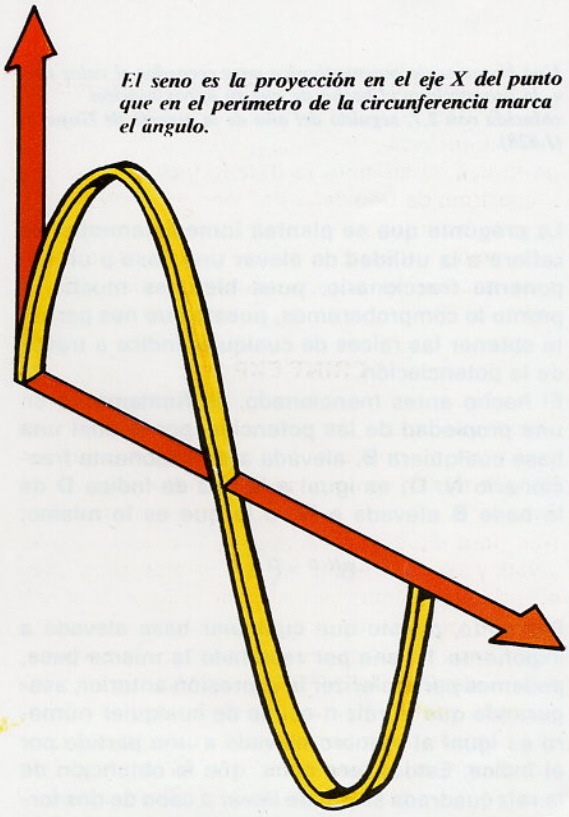
Ahora bien, el punto que más nos interesa en esta sección, es la forma en que nuestro Spectrum es capaz de llevar a cabo esta operación; en este sentido, hemos de hacer observar que el ordenador únicamente es capaz de hallar directamente logaritmos neperianos, que se obtienen mediante la función BASIC **LN**. Este tipo de logaritmos son empleados bastante frecuentemente en los cálculos matemáticos de una cierta complejidad; sin embargo, para el usuario con menos aspiraciones no reportan una gran utilidad.

Seguramente, sólo deseamos saber, a qué exponente hay que elevar 3 para que dé el resultado 243, es decir, obtener el logaritmo en base 3 de 243. No debemos preocuparnos por este pequeño detalle; podremos obtener logaritmos en cualquier base, apoyándose en la siguiente propiedad de los logaritmos:

El valor de la constante PI (π) se conoce ya desde la Antigua Grecia.



El seno es la proyección en el eje X del punto que en el perímetro de la circunferencia marca el ángulo.



Dadas dos bases **B** y **B'**, el logaritmo en base **B'** de cualquier número **(N)**, es igual al logaritmo de **N** en base **B**, partido por el logaritmo de **B'** en base **B**:

$$\text{Log}_{B'}\ N = \frac{\text{Log}_B\ N}{\text{Log}_B\ B'}$$

Según esta propiedad, conocido el logaritmo de un número en cualquier base, en nuestro caso **e**, podemos saber el logaritmo en cualquier otra, simplemente dividiendo el resultado del logaritmo en la base conocida, por el logaritmo de la base en que se desea conocer. Si aplicamos esta regla a nuestro ejemplo, tendremos que:

$$\text{Log}_3\ 243 = \frac{\text{LN } 243}{\text{LN } 3} \Rightarrow \textbf{PRINT LN 243/LN 3}$$

Aún nos queda por hablar de un par de propiedades más de los logaritmos, que quizás nos sean de utilidad en la simplificación de algunos cálculos con estas operaciones: el logaritmo de una potencia es igual al exponente multiplicado por el logaritmo de la base de la potencia **(Log B ↑ E = E * Log B)**; el logaritmo de la base logarítmica es siempre la unidad **(Log_N N = 1)**; y el logaritmo de uno es cero **(Log_N 1=0)**.

Finalmente, nos resta hablar de la función **(EXP)**, que realiza la operación inversa al logaritmo (antilogaritmo). Gracias a ella, podremos obtener el número cuyo logaritmo produce determinado resultado. Utilizando esta función, podemos escri-

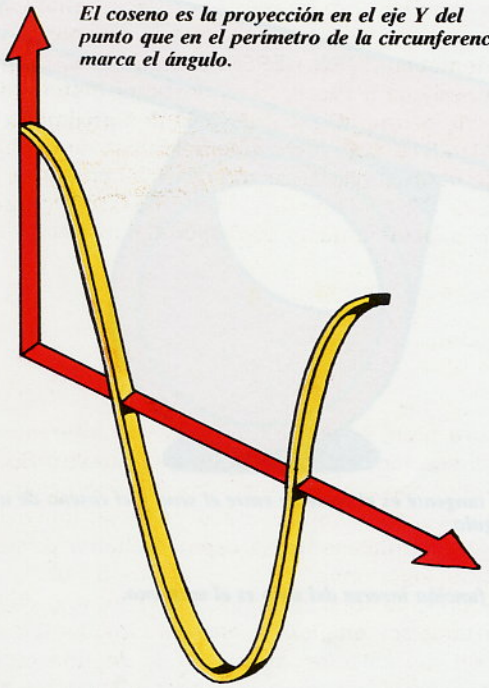
bir la aproximación del número **e** que el ordenador emplea en sus cálculos: según la propiedad vista anteriormente, el logaritmo de la base logarítmica es siempre la unidad, por tanto el antilogaritmo de uno debe dar siempre como resultado la base logarítmica; así pues, para obtener **e**, base de los logaritmos neperianos, sólo debemos escribir el antilogaritmo neperiano de la unidad:

PRINT EXP 1

Como es lógico, los antilogaritmos que en el Spectrum se pueden obtener directamente, son los que toman como base el número **e**.
Para finalizar con las funciones logarítmicas, y sólo para aquellos que tengan la especial curiosidad, y los suficientes conocimientos, para comprender el sistema por el cual el Spectrum calcula los logaritmos, diremos que una parte de su ROM, más concretamente dentro de la zona denominada CALCULADOR, incorpora una rutina que genera la denominada serie de polinomios de Chebyshev, gracias a lo cual, calcula las aproximaciones a las funciones **SIN, ATN, LN** y **EXP**, y por tanto a sus derivadas **(COS, TAN, ASN,** ↑ y **SQR)**.

FUNCIONES TRIGONOMETRICAS

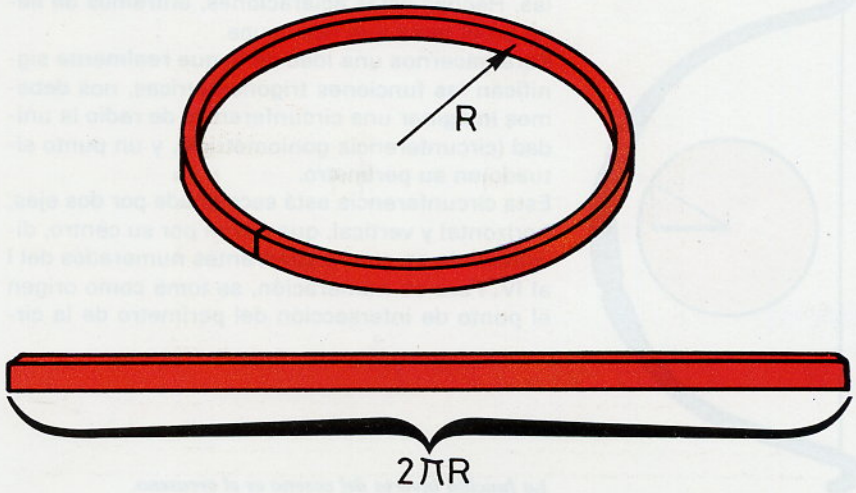
En este apartado, vamos a tratar tres funciones y sus correspondientes inversas. Los nombres de las funciones son: **SIN, COS, TAN** y **ASN, ACS, ATN;** estas se corresponden con SENO, COSENO, TANGENTE y ARCOSENO, ARCOCOSENO, ARCOTANGENTE. En todos los casos, la unidad angular que se ha de utilizar para expresar el argumento de las tres primeras funciones (no de sus inversas), es el RADIAN: arco cuya longitud es igual al radio de la circunferencia; por tanto, los ángulos posibles en una circunferencia, expresados en radianes, oscilan entre 0 y 2π. Como la mayoría de nosotros sabremos, PI es la constante matemática que relaciona la longitud de una circunferencia con su diámetro, y tiene el valor aproximado 3.1415927. Manejando RADIANES, se hacen muy frecuentes referencias a este valor, y el Spectrum lo tiene almacenado como una constante interna, de forma que puede ser empleada por el usuario como si se tratase de una variable de nombre PI, cuyo valor está predefinido y no es posible alterar. Así, por ejemplo, si deseamos conocer el valor que esta constante



El coseno es la proyección en el eje Y del punto que en el perímetro de la circunferencia marca el ángulo.

tiene en el ordenador, podemos escribir: **PRINT PI.**
Debemos tener mucho cuidado, para no confundir en el Spectrum la constante **PI** con una posible variable numérica PI; a efectos del ordenador, son absolutamente diferentes. La constante **PI,** se obtiene pulsando la tecla **M** en el modo extendido, y corresponde al carácter 167 en el código del Spectrum, mientras que la posible variable PI, se introduce carácter a carácter, y es equivalente a pi, Pi o pl.
Si al encender el ordenador, le pedimos que nos diga el valor de la variable PI, **(PRINT PI)** veremos que nos contesta con el error **2 Variable not**

La constante PI es la relación entre la longitud de una circunferencia y su diámetro; su valor aproximado es 3.1415927.

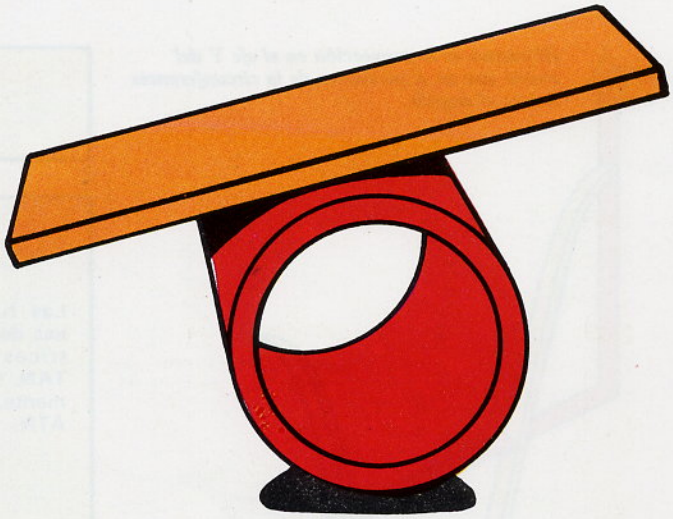


¡!

Las funciones inversas de las trigonometricas **SIN, COS** y **TAN,** son, respectivamente, **ASN, ACS** y **ATN.**

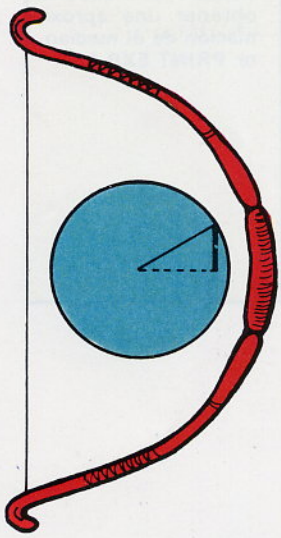
*

La base de los logaritmos neperianos es el número **e**. Podemos obtener una aproximación de él mediante: **PRINT EXP 1.**



La tangente es el cociente entre el seno y el coseno de un ángulo.

La función inversa del seno es el arcoseno.



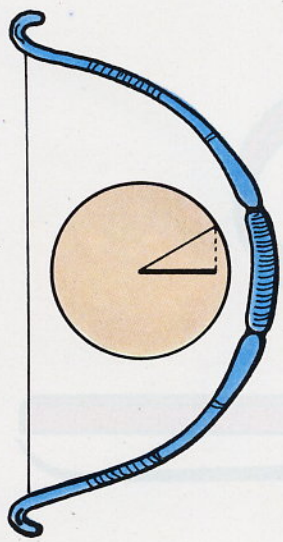
found (variable no encontrada), mientras que si ejecutamos **PRINT PI,** siendo **PI** la constante (**EXTENDED MODE** y **M**), obtendremos el valor que ya conocemos.

Del mismo modo, si intentamos asignar un valor a la posible variable PI, observaremos que la tarea se lleva a cabo sin ninguna complicación, pero los valores obtenidos a continuación con **PRINT** PI (variable) y **PRINT PI** (constante), serán absolutamente diferentes (¡a no ser que hayamos ejecutado **LET** PI=**PI!**). Por otra parte, cualquier intento por alterar el valor de dicha constante, ya sea mediante **LET, INPUT** o cualquier otro sistema, será rechazado por el *syntax checker.*

Una buena idea, para evitar posibles confusiones en los listados, entre la variable PI y la constante **PI,** es escribir la variable siempre con minúsculas. Hechas estas aclaraciones, entremos de lleno en el tema que nos ocupa.

Para hacernos una idea de lo que realmente significan las funciones trigonométricas, nos debemos imaginar una circunferencia de radio la unidad (circunferencia goniométrica), y un punto situado en su perímetro.

Esta circunferencia está seccionada por dos ejes, horizontal y vertical, que pasan por su centro, dividiéndola en cuatro cuadrantes numerados del I al IV. Para su numeración, se toma como origen el punto de intersección del perímetro de la circunferencia con el extremo derecho del eje horizontal, considerando por convenio el sentido de desplazamiento positivo el antihorario, es decir, el contrario al de las agujas del reloj.



La función inversa del coseno es el arcoseno.

Del desplazamiento de este punto por el perímetro de la circunferencia, se deducen una serie de relaciones entre el ángulo descrito en el movimiento y las proyecciones sobre los ejes horizontal y vertical. Las proyecciones sobre el diámetro horizontal o eje X, corresponden a la representación de la función coseno, y las proyecciones sobre el diámetro vertical o eje Y, las del seno. Los valores del seno y del coseno en el primer cuadrante son positivos; en el segundo, el seno es positivo, mientras que el coseno es negativo; en el tercero, tanto el seno como el coseno resultan negativos; por último, en el cuarto cuadrante, el seno resulta negativo y el coseno positivo, con lo cual, a lo largo de la circunferencia, se producen todas las combinaciones posibles.

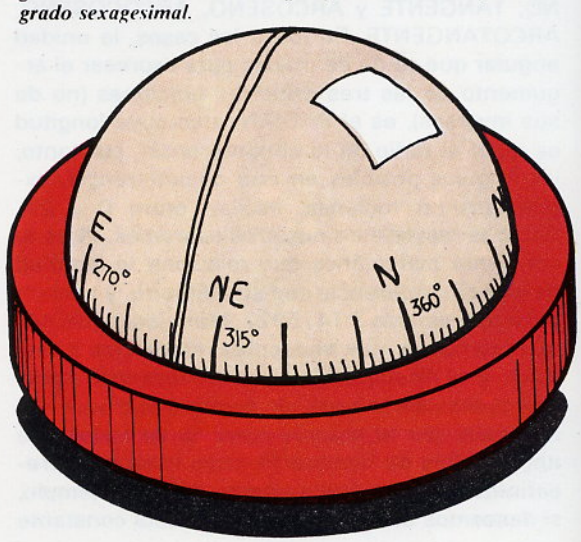
Otra propiedad interesante de las funciones angulares, que se nos hace evidente dada la estructura circular en que el imaginario punto del perímetro se desplaza, es que los valores de las funciones se repiten una vez cumplida una vuelta completa; por tanto, podemos afirmar que:

SIN X=SIN (X+N*PI) y COS X=COS (X+N*PI)

Siendo **X** el ángulo descrito, y **N** cualquier número entero múltiplo de dos, que expresa el doble del número de «vueltas» dadas a la circunferencia.

Existe una tercera función trigonométrica básica, que se obtiene como la relación existente entre el seno y el coseno de un ángulo. A esta relación se la conoce con el nombre de tangente, y pode-

Las dos unidades más frecuentemente utilizadas en la goniometría (medición de ángulos), son el radián y el grado sexagesimal.

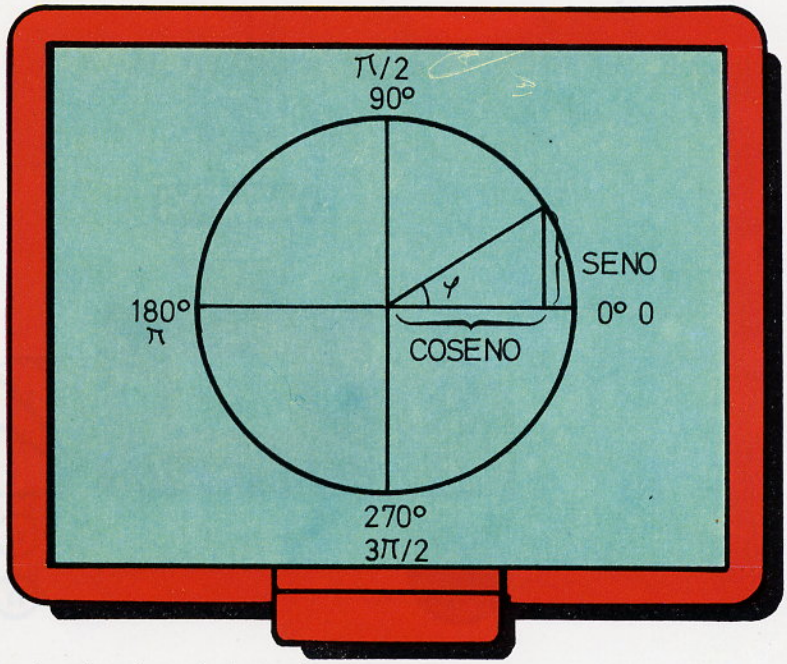


mos definirla como: **TAN X=SIN X/COS X.** Tanto la función seno como coseno, oscilan siempre entre los valores 0 y 1. Esto resulta obvio, si pensamos en que el valor mínimo que pueden adquirir es cero, como es el caso del seno de 0 o del coseno de $\pi/2$; y el valor máximo posible es 1, puesto que el radio de la circunferencia es la unidad, como es el caso del seno de $\pi/2$ o el coseno de 0.

De lo dicho anteriormente, podemos deducir que cualquier valor tendrá sentido como argumento de las funciones **SIN** y **COS,** pero no sucederá así con los argumentos de la función **TAN.** Dado que la función tangente es el cociente de las funciones seno y coseno, debemos preocuparnos de no emplear argumentos que al hacer cero el denominador, puedan llevarnos a un error del tipo **6 Number too big** (número demasiado grande). Esto se produce, lógicamente, con aquellos valores angulares cuyo coseno sea cero, es decir $\pi/2$ y $3/2\pi$.

Por otro lado, podemos disponer de las funciones inversas a las anteriormente citadas, que son el arcoseno **(ASN),** arcocoseno **(ACS),** y arcotangente **(ATN).** Estas funciones nos permiten obtener, a partir del valor de un seno, un coseno o una tangente, respectivamente, el ángulo (arco) del cual provienen.

Esto implica que los argumentos de las funciones trigonométricas inversas, deberán estar siempre comprendidos entre 0 y 1, puesto que, como hemos visto anteriormente, los valores entre los cuales se mueven los resultados de las funciones **SIN, COS** y **TAN** son éstos. Si pese a esta advertencia, intentamos efectuar una función inversa de este tipo, con argumento no comprendido entre 0 y 1, nos las veremos cara a cara con el mensaje **A Invalid argument** (argumento erróneo).



Una circunferencia goniométrica (de radio uno), nos facilita la labor de representar las relaciones trigonométricas.

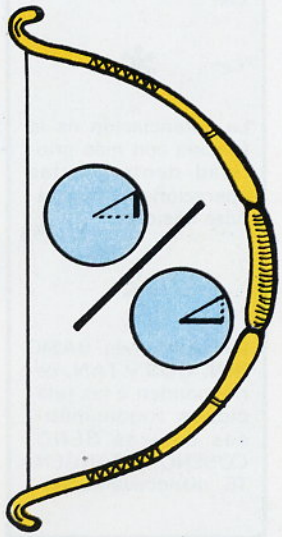
UNIDADES ANGULARES

Cuando comenzamos a hablar de las funciones trigonométricas, aseguramos que la unidad en que se miden las magnitudes angulares es el radián; sin embargo, si bien es cierto que esta es la unidad que más se maneja en medios técnicos, y que emplea el ordenador, la mayoría de nosotros utilizamos a tal fin el llamado sistema SEXAGESIMAL, en el cual la unidad es el grado sexagesimal, y sus submúltiplos: el minuto (60 minutos=1 grado) y el segundo (60 segundos=1 minuto).

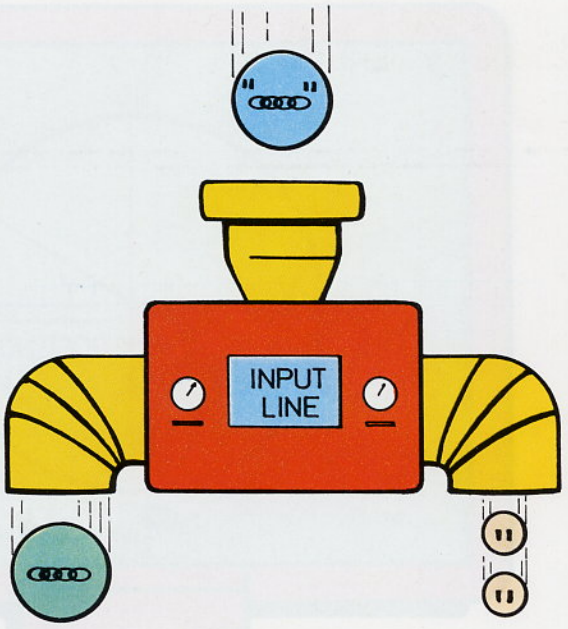
La conversión de un sistema a otro es muy sencilla. Sólo nos hace falta saber que el sistema sexagesimal divide la circunferencia en 360 grados; por tanto, existe una relación de 360 grados por cada 2 radianes, o lo que es lo mismo, π radianes equivalen a 180 grados. Así pues, para pasar radianes a grados, sólo hemos de multiplicarlos por 180 y dividirlos por π; y para la operación contraria, es decir, convertir grados en radianes, multiplicar los grados por π y dividirlos por 180. El siguiente programa nos muestra como efectuar la conversión de una forma práctica:

```
10 REM CONVERSION ANGULAR - J.M.
LOPEZ MARTINEZ
20 PRINT "GRADOS","RADIANES"
30 FOR I=0 TO 31:PRINT "-";:NEXT I
40 INPUT "Angulo:";A,LINE U$
50 IF U$<>"G" AND U$<>"R" THEN
GO TO 40
60 IF U$="R" THEN GO TO 90
70 IF A<-360 OR A>360 THEN GO TO 40
80 PRINT A,A/180*PI: GO TO 40
90 IF A<-2*PI OR A>2*PI THEN GO TO 40
100 PRINT A/PI*180,A: GO TO 40
```

Para el manejo del programa, sólo hemos de introducir el ángulo a convertir (seguido de **ENTER**)



La función inversa de la tangente es el arcotangente.



INPUT LINE *suprime las comillas que acompañan la entrada de variables de cadena.*

y a continuación la unidad en que se encuentra; para ello, utilizaremos una **R** cuando sean radianes, y una **G** cuando sean grados sexagesimales. A este respecto, conviene advertir que dichas letras debemos entrarlas en mayúsculas, para que sean reconocidas por el programa (líneas 50 y 60).

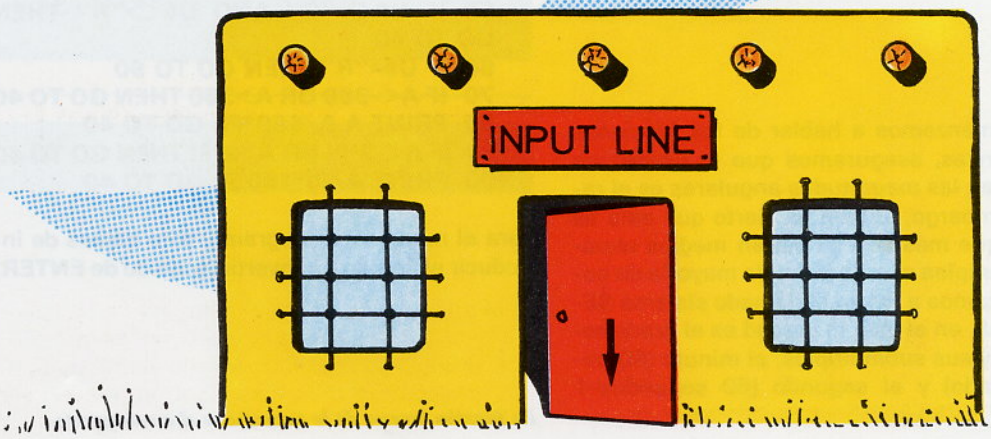
En lo referente al programa, en su mayoría se utilizan elementos que ya conocemos: la línea 10 contiene el **REM** de nombre y autoría, la 20 y 30 realizan la impresión de una cabecera (GRADOS RADIANES) subrayada con guiones, las líneas 70 y 90 efectúan depuraciones de los valores introducidos y, finalmente, las líneas 80 y 100, llevan a cabo el cálculo e impresión de los resultados. Ahora bien, intencionadamente hemos dejado para el final la instrucción 40, puesto que en ella aplicamos a la entrada de datos **(INPUT)** algunos conceptos nuevos, que conviene analizar más detenidamente. En primer lugar, utilizaremos un **INPUT** en el que se visualizará el literal especificado, pero en vez de una sola variable, se producirá la entrada de dos; una numérica **(A),** que indica la magnitud del ángulo a convertir, y otra de cadena **(U$),** que portará la unidad de medida angular (GRADOS=G, RADIANES=R).

Observemos que la primera variable del **INPUT** será solicitada inmediatamente a continuación del mensaje de petición (para designar este tipo de mensajes se utiliza la palabra inglesa *PROMPT),* mientras que la segunda lo hará en la columna dieciséis, debido al separador coma **(,)** que se ha introducido entre las dos variables. Para la entrada de la segunda variable **(U$),** del tipo cadena, hemos utilizado el formato **INPUT LINE,** cuyo objetivo es la supresión de las comillas que en los **INPUT** acompañan a los datos de este tipo. De esto deducimos, que **INPUT LINE** sólo puede ser empleado con variables de cadena, y que tiene una finalidad meramente estética.

Por último, conviene recordar que para abandonar los **INPUT** de variables numéricas, empleábamos el comando **STOP,** y para los de variables de cadena, previamente a la realización de esta operación, procedíamos a la eliminación de las comillas, que señalan este tipo de datos, bien mediante **DELETE,** o bien mediante **EDIT.** Sin embargo, en el caso del **INPUT LINE** ninguno de los dos sistemas funcionará, puesto que ni la variable a entrar es numérica, ni aparecen comillas que poder suprimir.

Para detener los programas durante la ejecución de este tercer tipo de **INPUT,** deberemos pulsar la tecla CURSOR ABAJO **(CAPS SHIFT** + 6), lo que producirá la generación de un mensaje del tipo **H STOP in INPUT.**

***Para producir la detención de un programa durante la ejecución de un* INPUT LINE, *es necesario pulsar CURSOR ABAJO* (CAPS SHIFT + 6).**



¡!

La función logarítmica (aplicada a neperianos) se obtiene en el Spectrum mediante **LN.**

*

La potenciación es la función con más prioridad dentro de las operaciones matemáticas básicas.

*

Las funciones BASIC **SIN, COS** y **TAN,** corresponden a las relaciones trigonométricas básicas SENO, COSENO y TANGENTE, respectivamente.

DINAMISMO DE PANTALLAS

COMO ya mencionamos anteriormente, la pantalla es el medio fundamental de comunicación con el usuario y, por tanto, a la hora de la realización de un programa, hemos de hacer hincapié en la importancia de la misma. Como vimos anteriormente, el tratamiento de la pantalla se apoya en tres puntos fundamentales:

— Diseño de la pantalla.
— Utilización del color.
— Dinamismo.

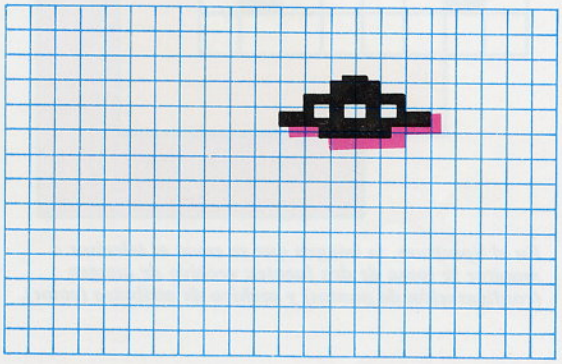
Pero ¿qué es el dinamismo? Caeremos en la tentación del chiste fácil, y diremos que no es sólo importante cuidar la estética de la pantalla, sino también su «estática»; esto debe pasar de ser un mero juego de palabras y convertirse en una máxima que nos acompañe durante todas nuestras tareas de programación.
El ordenador, al contrario que otros vehículos de información, como el cartel, la fotografía o el libro, dispone de la gran ventaja de ser dinámico, es decir, en la mano del programador está el alterar, a su gusto, el contenido informativo, y como no, la forma en que éste se presenta. Desaprovechar esta cualidad del ordenador, sería un error imperdonable. Encendamos el ordenador y veamos en un ejemplo el efecto que el dinamismo puede imprimir a la simple presentación de un mensaje:

```
10 PRINT AT 11,7;"PULSA UNA TECLA"
20 IF INKEY$="" THEN GO TO 20
30 CLS
40 PRINT AT 11,7;OVER 1;"PULSA UNA
   TECLA"
50 BEEP .5,60
60 IF INKEY$="" THEN GO TO 40
```

Evidentemente, el objetivo de la presentación del mensaje (transmitir al usuario una breve información), ha sido cumplido más ampliamente en el segundo caso, puesto que, además de esa transmisión de información, se ha reclamado previamente la atención del destinatario de la misma.
Los efectos dinámicos son, por tanto, esenciales en la programación, y con vistas a facilitar su estudio pueden ser divididos en dos grandes grupos, según la magnitud de la zona de pantalla que se vea afectada por su acción:

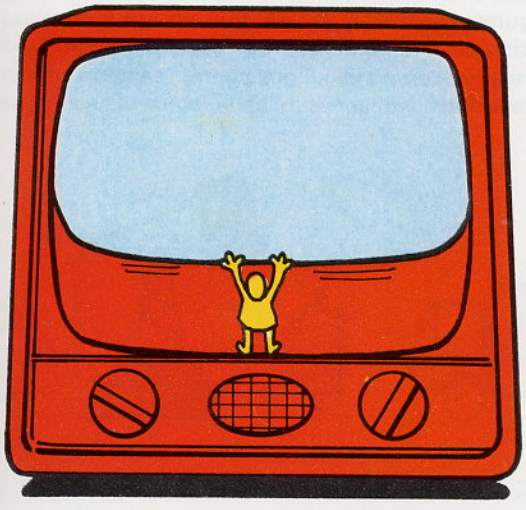
— Dinamismo de pantalla.
— Dinamismo de gráficos.

En el primer caso, la zona sobre la que se aplica el efecto de dinamismo es de dimensiones considerables, generalmente toda la pantalla, aunque las técnicas utilizadas a tal fin pueden ser empleadas también en áreas más restringidas



En la animación de gráficos, se efectúa el desplazamiento de un sprite sobre un fondo fijo.

El scroll es un ejemplo claro de dinamismo a pantalla completa, puesto que toda ella (fondo y primer término), es desplazada de manera solidaria.



¡!

Los bits dentro de un byte se numeran del 0 al 7, a partir de la derecha. El término «significativo» determina la importancia que un bit tiene en el valor de un byte. Así pues, los bits serán tanto más significativos cuanto más a la izquierda se encuentren, es decir, cuanto mayor sea la posición que ocupan.

*

Debido a la disposición de los bits dentro de los bytes del área de atributos, el valor decimal de uno de estos bytes se puede obtener por la siguiente fórmula: valor = código de tinta + código de fondo * 8 + brillo * 64 + intermitencia * 128.

¡!

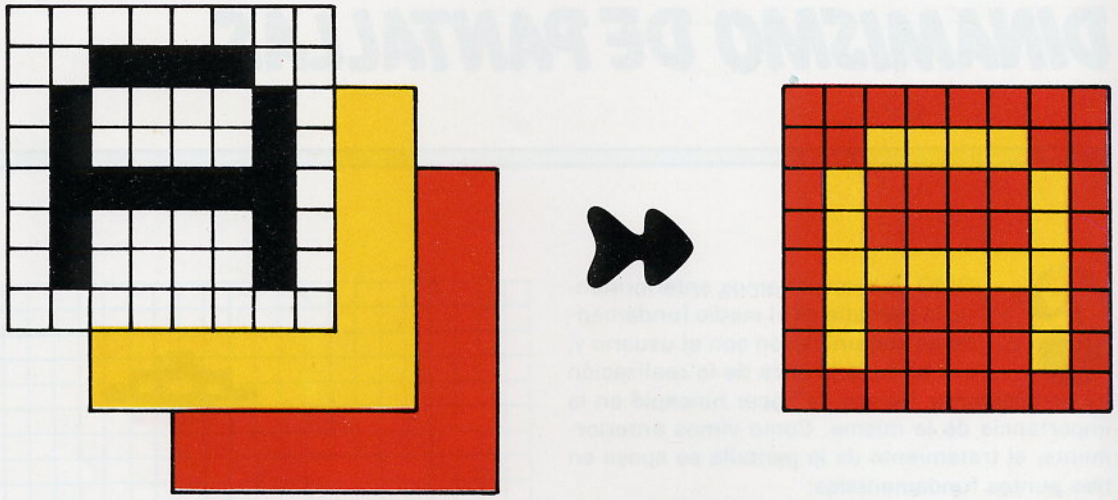
Cada bit del archivo de imagen corresponde a un punto de la pantalla del Spectrum. Puesto que cada uno de ellos es controlable independientemente, se dice que la pantalla de nuestro ordenador es DIRECCIONABLE PUNTO A PUNTO.

*

A la mínina unidad gráfica que en un ordenador se puede representar se le denomina *pixel*, abreviatura de la palabra inglesa *picture element*, cuyo significado es ELEMENTO GRAFICO.

*

La cantidad de *pixels* que se pueden representar simultáneamente en un ordenador, miden la RESOLUCION de la pantalla del mismo. En el caso del Spectrum, se pueden representar 49.152 *pixels*, por lo que se le considera de alta resolución.



En el Spectrum, la imagen que se envía al televisor o monitor se toma de dos fuentes: archivo de imagen (configuración de puntos) y archivo de atributos (color).

como, por ejemplo, un tercio o dos tercios de la pantalla. En todo caso, podemos decir que las zonas afectadas, constituyen generalmente rectángulos con el marco de la pantalla, y el efecto de dinamismo se distingue particularmente porque se aplica sobre el área completa que afecta, incluyendo tanto el fondo como el primer término, que puede estar constituido por una o varias unidades gráficas (texto, gráficos, etc...).

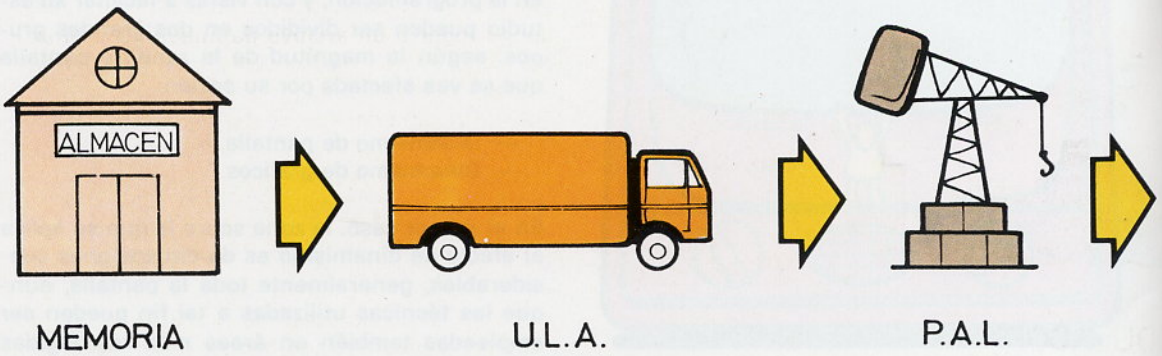
El segundo tipo de dinamismo, se distingue del anterior precisamente porque el objeto del mismo es la animación de determinadas unidades gráficas, generalmente reducidas en tamaño, y sobre las cuales se actúa independientemente del fondo; por ende, los sujetos pasivos de su acción, que en el argot microinformático se conocen como *sprites* (unidades gráficas), no suelen ser superficies regulares, debido a lo cual las técnicas a emplear son absolutamente distintas de las utilizadas para el dinamismo a pantalla completa.

Finalmente, podemos hablar del dinamismo en aplicaciones muy concretas, como es por ejemplo la presentación de mensajes, de extraordinaria importancia debido a su frecuente uso, y que disfruta del beneficio de ciertos trucos y técnicas especiales, obtenidos por extensión de los que se emplean para la animación de gráficos, y que se benefician de la mayor facilidad de tratamiento que el lenguaje BASIC aporta para los textos.

Un último punto a destacar en el tema del dinamismo, es la aplicación del sonido a este fin; sin embargo, éste merece un tratamiento independiente, por lo que será excluido de las técnicas de animación general que estudiaremos en las próximas páginas.

Aunque sin duda estaremos deseosos de adentrarnos en las técnicas de dinamismo, con las cuales podremos obtener espectaculares efectos, deberemos frenar nuestro ímpetu y adquirir previamente unas ciertas ideas básicas sobre la forma en que nuestro Spectrum trata la información que se envía al televisor o monitor. Estos conocimientos nos ayudarán a comprender mejor algunas técnicas y trucos, que debido a la utilización del código máquina y otras herramientas de relativa complejidad, nos podrían resultar un tanto arduas o difícilmente asimilables sin esta base.

El proceso de emisión de imagen en el Spectrum es llevado a cabo por varios elementos de su hardware.



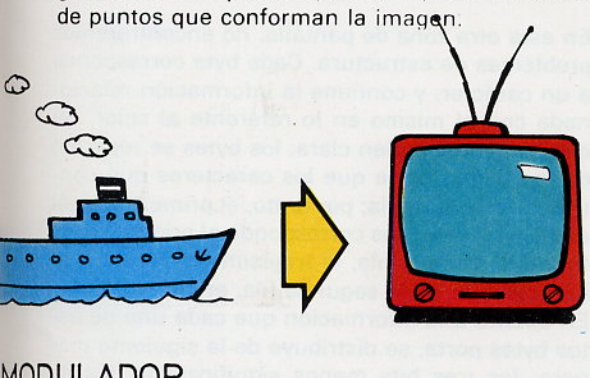
LA PANTALLA EN LA MEMORIA

La emisión de imagen en el Spectrum es un proceso de relativa complejidad, que involucra varios elementos del ordenador: la U.L.A. se encarga de recoger la información a transmitir de la memoria R.A.M.; desde aquí, los datos son enviados al codificador P.A.L. y posteriormente al modulador de U.H.F., que se ocupa de preparar la señal para que sea recibida por el televisor a través del canal 36 de U.H.F. (este último paso se ahorra al utilizar un monitor).

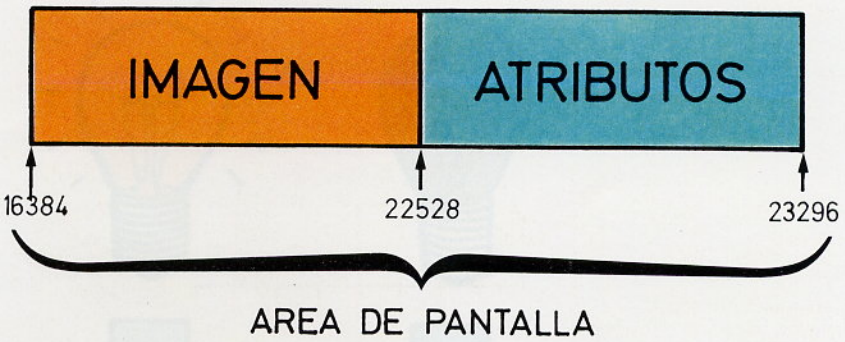
Como hemos visto, muy a grandes rasgos, el proceso de emisión de imagen es automático y concierne directamente a dispositivos físicos del aparato *(hardware)*, gracias a lo cual, nuestra tarea de programación se limitará única y exclusivamente a depositar en la zona de memoria adecuada, los datos que se desean transferir a la pantalla; este área de la memoria se denomina MEMORIA DE PANTALLA, y ocupa un total de 6.912 bytes, comprendidos desde la dirección 16.384 (primera de la memoria R.A.M.) hasta la 23295, ambas inclusive.

Para cualquier profano en la microinformática, la imagen transmitida por el ordenador es una determinada forma multicolor; sin embargo, para el Spectrum, y por ende para el programador, la imagen se divide en dos partes claramente diferenciadas: por un lado, los puntos que la integran, y por otro el color de éstos. Del mismo modo, la zona de memoria de pantalla se divide en dos áreas claramente distintas, destinadas cada una de ellas a contener uno de estos elementos de la imagen (figura y color).

La primera de ellas se denomina ARCHIVO DE IMAGEN, y ocupa 6.144 bytes (desde la dirección 16384 hasta la 22527, ambas inclusive). Esta zona se destina al almacenamiento de la configuración de la pantalla, es decir, la distribución de puntos que conforman la imagen.



MODULADOR



El área de memoria dedicada al almacenamiento de la pantalla, se divide en dos zonas: una para imagen y otra para atributos.

La otra zona se conoce como ARCHIVO DE ATRIBUTOS, y ocupa los últimos 768 bytes de la memoria de pantalla (desde 22528 hasta 23295), siendo su misión almacenar la información referente al color de la pantalla.

EL ARCHIVO DE IMAGEN

Cada uno de los puntos que configuran la imagen en la pantalla, puede estar «encendido» o «apagado» (color de primer término o color de fondo); dado que esta circunstancia supone únicamente dos posibles estados, cabe su representación informática mediante un bit, en el que se adopta el convenio 1-ENCENDIDO, 0-APAGADO.

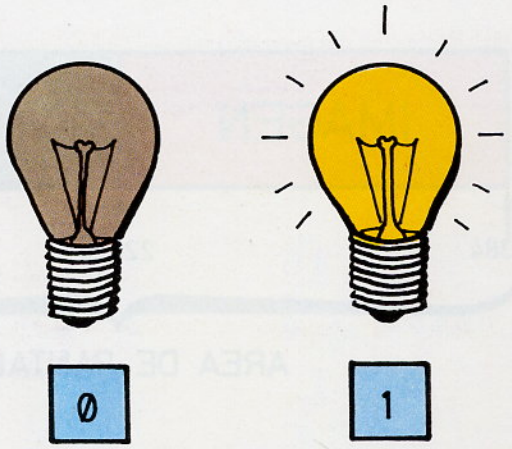
La imagen en el Spectrum está constituida por un total de 49.152 puntos, distribuidos en 192 líneas de 256 puntos cada una. Puesto que cada 8 bits suponen un byte de información, los puntos horizontales se agrupan de ocho en ocho, dando lugar a un archivo de imagen compuesto por 192 líneas de 32 bytes cada una (6.144 bytes).

Por tanto, para conectar determinado punto de la pantalla, sólo tendremos que depositar la información binaria conveniente en el byte del archivo de imagen que deseemos. Así, por ejemplo, efectuando **POKE 16384,255** conectaremos los ocho primeros puntos horizontales de la pantalla, puesto que la configuración binaria de 255 es 11111111; si sólo deseamos conectar un punto, no tendremos más que buscar la configuración

¡!

Todas las líneas que componen un programa escrito en BASIC, deben comenzar por un número comprendido entre 1 y 9999. Por esto, al ejecutar, por ejemplo, LIST 20000, la pantalla en blanco y el informe OK, corroboran la imposibilidad de encontrar tal línea. Sin embargo, al efectuar LIST 49162, obtenemos el mismo resultado que con LIST 10, es decir, el ciclo del argumento del comando LIST se repite a partir de 49152. Idéntica particularidad se cumple con LLIST. Probemos el siguiente programa:

```
10 INPUT "LISTADO A PARTIR DE ";X
20 CLS: PRINT "LIST ";X: PRINT
30 LIST X : GOTO 10
```



BIT

Cada bit del archivo de imagen representa en la pantalla un punto encendido o apagado.

¡!

Cuando realizamos la impresión mediante el tabulador **TAB**, durante el salto de una columna a otra, el ordenador memoriza el color en curso en ese momento. El siguiente programa lo demuestra:

```
10 FOR A=0 TO 20
20 PRINT PAPER 7; TAB 0;A; PAPER RND*6.5;TAB 15;A; PAPER 7; A;PAPER RND*6.5;TAB 29; PAPER 7;A
30 NEXT A
```

Si cambiamos los **TAB** anteriores por **AT**, veremos que cuando utilizamos este último no sucede lo mismo.

*

Cuando realizamos cálculos matemáticos con nuestro Spectrum, muchos resultados se presentan con gran número de decimales, a veces, innecesarios. Si queremos limitar el número de estos, podemos incluir como subrutina en nuestros programas, la línea 20 del listado siguiente:

```
10 INPUT "CUANTOS DECIMALES ";D,"NUMERO A REPRESENTAR";N
20 LET A= INT (N*10↑D)/10↑D
30 PRINT A: GOTO 10
```

binaria adecuada; por ejemplo: **POKE 16384,1** conecta el octavo punto de la pantalla (1 decimal = 00000001 binario), **POKE 16384,128** el primero (128 decimal = 10000000 binario), o **POKE 16384,197** los puntos 1, 3, 7 y 8.

Habremos observado claramente, que la modificación del contenido de la dirección 16.384 (primera del archivo de imagen), altera lo que sería la primera línea del primer carácter de la primera fila; de forma similar, 16.385 controla la primera línea del segundo carácter de la primera fila. Hasta este punto todo es muy fácil de retener, ya que cada byte se corresponde con una columna de caracteres; sin embargo, según este razonamiento, el byte 33 del archivo de imagen, debería corresponder a la segunda línea del primer carácter de la primera fila, y esto lamentablemente no se cumple: este byte es el que controla la primera línea del primer carácter de la segunda fila.

Así pues, las complicaciones aumentan, ya que en la memoria no se ordenan los bytes tal y como luego aparecen en la pantalla del televisor, sino que primeramente se encuentran las primeras líneas de todas las filas de caracteres, a continuación las segundas, las terceras, etc. hasta llegar a las octavas (últimas).

Lamentablemente, las desgracias nunca vienen solas, y esta distribución tampoco es del todo cierta. A efectos de estructura de la memoria, la pantalla se divide en tres tercios de ocho filas cada uno (64 líneas). Cada uno de ellos ocupa, por tanto, 2 K de memoria (2 × 1.024 = 2.048 bytes), lo que suma el total de 6 K del archivo de imagen (6 × 1.024 = 6.144 bytes).

Así pues, la exacta distribución de este archivo es: primeras líneas de las ocho primeras filas, segundas líneas de las ocho primeras filas, etc. hasta las octavas líneas de las ocho primeras filas; a continuación, se representan las ocho primeras líneas de las filas 9 a 15, y unas vez finalizado este tercio, es decir, representadas las ocho octavas líneas de la fila 9 a la 15, se pasa a la representación del último tercio, siguiendo el mismo sistema. Bueno, ¡por lo menos los tres tercios si que están por orden!

Esta organización, aparentemente un tanto caótica, se debe a razones de *hardware*, que en nuestro caso es como decir «razones de peso». En todo caso, la distribución de la pantalla es algo que deberemos tener muy presente al experimentar con las técnicas de dinamismo a pantalla completa. Por otra parte, esta es la explicación de por qué las pantallas que el Spectrum almacena, se cargan de una manera un tanto peculiar.

Para dar un repaso a como se estructura el archivo de imagen en nuestro ordenador, ejecutemos el siguiente programa, en el que la pantalla se irá llenando de líneas según el orden de los byes que las controlan:

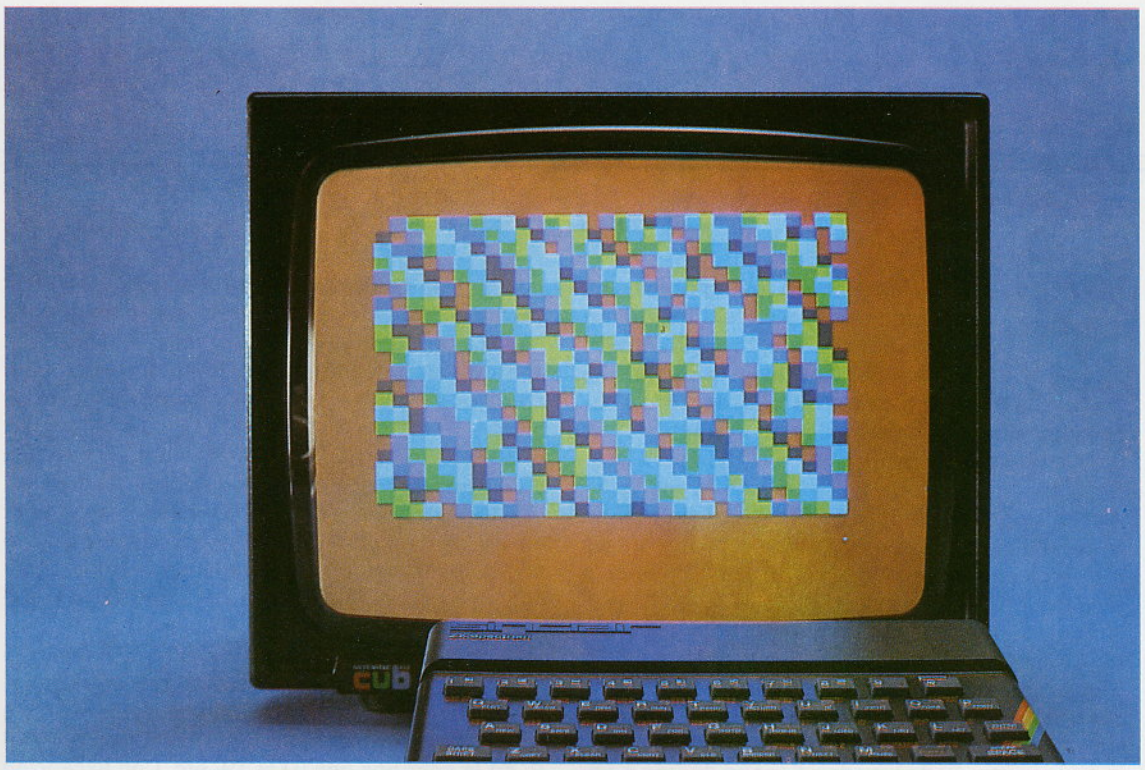
```
10 PAPER 7:INK 0: BORDER 6: CLS
20 FOR I=0 TO 21
30 PRINT I
40 NEXT I
50 FOR I=16384 TO 22527
60 POKE I ,255
70 NEXT I
80 GO TO 80
```

Como habremos podido observar, la diferenciación que desde el punto de vista de la programación se hace entre la zona de edición (inicialmente, las dos últimas líneas) y la de pantalla, no tiene ningún efecto en el archivo de imagen, que considera la pantalla como un todo único.

EL ARCHIVO DE ATRIBUTOS

En esta otra zona de pantalla, no encontraremos problemas de estructura. Cada byte corresponde a un carácter, y contiene la información relacionada con el mismo en lo referente al color. Su distibución está bien clara: los bytes se reparten de la misma forma que los caracteres que controlan en la pantalla; por tanto, el primer byte del archivo de atributos corresponde al primer carácter de la primera fila, el trigésimo tercero al primer carácter de la segunda fila, etc.

En cuanto a la información que cada uno de estos bytes porta, se distribuye de la siguiente manera: los tres bits menos significativos (los de



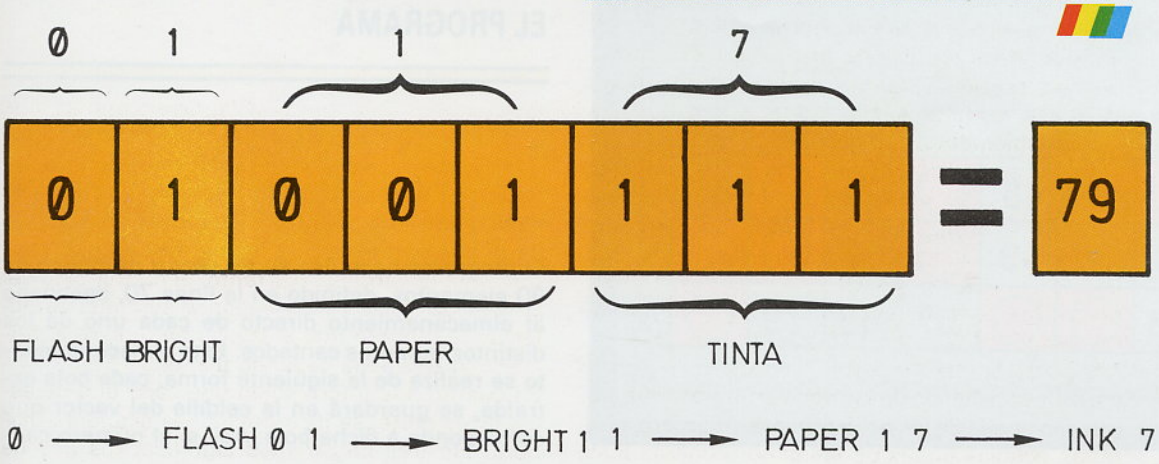
La organización del archivo de imagen divide la pantalla en tres tercios consecutivos, cada uno de los cuales ocupa 2 K. Cada uno de estos tercios, se encuentra a su vez compuesto por 64 líneas de 32 bytes.

más a la derecha), codifican el color de la tinta, los tres siguientes (bits 3, 4 y 5) el color del papel, el sexto bit controla el brillo, y el más significativo el **FLASH**.

En el siguiente programa escribiremos un asterisco en la pantalla, y veremos como mediante **POKE**s al área de atributos, podemos alterar sus características de color, sin cambiar su configuración de puntos. Si queremos aumentar la velocidad a la que discurre el programa, pulsemos cualquier tecla:

```
  10 CLS : PRINT "*": LET F=0
  20 PRINT '''''FLASH.:'''''BRIGHT:'''
"PAPER.:'''''INK...:"
  30 FOR I=0 TO 255
  40 POKE 22528,I
  50 LET C=I
  60 IF C>127 THEN LET F=1: LET C=C-128
  70 LET B=0: IF C>63 THEN LET B=1: LET
C=C-64
  80 LET P=INT (C/8)
  90 LET T=C-8*P
 100 PRINT AT 4,7;F;AT 5,7;B;AT 6,7;P;AT
7,7;T
 110 PAUSE 25
 120 NEXT I
```

Cada byte del área de atributos porta la información sobre tinta, fondo, brillo e intermitencia de un carácter de la pantalla.



¡!

Los aficionados a las bromas, encontrarán útiles, sin duda, las siguientes líneas:

```
10 CLS: PAUSE 15:
POKE 23624,0: CLS:
PAPER 0: CLS
15 PAUSE 55: PA-
PER 7: POKE
23624,56: CLS
20 PRINT #1;"©
1982 Sinclair Re-
search Ltd": PAUSE
0
```

*

En la foto de la izquierda, aparece una pantalla muy alegre que demuestra el efecto de **POKE** sobre el área de atributos. He aquí el programa que la genera: **10 POKE 22528+RND*768, RND*255: GOTO 10**

¡BINGO!

EL juego del bingo o lotería casera, ha alcanzado con el paso del tiempo un gran número de adeptos. Así como el palé, el parchís o los juegos de cartas, el bingo es uno de los pasatiempos más socorridos para amenizar tardes aburridas.

El programa que a continuación estudiamos, gestiona la extracción de bolas de una forma totalmente aleatoria, como si se tratase de un perfecto bombo de bingo, memorizando cada uno de los números para la posterior verificación de línea o bingo respectivamente.

Aunque la mayoría de nosotros conoce la mecánica general de este juego, daremos un pequeño repaso a las reglas fundamentales: el elemento central es un bombo, dentro del cual se encuentran noventa bolas numeradas desde la uno hasta la noventa. Cada jugador dispone de uno o varios cartones con números, de forma que según se van extrayendo las bolas del bombo, se puede ir señalando en los mismos qué números de los presentes en nuestras tarjetas han aparecido. El jugador que primero tenga la suerte de tachar los cinco números de una misma línea de su cartón, será agraciado con el premio de LINEA (si esto ocurriera en una misma jugada con otros cartones, se repartiría el premio entre los afortunados). A partir de este momento, el premio de línea ha sido repartido, y ningún beneficio se otorgará a los próximos jugadores que obtengan línea. El siguiente premio, y el más codiciado, es el de BINGO, destinado a aquel o aquellos jugadores que antes tachen todos los números de su cartón.

El Spectrum no genera los cartones, debido a lo cual tendremos que conseguirlos de algún antiguo juego de bingo perdido por casa, o en el peor de los casos, fabricarlos nosotros mismos. Para ello debemos seguir determinadas normas:

— Los cartones se agrupan en series de seis, que contienen el total de bolas del bombo.

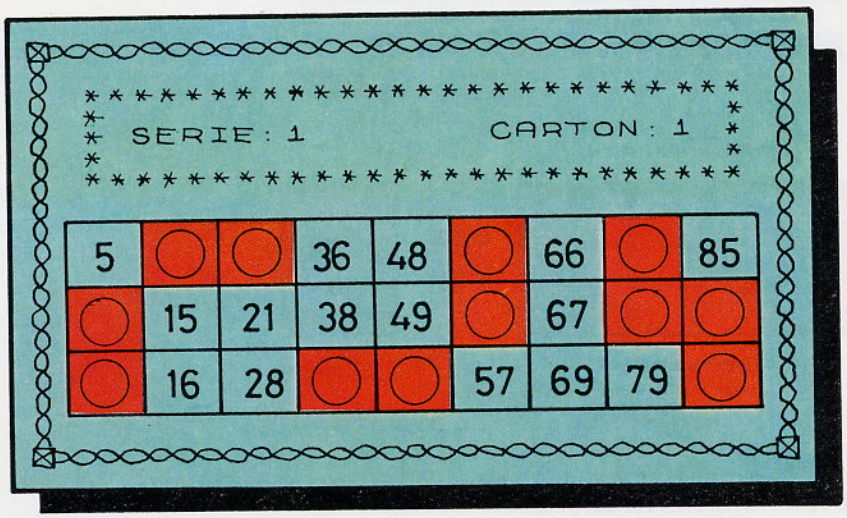
— Cada cartón presenta tres filas divididas en nueve columnas, en cada una de las cuales se colocan los números del cartón correspondientes a una decena: en la primera, los números del 1 al 9; en la segunda del 10 al 19, etc. Con excepción del 90, que se añadirá en la última columna, junto con las decenas de ocho.

— En cada casilla sólo habrá un número, lo cual implica que no podrá existir más de un número perteneciente a una determinada decena en una misma línea.

— Cada columna del cartón contendrá al menos un número, y nunca tres.

En cuanto a los premios, sugerimos que la línea se pague a una tercera parte del total de cartones recaudados, y el bingo con las dos terceras partes restantes; pudiéndose utilizar como moneda, el popular «dólar lentejiense», el «petro-garbanzo» o cualquier otra moneda admitida internacionalmente, para este tipo de juegos de azar.

Si carecemos de cartones de bingo o de alguna lotería casera, podemos fabricarlos nosotros mismos.



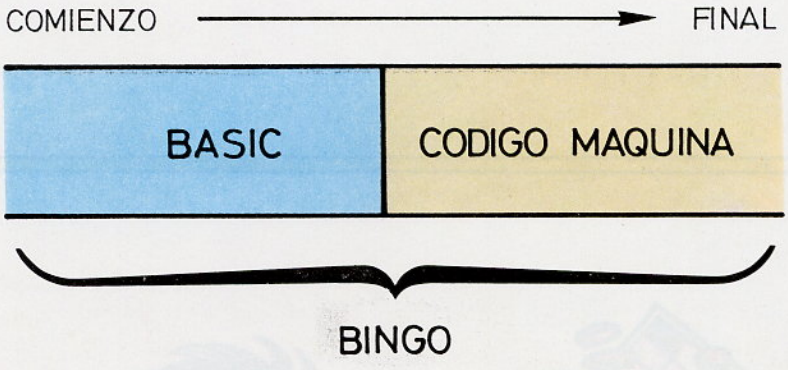
EL PROGRAMA

El programa hace uso de un vector numérico de 90 elementos, definido en la línea 70, destinado al almacenamiento directo de cada uno de los distintos números cantados. Este almacenamiento se realiza de la siguiente forma: cada bola extraída, se guardará en la celdilla del vector que corresponda a dicha bola. Así, si el número can-

tado es el 43, éste se almacenará en la casilla 43 del vector.

Para la verificación de línea o bingo, nuestro Spectrum utiliza el mismo sistema. Recoge mediante un **INPUT** el número a verificar, recupera el contenido de la celdilla asignada a ese número, y compara dicho contenido con el valor introducido. Si los dos valores son iguales, el ordenador pide el siguiente número a verificar; en caso contrario, emite un mensaje indicando que la línea o el bingo, dependiendo de la verificación que estemos efectuando, no es correcto.

La ejecución del programa puede ser interrumpida en cualquier momento, pulsando la tecla «B»,



El programa BINGO está compuesto por un soporte BASIC y una subrutina en código máquina, que deberán quedar almacenados en la cinta, según el orden que se indica en el gráfico.

lo que dará acceso a un menú de tres opciones: comenzar un nuevo bingo, verificar la línea cantada y verificar el bingo cantado.

En este programa, hemos utilizado nuevamente la subrutina en código máquina de caracteres gigantes (PSION Computers), por lo cual, aquellos que deseemos disfrutar del presente programa de bingo, deberemos introducir la citada subrutina aparecida anteriormente (programa LA BOMBA); bien a través del listado, siguiendo las instrucciones que a tal fin se brindaron en el comentario del programa mencionado, o bien grabándolo a continuación del soporte BASIC, si es que ya fue introducido en aquel momento.

La auto-ejecución del programa es obligatoria, por tanto, para la grabación del programa, tendremos que emplear el comando **SAVE "BINGO" LINE 1030.**

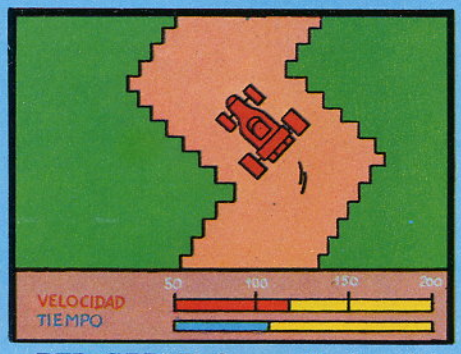
```
 10 REM ************************          J.M.MAYORAL SERRANO          *******
****************
 20 POKE 23609,100: LET conta=1: INK 9: CLS : LET contador=0
 30 GO TO 260
 40 CLS : BORDER 0: PAPER 0: CLS : LET p$=" COMENZAMOS ": INK 1: PAPER 6: LET y
y=90: LET xs=2: LET ys=4: GO SUB 690: PAPER 0: LET ca=1
 50 FOR N=1 TO 10: BEEP .2,20: BEEP .2,5: NEXT N
 60 POKE 23609,100: LET conta=1: INK 9: CLS : LET contador=0
 70 CLS : DIM B(90)
 80 BORDER 0: PAPER 0: INK 9: CLS
 85 RANDOMIZE
 90 LET n=(INT (RND*90))+1
100 IF B(n)=n THEN  GO TO 90
110 LET B(n)=n
120 LET CONTADOR=CONTADOR+1
130 PAPER 0: CLS : BEEP .1,30: BEEP .15,40
140 PLOT 81,50: DRAW 95,0: DRAW 0,77: DRAW -95,0: DRAW 0,-77
150 PLOT 81,127: DRAW 20,10: DRAW 95,0: DRAW 0,-77: DRAW -20,-10
160 PLOT 81+95,127: DRAW 20,10
170 LET p$=STR$ B(n): LET ca=ca+1: GO SUB 680
180 LET p$=STR$ contador: LET yy=5: LET xs=3: LET ys=3: LET xx=5: INK 6: PAPER
0: GO SUB 0700
190 LET p$=STR$ contador: LET yy=150: LET xs=3: LET ys=3: LET xx=200: INK 6: PA
PER 0: GO SUB 700
200 PAUSE 60
210 IF INKEY$="b" THEN  INK 9: GO TO 260
220 IF ca=91 THEN  GO TO 240
230 GO TO 90
240 PAPER 0: CLS : LET xs=1: LET ys=3: LET yy=100: LET p$=" SE ACABARON LAS BOL
AS ": PAPER 2: GO SUB 690: LET yy=150: LET ys=2: LET p$="Pulse una tecla para em
pezar": PAPER 0: GO SUB 690: PAUSE 0: RUN 40
250 REM ************************          FIN EJECUCION BINGO          *******
****************
260 PAPER 1: BORDER 1: CLS : PRINT TAB 12;"OPCIONES"
270 PRINT  PAPER 7; FLASH 1;AT 2,2;"1": PRINT AT 2,5;"EJECUCION DE BINGO"
280 PRINT  PAPER 7; FLASH 1;AT 4,2;"2": PRINT AT 4,5;"VERIFICACION DE LINEA"
290 PRINT  PAPER 7; FLASH 1;AT 6,2;"3": PRINT AT 6,5;"VERIFICACION DE BINGO"
300 PRINT  PAPER 2;AT 10,0;"  Al cantar linea o bingo"; INVERSE 1;" pulsar  la
letra B
310 PLOT 0,50: DRAW 255,0:            PLOT 0,48: DRAW 255,0
320 FOR f=17 TO 21: FOR c=0 TO 31: PRINT  INK 6;AT f,c;"B": NEXT c: NEXT f
330 PRINT  INK 9;AT 19,3;"INTRODUZCA NUMERO DE OPCION"
340 LET K$=INKEY$
350 IF K$="" THEN  GO TO 340
360 IF K$="1" THEN : BEEP .2,40: LET conta=1: GO TO 40
370 IF K$="2" AND CONTADOR>=5 THEN  BEEP .2,40: LET conta=1: GO TO 460
380 IF K$="3" AND contador>14 THEN  BEEP .2,40: LET conta=1: GO TO 560
385 IF contador<15 THEN  GO TO 340
390 IF conta>=3 THEN  GO TO 1050
400 CLS : IF conta>=2 THEN  LET conta=conta+1: LET p$="Mira que eres BRUTO": LE
T yy=60: LET xs=1: LET ys=2: GO SUB 690: GO TO 420
410 GO TO 430
420 LET xs=1: LET ys=2: LET yy=80: LET p$="Si solo tienes 3 opciones,": GO SUB
690: LET yy=100: LET p$="QUE PORRAS HACES PULSANDO UN ": GO SUB 690: LET yy=120:
 LET p$="NUMERO MAYOR DE 3 ?!!!": GO SUB 690: PAUSE 200: GO TO 260
430 INK 9: CLS : LET p$="POR FAVOR!": LET xs=2: LET ys=3: LET yy=50: GO SUB 690
: LET p$="Sabes leer?": LET yy=80: LET xs=2: LET ys=2: GO SUB 690: BEEP 3,-10
440 LET conta=conta+1
450 CLS : LET p$="Solo hay 3 opciones": LET yy=80: LET xs=1: LET ys=2: GO SUB 6
90: LET yy=110: LET p$="FIJATE BIEN": GO SUB 690: LET yy=140: LET p$="No creo qu
e sea tan dificil": GO SUB 690: BEEP 3,-20: GO TO 260
460 LET cont=0
470 CLS : PRINT AT 10,2;"INTRODUCIR UN SOLO NUMERO EN    CADA SECUENCIA"
480 INPUT "Numero=";l
485 IF L<=0 OR L>90 OR L-INT L THEN  BEEP 1,-10: GO TO 480
490 IF B(l)=l THEN  LET cont=cont+1: GO TO 540
500 CLS : LET p$="EL": LET yy=10: LET xs=2: LET ys=4: GO SUB 690: LET p$=STR$ l
: LET yy=50: LET xs=5: LET ys=5: INK 5: BRIGHT 1: GO SUB 690: FLASH 0
510 PAPER 1: INK 9: LET p$="NO LO HE DICHO": LET yy=100: LET xs=2: LET ys=3: GO
 SUB 690: PAPER 0
520 BRIGHT 0: BEEP .5,20: BEEP .5,30: PAUSE 100: PAPER 6: BORDER 6: INK 2: CLS
: LET p$="La linea no es correcta": BRIGHT 1: LET yy=70: LET xs=1: LET ys=3: GO
SUB 690: BRIGHT 0: PAPER 0:
530 BEEP .5,30: PAUSE 150: CLS : GO TO 80
540 INK 9: IF CONT=5 THEN  CLS : LET p$="LINEA CORRECTA": LET yy=80: LET xs=2:
LET ys=3: GO SUB 690: BEEP 2,50: PAUSE 100: CLS : GO TO 80
550 GO TO 470
560 LET cont=0
570 CLS : PRINT AT 10,2;"INTRODUCIR UN SOLO NUMERO EN    CADA SECUENCIA"
580 INPUT "Numero=";l
590 IF B(l)=l THEN  LET cont=cont+1: GO TO 650
600 CLS : LET p$="EL": LET yy=10: LET xs=2: LET ys=3: GO SUB 690: PAPER 7: INK
2
610 LET p$=STR$ l: LET yy=50: LET xs=5: LET ys=5: PAPER 1: INK 5: GO SUB 690: P
APER 1: INK 9
620 LET p$="NO LO HE DICHO": LET yy=110: LET xs=2: LET ys=3: GO SUB 690: BEEP .
2,40: BEEP .2,10: PAUSE 100
630 CLS : LET p$="El bingo no es correcto": LET yy=90: LET xs=1: LET ys=3: INK
6: BRIGHT 1: GO SUB 690: BRIGHT 0
640 BEEP .2,40: BEEP .2,10:: PAUSE 150: PAPER 0: INK 9: CLS : GO TO 80
650 INK 9: IF cont=15 THEN  CLS : LET P$=" BINGO CORRECTO": LET YY=80: LET XS=2
: LET YS=3: GO SUB 690: BEEP 2,50: PAUSE 100: CLS : GO TO 670
660 GO TO 570
670 RUN
680 INK 7: BRIGHT 1: PAPER 2: LET yy=70: LET xs=5: LET ys=5
690 LET xx=(256-8*xs*LEN p$)/2
700 LET i=23306
710 POKE i,xx: POKE i+1,yy: POKE i+2,xs: POKE i+3,ys: POKE i+4,8
720 LET i=i+4: LET u=LEN p$
730 FOR q=1 TO u
740 POKE i+q,CODE p$(q)
750 NEXT q
760 POKE i+u+1,255
770 LET u=USR 32256
780 RETURN
790 BORDER 7: PAPER 7:
800 CLS
810 INK 3
820 LET p$="BINGO": BRIGHT 1: LET yy=0: LET xs=6: LET ys=10
830 GO SUB 690
840 LET p$="ZX.Spectrum": BRIGHT 0: LET yy=80: LET xs=2: LET ys=4: INK 0
850 GO SUB 690
860 RESTORE 930
870 FOR f=14 TO 17
880 READ k
890 FOR c=0 TO 31
900 PRINT  PAPER k; BRIGHT 1;AT f,c;" "
910 NEXT c: NEXT f: BRIGHT 0
920 GO TO 940
930 DATA 2,6,4,5
940 RESTORE 950
950 DATA .1,7,.3,12,.1,7,.3,12,.1,16,.6,19,.12,16,.12,19,.12,19,.12,19,.25,17,.
12,14,.6,12
960 DATA  .4,-60,.4,19,.4,18,.4,17,.4,16,.2,14,.15,12,.1,11,.1,-60,.15,9,.3,7
970 FOR b=1 TO 24
980 READ t,n: BEEP t,n
990 NEXT b
1000 PRINT AT 21,2;"(C)"
1010 LET p$="J.M. MAYORAL SERRANO": LET yy=160: LET xs=1: LET ys=2: INK 0: GO SU
B 690
1020 PAUSE 150: RUN 40
1030 CLEAR 32255: LOAD "LIT"CODE
1040 GO TO 1070
1050 CLS : LET yy=60: LET xs=1: LET ys=2: LET p$="Vuelvo a REPETIRTELO!!": GO SU
B 690
1060 PAUSE 100: CLS : GO TO 400
1070 BRIGHT 0: BORDER 7: PAPER 7: INK 1: CLS
1080 LET yy=80: LET ys=3: LET xs=2: LET p$="Para la cinta": GO SUB 690
1090 PAUSE 200
1100 CLS : BORDER 0: PAPER 0: INK 6: BRIGHT 1: CLS
1110 LET P$="INTRODUCE UN NUMERO   (0-9999)": LET YY=160: LET XS=1: LET YS=2: IN
K 4: GO SUB 690
1120 INPUT  LINE M$
1130 IF LEN M$>4 THEN  GO TO 1120
1140 RANDOMIZE INT (RND*VAL M$)
1150 INK 6: CLS
1160 LET p$="IMPORTANTE": LET yy=10: LET xs=3: LET ys=5: GO SUB 690: BRIGHT 0:
1170 LET yy=70: LET ys=2: LET xs=1: LET p$="En el momento de cantar": INK 7: GO
SUB 690
1180 LET yy=90: LET p$="LINEA o BINGO": GO SUB 690
1190 LET yy=110: LET p$="MANTENER pulsada la letra ''B''": GO SUB 690
1200 LET yy=130: LET p$="hasta que aparezca un menu": GO SUB 690
1210 LET yy=170: LET xs=2: LET ys=2: LET p$="VALE??": GO SUB 690
1220 PAUSE 300: GO TO 790
```

Primera Revista Española en Cassette

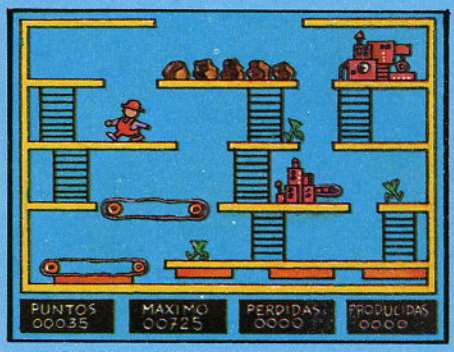
SPECTRUMANIA 2

para Spectrum 16K ó 48K

II época

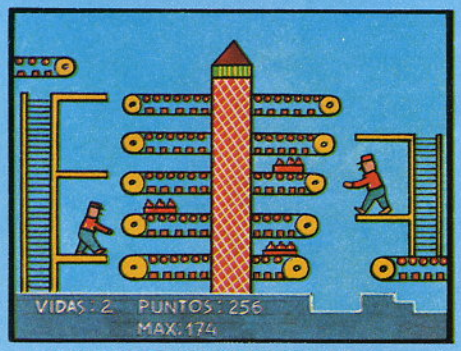


RED CAR: En la parrilla de salida rugen los motores, ¿serás capaz de superar tus propias marcas?

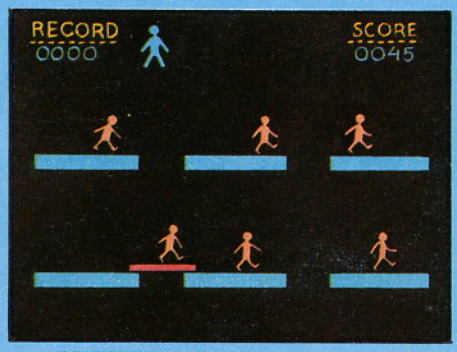


ROBOT FACTORY: Unos defectuosos robots-monstruos te persiguen mientras intentas cumplir tu misión.

ASTRONOMIA: Quasares, agujeros negros, nebulosas, planetas... El Universo no tiene secretos para ti.



LOS HERMANOS MARIO: La rapidez de reflejos es esencial para ayudar a los Mario Bros.



PUENTES : Si no consigues controlar los puentes colgantes... muchos sufrirán las consecuencias.

ADEMAS:

* WRECKAGE
* PUZZLE
* TENIS
* CUATRO EN RAYA
* SUPER-STORE
* CODIGO MAQUINA: QUICK SORT
* CONCURSOS
* COMENTARIOS PROGRAMAS Y PANTALLAS DE:
  - MONTY IS INNOCENT
  - STARSTRIKE
  - TIR NA NOG
  - MATCH DAY Y ... MUCHO MAS

¡YA ESTA A LA VENTA! COMPRALA EN TU QUIOSCO HABITUAL

BOLETIN DE PEDIDO
Enviar a: VENTAMATIC - Avda. de Rhode, 253 - ROSES (Girona) - Tel. (972) 25 79 20.

Deseo:

- ☐ Recibir el N.º 2 (2.ª época) / N.º 1 (2.ª época) de SPECTRUMANIA, al precio de 695,– ptas. cada uno.
- ☐ Recibir el N.º 1 (1.ª época) / N.º 2 (1.ª época) de SPECTRUMANIA, al precio de 500,– ptas. cada uno.
- ☐ Suscribirme por 6 números a la revista SPECTRUMANIA, a partir del N.º _____, al precio de 4.000,– ptas. (SOCIOS CLUB NACIONAL DE USUARIOS DE LOS ZX: 3.600,– ptas.).
- ☐ Recibir el CATALOGO COMPLETO VENTAMATIC (32 páginas) de artículos de micro-informática, al precio de 200,– ptas., a deducir de mi próximo pedido a VENTAMATIC.
- ☐ Ser inscrito como socio del Club Nacional de Usuarios de los ZX y recibir el Carnet de Socio y 6 boletines a partir del último publicado, al precio de 2.500,– ptas.

ATENCION: Las personas que se suscriban por 6 números de SPECTRUMANIA antes del 30 de julio de 1985, recibirán un PROGRAMA-SORPRESA DE REGALO.

Fecha: _____

Nombre: _____

Apellidos: _____

Dirección: _____

Población: _____

Provincia: _____ D.P.: _____

Señalar con una cruz la forma de pago:

- ☐ Talón adjunto (sin gastos de envío).
- ☐ Giro Postal N.º _____ (sin gastos de envío).
- ☐ Contra-Reembolso (+ 200,– ptas. de gastos de envío).
- ☐ Tarjeta VISA / MASTERCARD / AMERICAN EXPRESS (+ 200,– ptas. de gastos de envío), N.º _____
  Caduca: _____ FIRMA:

GARANTIA DE CARGA

8 VIDEO-JUEGOS Y 1 UTILIDAD EN CODIGO MAQUINA, 1 PROGRAMA DE GESTION, 1 PROGRAMA EDUCATIVO, COMENTARIOS DE SOFTWARE, CONCURSOS.

OFERTA ESPECIAL
Vale por 10 % de DESCUENTO y un REGALO-SORPRESA en tu próxima compra de HARDWARE y SOFTWARE directamente a VENTAMATIC
(este descuento no es aplicable a ejemplares de SPECTRUMANIA)

VENTAMATIC

CONTIENE REGALOS SORPRESA GRATIS
2 Wafadrives y 200 Programas
¡Busca en tu pantalla!