Nº 18 JUNHO DE 1992 PREÇO: 550$00

Spooler



MAGAZINE DOS UTILIZADORES PC



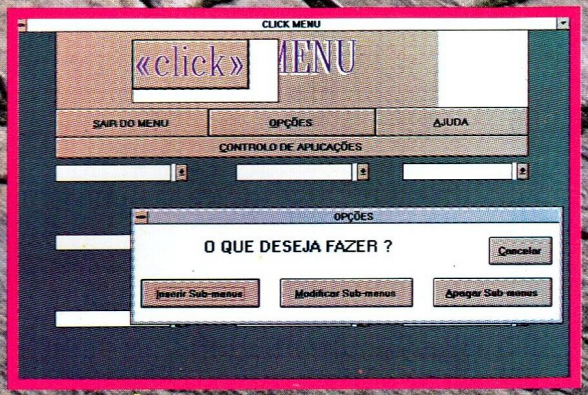
MÓDULOS BINÁRIOS

- HELP
- EXCLUI
- COMPRIME

A Dinastia INTEL (II)

PROGRAMAR EM:

- ASSEMBLY
- C
- CLIPPER



MENÚ PARA WINDOWS

A disqueta gratuita que acompanha cada exemplar da revista, é parte integrante da mesma e não pode ser comercializada separadamente.

NOVA SÉRIE INVES PC



PROMOÇÃO DE LANÇAMENTO

320 000$00 + IVA



INVES PC-325 S — 220 000$00 + IVA

- Proc. 80386SX-25 MHz
- FDD 1,44 Mb
- Placa SVGA
- 2 Mb RAM
- HD 52 Mb
- Monitor SVGA 14" – côr (1024x768)



INVES NB-320S (Notebook)

- Processador 80386SX-20 MHz
- 2 a 4 Mb RAM
- 2 Portas série, 1 Paralela, 1 Game
- Porta 8 pinos para Handy Scanner
- Drive 3 ½" – 1,44 Mb
- Disco rígido 2½" – 80 Mb
- Ecrã cristais líquidos contra-iluminados VGA 640x480, 32 tonalidades de cinzento
- Socket para 80387SX
- Teclado 81 teclas, 12 funções
- Dimensões: 290x220x53 mm
- Peso: 2,6 Kg

INVES PC-340D — 320 000$00 + IVA

- Proc. 80386DX-40 MHz
- FDD 1,44 Mb
- Placa SVGA 1 Mb
- 2 Mb RAM
- HD 105 Mb
- Monitor SVGA 14" – côr (1024x768)

INVES PC-433D — 420 000$00 + IVA

- Proc. 80486DX-33 MHz
- FDD 1,44 Mb
- Placa SVGA 1 Mb
- 4 Mb RAM
- HD 105 Mb
- Monitor SVGA 14" – côr (1024x768)

INVES PC-450D — 499 000$00 + IVA

- Proc. 80486DX-50 MHz
- FDD 1,44 Mb
- Placa SVGA 1 Mb
- 8 Mb RAM
- HD 105 Mb
- Monitor SVGA 14" – côr (1024x768)

OUTRAS CONFIGURAÇÕES POSSÍVEIS (todos os modelos):

- Drives de 3½" - 1,44 Mb e 5¼" - 1,2 Mb
- Discos de diversas capacidades
- Placas vídeo VGA de 256 Kb a 1 Mb RAM
- Monitores VGA mono ou SuperVGA (1024x768)

GARANTIA 2 ANOS

CONDIÇÕES ESPECIAIS DE PAGAMENTO

SAT
SERVIÇO ASSISTÊNCIA TÉCNICA
TODAS AS MARCAS
Tel. 778 73 24
Fax 778 53 01

inves ®

REPRESENTANTE OFICIAL:

INVESMÁTICA – Investigação e Informática, Lda.
Rua Abel Botelho, 8-A – 1500 LISBOA – PORTUGAL
Telefs.: 351-1-7740880/7787324/7787326 – Fax: 351-1-7785301

Graficria pub

Nº 18 – Junho de 1992

Spooler



QUEM SÃO OS LEITORES DA SPOOLER?

Uma ocasional consulta ao nosso ficheiro de assinantes revelou-nos alguns dados curiosos os quais, pelo seu significado, gostaríamos de partilhar convosco. Se bem que o número de leitores da **Spooler** seja muito superior ao número de assinantes, podemos considerar o nosso ficheiro como suficientemente representativo de diferentes «classes». Entre as mais significativas escolhemos as classes etária e profissional. E, para começar, sabia que 87,7 por cento dos leitores se situam na classe etária dos 15 aos 45 anos e que, desses, 52,2 por cento têm entre 21 e 35 anos? Curiosamente, os extremos estão muito próximos: 1,4 por cento entre os 10 e os 14 anos e 1,8 por cento entre os 56 e os 60 anos. No que respeita às ocupações ou classes profissionais, como seria de esperar, a maior percentagem vai para os estudantes: 29,8 por cento, seguindo-se-lhes os técnicos de informática com 21,6 e, logo depois, os engenheiros com 11,2 por cento. O número de professores também é significativo: 9,7 por cento e, não muito longe, os administrativos com 7,8 e os comerciantes e industriais com 8,2. Como curiosidade, refira-se os médicos com 3,4, os militares com 2,7, os economistas e contabilistas com 5 e, finalmente, os juízes e advogados com 0,5 por cento. Deixamos as conclusões desta pequena estatística à consideração do leitor.

ESCRITORES, ENTÃO?

O apelo feito no último Editorial da **Spooler** não foi muito frutuoso. Não o vamos repetir, mas lembramos a nossa incumbência para apreciar e propor à Gerência da GRAFICRIA a publicação de livros na área da informática. Não acreditamos que, entre tantos profissionais e amadores, não existam competências a quem só faltaria um pequeno «empurrão» para concretizar, em letra de forma, a latente sabedoria. Talvez a época não fosse a mais propícia. Mas avizinham-se as férias e, com elas, uma maior disponibilidade para se «deitar mãos à obra». Continuamos à vossa espera.

Renato Casquilho

Spooler

- **Propriedade:** Graficria, Publicidade e Artes Gráficas, Lda.
- **Director de Edição:** Franco Gomes ■ **Director**: Renato Casquilho
- **Redacção:** Franco Gomes, Paulo Bernardoe Renato Casquilho.
- **Colaboradores:** Carlos Ladeira, Cunha Brandão, José Orlando Pereira, José Páscoa, Paulo Bernardo, Vítor Monteiro.
- **Ilustrações:** Leonor Gomes
- **Relações Públicas e Publicidade:** Fátima Ribeiro
- **Assinaturas e encomendas:** Paula Martinez
- **Redacção, Publicidade e Serviços Administrativos**: Rua Duarte Pacheco Pereira, 69-A – Damaia – 2700 AMADORA – Telfs.: 475 22 89/474 60 11/41
- **Produção Gráfica:** Graficria, Publicidade e Artes Gráficas, Lda.
- **Impressão:** Empresa Litográfica do Sul, S.A. ■ **Distribuição:** Electroliber
- **Revista bimestral** ■ **Preço:** 550$00 ■ **SRIP N.º** 213900
- **Tiragem:** 12500 exemplares ■ **Dep. Legal N.º:** 28587/89

2 Informática em Notícia

4 O Conteúdo da Disqueta

6 Correio dos Leitores

10 Programar em Assembly (II)

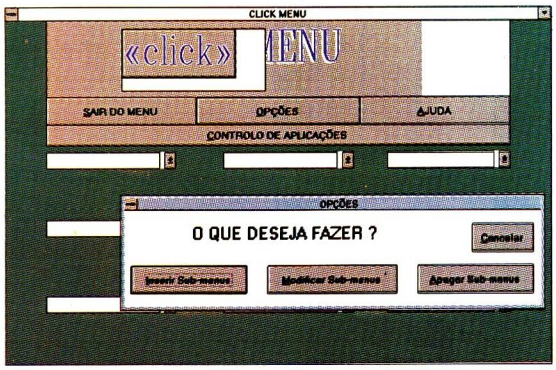
15 Help. Transforme o seu PC num Livro

18 Exclui

20 A Construção de Módulos Binários

22 Comprime

23 Testar o «Rato» em Pascal



24 Menú

26 Programar em C (II)

31 Gestão de Aniversários

32 Programar em Clipper (II)

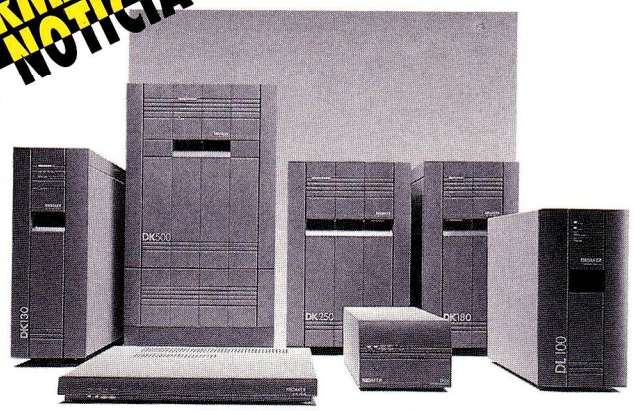
35 Livros

36 Livre. Fast

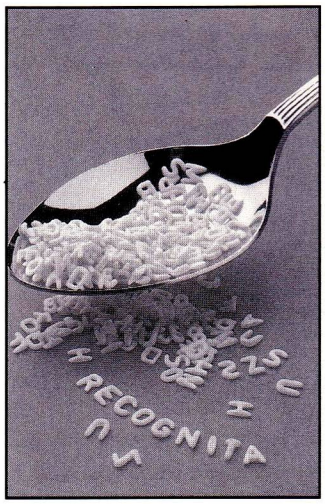
37 A Dinastia Intel (II)

40 Montra Spooler

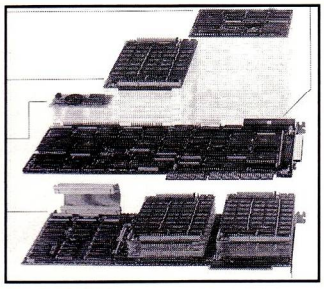
INFORMÁTICA EM NOTÍCIA



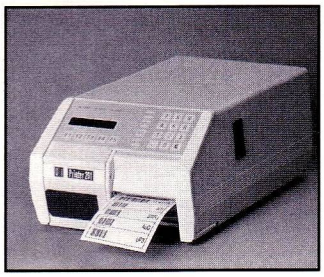
A gama de Unidades de Alimentação Ininterrupta da Daker



Recognita – Software de OCR



Controladores avançados DPT



UBI Printer 201

DATACOMP ANUNCIOU NOVOS PRODUTOS

A Datacomp, Sistemas de Informática Lda, apresentou recentemente à imprensa vários produtos que passa a representar no nosso país. De entre eles destacamos os produtos da DPT (Distributed Processing Technology), empresa considerada como um dos líderes mundiais no fabrico de controladores de disco que optimizam o seu desempenho e compatibilidade, tais como os premiados SmartCache, SmartConnex (um controlador SCSI de baixo custo para computadores ISA e EISA) e o SmartCache Plus.

Também foi anunciada a representação das Unidades de Alimentação Ininterrupta DAKER, que aliam a um atraente *design*, elevadas características técnicas.

Um sistema de leitura óptica de caracteres inscritos em cheques, recibos e outros documentos, da marca italiana SEAC BANCHE, passaram também a ser reprentados pela Datacomp, a fim de satisfazer as solicitações do mercado bancário.

Recognita Plus, outra das novidades anunciadas, é um programa OCR – ou seja, Reconhecimento Óptico de Caracteres. Recolhendo a imagem através de um *scanner* , este software permite a conversão da imagem de qualquer documento em papel, em caracteres, possibilitando o seu arquivo, tratamento e reformatação.

De particular interesse, este software suporta mais de 80 línguas, e opera sob o ambiente DOS e Windows 3.x. Este software dispõe ainda de da aplicação AUGE, destinada à utilização por deficientes visuais.

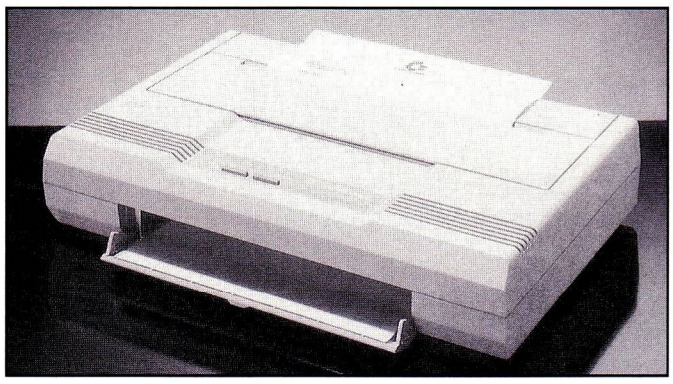
UBI – na categoria das impressoras/ etiquetadoras foi outra das representações anunciadadas.

SITAC – NOVA EMPRESA DO GRUPO «CATS»

CATS – empresa de informática, que para além dos seus serviços normais é possuidora da melhor (e maior) BBS em funcionamento no nosso país, anuncia a criação da SITAC – Sistemas Informáticos, Tecnologia, Assistência e Consultadoria, Lda., sediada na Rua Jacinto Nunes, 18 - r/c Dtº.. Telefs.: 815 51 39/815 51 06, Fax: 815 42 12.

Com esta nova empresa o Grupo Cats pensa poder responder de forma mais directa e eficaz às exigências e solicitações do mercado, especializando-se no desenvolvimento de software de qualidade.

COMMODORE MPS 1270 SILENCIOSA E BARATA



Com um preço incrivelmente baixo de 26 990$00 (+IVA), esta impressora de jacto de tinta, possui uma resolução máxima de 240 pontos/polegada, velocidade de impressão de 160 c. p. s. e 5 fontes residentes. Usa papel contínuo ou folha-a-folha. As melhores características na relação preço/qualidade.

SISTEMA DE IDENTIFICAÇÃO DA PRINTOPLAST

Este sistema – que emite um cartão em 40 segundos, com fotografia a cores de grande qualidade – utiliza captação óptica de imagem com uma definição de 640x580 pixels e tem como unidade central um PC compatível 80386SX a 16 MHz (podendo como opção, recorrer a um 30386SX a 33 MHz ou a 80486 a 25 MHz).

Possibilitando a ligação – através de redes de dados Novell, Arcnet ou Ethernet – a sistemas informáticos já existentes, o «Digital Video Image System», permite a produção muito rápida de cartões de alta qualidade, segurança e durabilidade.

Representado por Equisistemas, Lda.



TOPIS INTERNACIONAL NOMEADA REVENDEDOR AUTORIZADO NOVELL

Tendo como objectivo aumentar a eficiência e a eficácia das soluções apresentadas aos seus clientes, a TOPIS INTERNACIONAL assinou com a Portulan um acordo de colaboração respeitante à distribuição de produtos Novell.

Deste modo, os revendedores TOPIS passam a poder adquirir os produtos Novell em condições vantajosas em relação à concorrência, passando igualmente a dispor de um apoio técnico e comercial de segunda linha, assegurado pela Portulan.

O acordo foi assinado por Luis Maggiolli, Director Geral da Topis Internacional e por Paulo Silva, Director Comercial da Portulan.

A colaboração entre as duas empresas estender-se-á dentro em breve à área de forrnação, um dos projectos que a TOPIS encara com grande entusiasmo no sentido de contribuir para um mais consciente aproveitamento das potencialidades da informática, entre nós, nomeadamente nos diversos sectores empresariais que a utilizam.



Os **Nootebooks** *da série Epson NB-SL*



Impressora Laser Epson EPL-8100

NOVIDADES EPSON

Notebooks **da gama NB-SL já estão disponíveis em Portugal**

A EPSON anunciou uma nova gama de notebooks – NB-SL – que integra as mais recentes tecnologias.
Os *notebooks* da série NB-SL estão equipados com os microprocessadores i386SL, havendo um modelo que funciona a 20MHz (NB-SL/20) e outro a 25 MHz (NB-SL/25).
A memória de base do NB-SL/20 é de 2MB (extensível a 14MB, através de expansões de memória de 2, 4 ou 8 MB). 0 NB-SL/25 apresenta uma memória *standard* de 4MB (extensível a 20MB).
A Série NB-SL possui uma gama de discos amovíveis (com capacidades que podem ser aceites pelos PCs de secretária da Série EL entre 40 e 80 MB), (Kit opcional).
Equipados com ecrã de cristais líquidos VGA do tipo FTN *(Film Twist Némátique),* os computadores da Série NB-SL permitem uma visualização de 64 níveis de cinzento, podendo posteriormente ser substituído por um ecrã policromático.

Impressora laser EPL-8100

Com uma velocidade de 10 ppm (páginas por minuto) – e a edição da primeira cópia em 18 segundos – a EPL-8100 dispõe de um processador 68000, a 16,6 MHz.
A EPL-8 100 apresenta uma resolução de 300 ppp (pontos por polegada), valorizada pela inclusão da tecnologia RIT *(Resolution Improvement Technology)* e da qualidade de impressão proporcionada por Tonner Ultra Fino e *Micro Art Printing*, de que resulta uma edição de elevada qualidade com negros bem definidos e uma resoluçáo equivalente a 600 ppp.
Apresentando *interfaces standard* paralela e série RS-232C e RS-422, a EPL-8100 pode adoptar (opcionalmente) outras *interfaces*, incluindo AppleTalk, para utilizadores de sistemas Apple Macintosh.

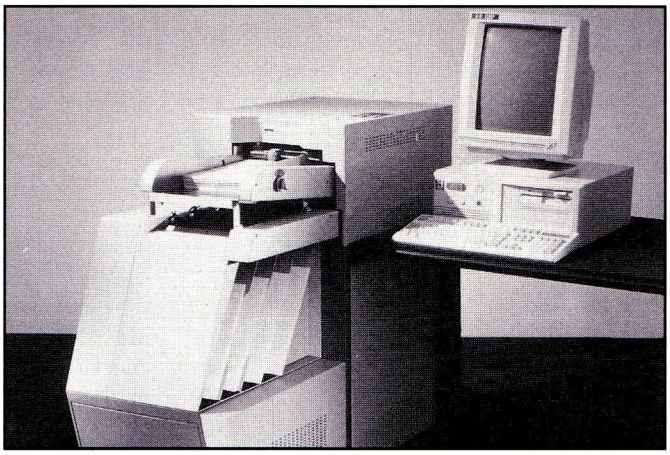
Também disponível em versão PostScript (EPL-8 100/2 MB PS) – através de um cartão opcional. Este cartão necessita de 2 MB de memória e é complementar aos quatro modos já existentes. Além disso, é de referir que a EPL-8100 detecta automaticamente se os dados enviados são em modo PostScript ou não-PostSript (PCL5, GQ, LQ ou FX). Podem utilizar-se simultaneamente dois modos (como o PostScript e PCL5) duma forma totalmente transparente e com grande facilidade. A EPL-8100 é partilhável até três utilizadores, através de detecção automática.

Impressora LQ– 100

Foi, também, lançada no mercado nacional a impressora LQ-100 de 24 agulhas e tecnologia ESC/P2 – por apenas 75 000$00.
Uma das características desta impressora é a sua pequena dimensão, que é ainda valorizada pelo facto de poder funcionar na posição normal ou colocada na vertical.
De modo a melhorar o conforto do ambiente de trabalho, o nível sonoro da LQ-100 foi substancialmente reduzido em relação às impressoras clássicas.
A utilização da LQ-100 é muito facilitada pela inclusão de um novo painel de controlo, que apresenta ao utilizador várias opções, que este vai seleccionando através de respostas SIM/NÃO. Assim, a introdução dos parâmetros pode ser feita, quer de forma interactiva, quer directamente a partir do computador e dum utilitário.
Beneficiando do *know-how* EPSON – associado a um esforço conjunto EPSON/utilizadores – a LQ-100 não fica a dever nada aos modelos profissionais. Com efeito, ela abandona o princípio da alimentação folha a folha em benefício da cassete de papel para 50 folhas (herdada das impressoras laser), que está incluída no preço indicado.



Impressora Epson LQ-100



LEITOR ÓPTICO CMK 2004

A Equisistemas anunciou o lançamento do leitor óptico CMK 2004, com o qual inicia a comercialização em Portugal dos equipamentos da Kobrak.
O CMK 2004 é um leitor para «captura» de dados, a partir de textos editados ou escritos à mão, com o consequente registo electrónico.
Controlado por um IBM PC, ou compatível, o CMK 2004 reconhece caracteres OCR-A, OCR-B, OMR e outros editados por impressoras ou manualmente.
Podendo ler documentos com dimensões até 210 x 356 mm – que vai encaminhando para diversos cacifos, previamente seleccionados – o leitor CMK 2004 é especialmente indicado para os sectores de Seguros, Banca, Instituições de Segurança Social e Saúde, Administração Pública, companhias de electricidade e de telefones, etc.
A velocidade de leitura e encaminhamento de documentos situa-se entre os 500 e os 1400/hora, consoante a sua dimensão.

O CONTEÚDO DA DISQUETA

Mais uma vez tivemos de recorrer a umas «habilidades» para conseguir encaixar na disqueta o máximo possível de programas. Acabaram-se os directórios. Aliás desnecessários, pois os «compactados» encontram-se devidamente identificados nas linhas que se seguem. A disqueta deste número inclui um programa que foi o principal responsável pelas nossas «dores de cabeça». Trata-se do CLICKMENU, um utilitário para Windows escrito em Visual Basic, mas que exige a presença de um *Run Time* denominado VBRUN100.DLL com a módica dimensão de 271.264 bytes. Mas é com muito agrado que publicamos este programa, o primeiro utilitário para Windows apresentado na **Spooler**.

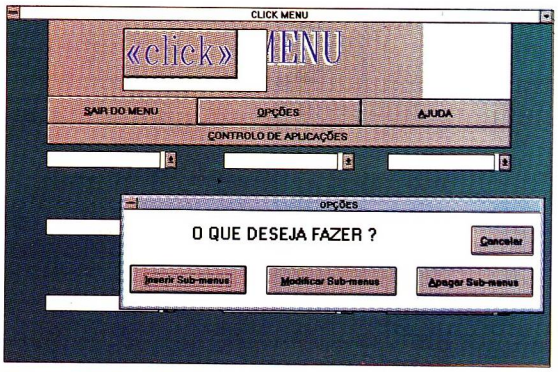
Passemos, então, à descrição do conteúdo da disqueta:

ANOSEXE.EXE

Um interessante utilitário escrito em Clipper que, quando carregado no arranque do sistema, nos avisa se alguém faz anos «hoje». Veja-se o artigo para a descompactação, instalação e utilização do programa.

BIN.EXE

Construa os seus próprios módulos binários para integração em programas escritos em dBASE III Plus ou dBASE IV. Todas as explicações encontram-se no artigo do Eng. José Páscoa.



WINMENU2.EXE

A descompactação deste ficheiro dará origem ao MENU.EXE e ao *Run Time* VBRUN100.DLL. Ambos são para instalar no seu directório Windows. Depois... leia o artigo, se faz favor, e delicie-se!

COMPRIME.EXE

Este conjunto destina-se aos «fãs» dos utilitários para compressão e descompressão de ficheiros. Com ele tudo se torna mais fácil para o utilizador. A sua instalação obedece a uns certos requisitos, pelo que deverá ler o artigo com toda a atenção.

C_2.EXE

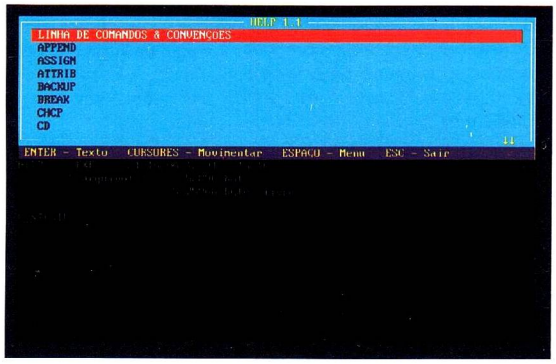
Tal como o nome sugere, trata-se do autodescompactável contendo o material objecto da Parte II do curso «Programar em C».

RATOEXE.EXE

Mais uma lição do Eng. Paulo Bernardo sobre Pascal. Desta feita sobre a inclusão das funções necessárias para o funcionamento do rato. O artigo e o ficheiro .PAS explicam tudo.

EXCLUI.EXE

Descompacte-se e «exclua-se» o que não se quizer apagar. É indispensável ler o artigo a fim de se evitarem apagamentos indesejados...



HELPEXE.EXE

O nosso colaborador Carlos Ladeira conseguiu, finalmente, ver publicado o TSR com que, em tempos, concorreu ao passatempo «Programas Didáticos». O HELP instala-se em memória, fica residente e pode ser invocado sempre que necessário. Infelizmente, o espaço na disqueta não permitiu a inclusão das fontes. Mas dada a sua importância em termos didáticos, prometemos publicá-las no próximo número.

FASTLIVR.EXE

FAST e LIVRE são os dois utilitários do conjunto de quatro que o Nelson Pires nos enviou. No próximo número publicaremos os restantes, TECLADO e QCD, bem como as fontes respectivas. O artigo explica para que servem e como se utilizam.

Grafiteria pub

SENSACIONAL CAMPANHA

MOUSE ABC

3 Teclas, com tapete: Software de instalação e adaptador de 9/25 pinos
4 resoluções 100/200/400/800 Dot

de 5 a 10 unid.	+10 unid.
3 800$00	3 500$00

DISKETES

	de 10 a 50 Cx.(Cx. de 10)	+ 50 Cx.
ATLANTIC 3,5" DD	1 100$00	1 000$00
ATLANTIC 3,5" HD	1 850$00	1 750$00
SKC 5,25" DD	880$00	800$00
SKC 5,25" HD	1 300$00	1 200$00
SKC 3,5" DD	1 300$00	1 200$00
SKC 3,5" HD	2 300$00	2 200$00

COMPUTADORES da marca ABC

MINITOWER 386SX – 25 Mhz

- 2 Mb RAM
- Drive 3,5" a 1.44" Mb
- Hard DisK 80 Mb a 15 ms
- 2 Portas Série RS232
- 1 Porta Paralela
- Teclado português
- Sistema Operativo MS. DOS 5
- Monitor Super VGA Policromático (1024×768)

PREÇO: 239 000$00

BIGTOWER 486 – 33 MHz

- 4 Mb RAM
- 256 Kb Cache memória
- Drive 3,5" a 1.44" Mb
- Hard Disk 211 Mb a 15 ms
- 2 Portas Série RS232
- 1 Porta Paralela
- Teclado português
- Sistema Operativo MS. DOS 5
- Monitor Super VGA Policromático (1024× 768)

PREÇO: 480 000$00

16% de IVA a incluir a todos os preços indicados

Software

DIGA-NOS O QUE PRETENDE,

NÓS TEMOS A SOLUÇÃO PARA O SEU CASO*.



*Só temos Software original

COMPUTADORES PORTÁTEIS

TOSHIBA	ASI
Mod. 2200 SX	Mod. Premium Exec
Processador 386 SX	Processador 386 SX
Velocidade 20 MHz	Velocidade 25 Mhz
Memória RAM 2 Mb	Memória RAM 4 Mb
Disco 60 Mb	Disco 63 Mb
Ecran VGA	Ecran LCD Super VGA
Bateria Aut. 2,5 H	Bateria Aut. 3 H
Resolução (800×600)	Resolução (1024×768)
MS. DOS 4.01	MS. DOS 5.01
PREÇO: 580 000$00	**PREÇO: 595 000$00**

AGENTES CASA VIOLA

BRAGA:
Av. Central, 85 – 1º
Telefs.: 72 798-74 369
S: JOÃO ESTORIL:
Av. Florinda Leal, Lote 1-A
Telef.: 467 07 33
VISEU:
Rua Direita, 79 – 1º
Telef.: 27 664
PORTIMÃO:
Rua D. Carlos I
Telefs.: 83 653-85 670
SETÚBAL:
Largo da Misericórdia, 28
Telef.: 31 432

abc A VERDADE EM SOLUÇÕES INFORMÁTICAS

Rua dos Sapateiros, 160 – 2º — 1100 LISBOA
Telef.: (01) 342 44 48/347 80 94/347 78 96 – Telex: ABCINF 13348 P — Fax: 346 22 40/342 67 22

Correio dos Leitores



Por Renato Casquilho

Ao ler a Spooler Nº 9 encontrei uma mensagem que dizia que para ter acesso aos dados contidos na disqueta bastava colocá-la na drive e ligar o computador. Na Spooler Nº 16 não encontrei essa mensagem...
A partir de que número começaram os artigos «Programar em Pascal»?...
Se alguma vez tiver uma pergunta rápida, posso fazê-la pelo telefone?...
Aproveito para dizer que a película plástica que cobre a capa da Spooler descola-se facilmente...
Existe alguma maneira de criar um ficheiro batch a partir doutro?...Por ex.: Se o conteúdo do ficheiro for:

```
C:
COPY CON SPOOLER.BAT
CD\DISKETTE
DISK
```

Quando for a executar o comando COPY CON SPOOLER.BAT o computador pede-me para entrar o conteúdo do ficheiro. De que maneira poderia eu indicar dentro do ficheiro o conteúdo do ficheiro SPOOLER.BAT?«...
Para quem nem sequer tem ideia do que é programar em dBASE III Plus, acham que o livro «dBASE III Plus - Programação Avançada» será demasiado difícil?...
No programa FILATECA...depois de escolhermos uma opção já não podemos voltar atrás e se carregarmos em ESC o programa emite a mensagem «DO CANCELED»...Se mandarmos imprimir e a impressora não está pronta, acontece o mesmo...

Pedro Andrade – Barreiro

Com efeito, até ao número 12, a disqueta **Spooler** continha os ficheiros de sistema necessários para o arranque. A partir do Nº 13 foi-lhe retirada essa caracterísca – em benefício do espaço útil – pelo que o computador já deve ter sido arrancado antes da leitura da disqueta.
Os artigos «Programar em Pascal» iniciaram-se no Nº 10 e terminaram no Nº 14.
Os nossos leitores podem sempre telefonar-nos para perguntas «rápidas». Só não podemos garantir uma resposta imediata – depende de estarmos, ou não, em condições de o fazer. Mas podemos garantir que faremos tudo para encontrar a resposta adequada.
Talvez os primeiros números padecessem dessa «descolagem» quando ao sol ou na humidade. Actualmente não sucede.
As restantes perguntas vão ser respondidas pelo autor do FILATECA:
«É, na realidade, extremamente simples, mas não com o uso do comando «COPY CON SPOOLER.BAT». É que, ao activar esse comando, o controlo passa para o COPY, e é impossível, através dum batch, enviar dados para o SPOOLER.BAT usando o COPY CON. A melhor (e mais fácil) maneira de, a partir de um batch, se criar outro batch (neste caso o SPOOLER.BAT) é, muito simplesmente, usar o comando ECHO, redireccionando-o para o SPOOLER.BAT. Exemplificando:

```
ECHO (linha1) >> SPOOLER.BAT
ECHO (linha2) >> SPOOLER.BAT
ECHO (linha3) >> SPOOLER.BAT
.....
ECHO (linhaN) >> SPOOLER.BAT
```

... em que (linha1), (linha2), etc, são as linhas a enviar para o batch SPOOLER.BAT. Assim, se fizermos um batch com o seguinte conteúdo:

```
ECHO C: >> SPOOLER.BAT
ECHO CD\SPOOLER3 >>
SPOOLER.BAT
ECHO SPOOLER >> SPOOLER.BAT
```

teremos criado um batch SPOOLER.BAT contendo:

```
C:
CD\SPOOLER3
SPOOLER
```

a partir doutro batch.

«E para o computador premir a tecla F6 (ou outra qualquer) sem eu ter de utilizar o teclado? Existirá algum comando tipo PRESS F6?«

O comando ECHO faz exactamente isso. Se fizermos num batch «ECHO A» o computador agirá como se tivesse sido premida a tecla «A», imprimindo-a no ecrã, ou enviando-a para qualquer outro ficheiro ou dispositivo (como acontece no exemplo dado com a criação de um batch a partir de outro).
Em relação ao caso preciso da tecla F6, não se poderá escrever ECHO F6, já a tecla F6, em MS-DOS representa o ^Z (Control + Z). Assim, deverá sim ser escrito ECHO e depois ser premida a tecla Control e a Z simultaneamente. De qualquer modo, não me parece que isso seja de grande utilidade, já que o problema da criação de ficheiros batch a partir de outros é resolvido, não com o COPY CON (para o qual serviria esse F6) mas com o ECHO, com o qual não é necessário o F6 para marcar o EOF (End Of File - Fim do Ficheiro).
Na realidade, quando criei o FILATECA, não achei necessário criar uma opção de retorno ao menu principal, a não ser na opção «Apagar todo o ficheiro», devido ao perigo que esta opção apresenta. Nas outras opções os enganos na escolha feita pelo utilizador são facilmente corrigidos. A única opção em que poderia ter interesse que existisse uma opção «Voltar ao menu principal» seria na opção 2, Listar ficheiro. Terei isso em conta numa próxima oportunidade. De qualquer modo, é extremamente difícil introduzir uma opção por engano, já que é sempre necessário carregar a seguir em <Enter> ou noutra tecla numérica, no caso das opções 10, 11 e 12. Quanto à tecla ESC, nem sequer me passou pela cabeça que os utilizadores a iriam premir durante a execução do programa. O próprio dBase tem muitas limitações no manejo dessa tecla. Portanto, o melhor é não a premir... Pode ser que também numa próxima oportunidade o programa venha ou a «filtrá-la» (ignorando-a) ou a fazer uso dela para exactamente voltar ao menu principal...
Bom, eu tive o cuidado de escrever «Prepare a impressora e prima qualquer tecla»... O melhor é só premir a tecla quando a impressora estiver pronta... Mas também para a próxima colocarei um comando (que aliás é extremamente simples) que nessa situação (a impressora não estar pronta e uma tecla é premida) volta ao menu principal.

... é de me informar sobre os modems, se com os modems podemos apanhar um vírus...
... como é que funcionam as BBS's...
... tenho um problema com o CPAV, quando faço install ou CPAV o computador diz-me «DIVIDE ERROR»...se é um problema com o programa ou se é incompatibilidade...
...o que é comprar programas shareware... se além do valor que está na revista temos de pagar mais e quanto...
... o que é uma placa VGA 1024 x 768, qual a sua vantagem sobre VGA's com valores inferiores...
... é necessária alguma drive especial para ler e escrever em disquete de 2HD...

Daniel Ferreira – Pombal

Antes de mais, caro leitor, lembre-se que «vírus de computador» são programas especialmente escritos para desempenharem certas tarefas, normalmente maléficas. Portanto, não se apanham vírus com os modems. O que pode acontecer, isso sim, tendo um modem e um programa de comunicações, é fazer um *download* (vulgo «puxar ficheiros») e um desses ficheiros vir infectado com um vírus. Claro que o vírus só se manifestará quando correrrmos esse ficheiro. Medida profilática: antes de correr o ficheiro desconhecido testá-lo com um bom programa anti-vírus.
Já aqui falámos sobre BBS's. Para não nos repetirmos, queira ler o «Correio dos Leitores» do Nº 16 da **Spooler**.
Em princípio, o CPAV corre sem problemas em DOS 3xx ou superior. Não sabemos qual é a sua versão. Se essa mensagem só aparece com o CPAV, então é muito provável que o programa esteja avariado.
Tal como é claramente dito nas páginas da «Montra Spooler», o que o leitor compra e paga é a disqueta e o trabalho de cópia e administrativo correspondente. Os programas *shareware* contidos nas disquetas deverão ser pagos aos respectivos autores, nas condições por eles anunciadas na documentação que acompanha os programas.
Uma placa VGA com resolução de 1024x768 pixels é normalmente conhecida por Super VGA e possui muito mais resolução que a VGA Standard (640x480). Existe ainda a VGA de alta resolução que permite 1280x960 pixels. Quanto maior for a resolução, isto é, quantos mais «pontos» permitir no ecrã, mais nítida é a imagem.
Para a utilização de disquetas de 2HD (alta densidade) é necessário um leitor de disquetas também de alta densidade (1,4 Mb em 3 1/2" e 1,2 Mb em 5 1/4").

Do nosso leitor ***Renato Oliveira,*** *de* ***Gondomar****, recebemos várias perguntas, cujas respostas são dadas pelo Eng. José Páscoa:*

... Como é que eu posso gravar ecrãs em QB 4.5 recorrendo ao endereço HB800 mas sem gravar a imagem em disco, mas sim numa variável de trabalho de tipo Byte com comprimento dum ecrã...

«O QuickBasick não é nossa especialidade. Julgo contudo que o leitor procura algo como o seguinte:

```
DEF SEG = &HB800
' $DYNAMIC
DIM A(2000) AS INTEGER
FOR i = 1 TO 2000
        A(i) = PEEK(i)
NEXT i
```

... Qual é a diferença entre memória extendida e expandida.

A resposta a esta pergunta está contida no artigo «A Utilização da Memória Extra do Computador» publicado no Nº 17 da **Spooler**.

Comecei a limpar a drive C:... e por acidente madei ao ar o STACVOL.000 e posteriormente copiei informação do E: para C: (aqui meti água) ...

Creio que efectivamente meteu alguma (muita) água...

...tentei recuperar o STACVOL.000 e lá consegui com o PCSHELL 5.0...

De facto não conseguiu, não. Quando os sectores do disco são preenchidos com informação nova, a anterior perde-se irremediávelmente. No seu caso é provável que o *header* do STACKER tenha sido corrompido. Só existe um tipo de ferramenta que lhe dá a certeza quase absoluta que a informação será recuperada: o Backup.

O meu computador tem apenas 512K de RAM e disco de 20 MB.
...Será que tenho alguma vantagem em expandir a RAM na motherboard para 2 MB e um disco de 80 ? Ou devo comprar uma nova máquina ?

Se tiver possibilidades talvez seja de investir na compra de uma nova máquina. Mas acima tudo, a decisão deverá ter em conta as suas necessidades presentes e futuras e os programas que pretenda utilizar.

...Também queria adquirir um modem, qual o melhor que se adapte à minha máquina e às linhas telefónicas portuguesas.

Para ligar para as BBS é recomendável no mínimo um modem de 2400 baud que cumpra a norma CCITT v.22 bis. Se a qualidade das transmissões telefónicas na sua área forem más, então deverá procurar um modem que possua correcção de erros (integrada no *hardware*) de acordo com a norma MNP 4 (ou superior).»

O nosso leitor ***Armando Frazão*** *enviou-nos uma carta onde critica o método de avaliação dos testes de Pascal (recordam-se concerteza do nosso pequeno passatempo do programa que escrevia números por extenso) e onde pede também ajuda para encontrar o driver do Windows 3.0 para uma card video Tseng ET3000.Também coloca algumas questões sobre o programa «DLED» e sobre Assembly. As primeiras são respondidas pelo Eng. Paulo Bernardo; as segundas pelo nosso colaborador José Orlando Pereira:*
«Comecemos pelas críticas: o programa vencedor não faz a validação dos valores de entrada e produz um runtime error ao receber um valor não numérico. É verdade, sim senhor, mas este facto não foi considerado na avaliação do programa, pois como foi dito, e citamos, «a maior ou menor qualidade da apresentação do programa no ecrã não será considerada para a avaliação global». E não foi! De facto, o que mais nos preocupou foi, além das respostas correctas, a estrutura e correcção do algoritmo de resolução
do problema, e não o interface com o utilizador.
Quanto à questão do driver que o nosso leitor nos pede, de facto não o possuimos. Porém, e uma vez que ao adquirir um computador ou mesmo uma placa de vídeo é normal receber os drivers da dita da placa, sugerimos ao nosso leitor que entre em contacto com o fornecedor do equipamento, pois estamos em crer que este terá o maior gosto em prestar-lhe esse pequeno serviço. Paulo Bernardo».
«Efectivamente, é possível detectar um programa residente em memória, examinando os bytes apontados pelo vector de interrupção por ele direccionado. No entanto, caso este teste falhe, não está garantido que o programa não esteja residente em memória. No caso do dLED, se outro programa residente que também capture a interrupção 13h, for instalado posteriormente, o vector apontará para o segundo programa, continuando o dLED instalado e perfeitamente operacional. Resumindo, esta técnica só é conclusiva quando a resposta é positiva. Para concluir a negativa seria necessário recorrer a outras técnicas, das quais a mais óbvia é a escolhida para o dLED que, neste caso, torna redundante o teste sugerido por si.
Quanto à questão da passagem de parâmetros: o DOS, ao executar qualquer programa, associa-lhe um bloco constituido por diversas informações, o *Program Segment Prewfix*, ou PSP, contendo, entre outras coisas, uma cópia exacta da linha de comando tal qual a tínhamos escrito, omitindo o nome do executável. Esta linha pode ser tratada pelo método que entendermos, tendo em conta que é terminada por *Carriage Return, New Line*, ou simplesmente 0. Em ficheiros .COM, o endereço do PSP é CS:0000, sendo o seu tamanho de 256 bytes. Em ficheiros .EXE, como podem existir vários valores possíveis para CS, é mais seguro chamar a função 62h da int.21h que, sem receber nenhum parâmetro, devolve em BX o segmento do PSP. Este valor pode depois ser movido para ES ou DS para operar. Falta apenas dizer que a linha de comando se encontra no *offset* 128 (80h) do PSP, sendo também 128 o seu tamanho máximo. É preciso ter em conta que este espaço também é a *Disk Transfer Area* por defeito, pelo que quaisquer operações que aí devolvam resultados destroiem a linha de comando. José Orlando Pereira».

O nosso leitor Pedro Ribeiro, possuidor de um computador Schneider Euro At (passe a publicidade) colocou-nos algumas questões relativas ao hardware da sua máquina. Passo a palavra ao Eng. Paulo Bernardo:
... Posso adicionar outra placa VGA com mais memória sem que entre em conflito com o VGA original?...
«É tecnicamente possível instalar duas placas de vídeo no computador, se bem que dependa das capacidades de configuração das placas e da máquina a coexistência pacífica de todo este material. Em nossa opinião, o ideal será substituir pura e simplesmente uma placa por outra, pois não descortinamos qualquer vantagem em possuir duas placas instaladas em simultâneo.
...Quanto a memória RAM, o meu computador tem 1Mb mas eu gostaria de a aumentar para 2 ou 3Mb; o representante disse-me que cada Mb custava cerca de 35000$00; entretanto eu vi anúncios de firmas que vendem memórias por cerca de 9000$00 por MB; será que poderei comprar as de 9000$00?...
Não há qualquer razão para que as memórias mais baratas não funcionem bem no seu computador. Os únicos cuidados a ter são relativos à velocidade de acesso das RAMS (não se podem instalar RAMs mais «lentas» que as apropriadas à máquina em questão) e à configuração das ditas RAMS (saber se são SIMMs, SIPs, etc). Uma possível razão para que seja tão caro o serviço de expansão da máquina pode ter a ver com o facto de, eventualmente, esta não ser expansível on-board e necessitar de uma card de expansão o que encarecerá significativamente o trabalho. Mas atenção: se for este o caso, não será possível expandir a memória da máquina sem a dita card!

... O que é o «parity checking«?...

É o teste de paridade. Usam-se as rams de parity checking para associar 1 bit de paridade a cada byte de ram. A paridade serve para detecção de erros. O teste de paridade pode ser desligado através de um jumper (interruptor) especialmente existente para o efeito. Normalmente as RAMS de parity checking não vêm instaladas, por questões de economia.Paulo Bernardo».

... Quando instalei o MS DOS 5 alguns jogos que tinha deixaram de funcionar, antes tinha o MS DOS 4.01. Será que a compatibilidade entre o DOS 5.0 e as versões mais antigas não é 100 % ?...
... Será que existe a possibilidade de o PKZIP ou o LHARC pedirem mais disquetes no caso de o programa a compactar ser muito grande...
... o STACKER. Ouvi dizer que ele duplica, ou triplica o espaço em disco. O que é ? O que faz ? Como faz ?...
José Leça – Esmoriz

As respostas a este nosso leitor são dadas pelo Eng. José Páscoa:
«O leitor poucas pistas nos dá que nos possam ajudar a avaliar correctamente o que se possa estar a passar, designadamente qual a sua configuração em termos de AUTOEXEC.BAT e CONFIG.SYS, o equipamento que possui e também quais os programas que não consegue executar com o MS-DOS 5.0. De qualquer modo e em termos gerais podemos afirmar que o MS-DOS 5.0 é totalmente compatível com todas as suas versões anteriores e desde a 1.0! As duas causas principais de incompatibilidades que conhecemos residem em anomalias dos próprios programas e são:
1) Programas que foram concebidos tendo em conta caracteríticas próprias (designadamente funções não oficialmente documentadas) de algumas versões do MS-DOS as quais não se mantiveram nas versões posteriores. Num caso destes dever-se-á tentar obter uma actualização do programa em causa.
2) Muitos programas, jogos incluidos, foram produzidos em Microsoft C e os módulos ligados com o programa LINK.EXE utilizando o switch /EXEPACK (existe também um utilitário com esse nome que produz o mesmo efeito) o qual possiblita uma pequena diminuição do tamanho do programa executável resultante.
Tudo bem até aqui, só que a rotina de descompressão tem um comportamento desastroso quando é carregada nos primeiros 64 KB de memória. Como com o MS-DOS 5.0 é comum ter-se mais de 620 KB de memória convencional livres para as nossas aplicações então esses programas normalmente não executam, penduram o sistema ou na melhor das hipóteses dão o erro *Packed file corrupt.*. Experimente carregar o programa com o utilitário LOADFIX.COM que acompanha o MS-DOS.
Se conseguir, então tem o seu problema solucionado. Poderá então tentar obter nas BBS o *package* de domínio público LZEXE cuja ultima versão que conhecemos é a 0.91. Desse *package* faz parte o programa UPA-

BBSpooler

☎ **(01) 475 22 89**

Dias úteis: das 18.30 às 9.00 horas
Feriados e fins-de-semana: 24 horas

Correio dos Leitores

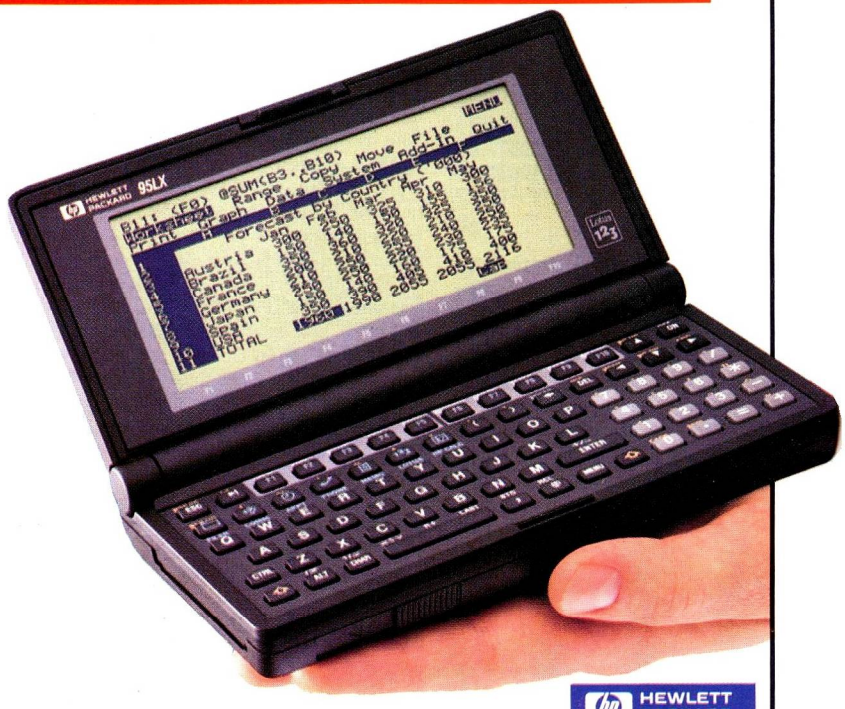
CKEXE.EXE, que elimina o efeito do EXEPACK. As versões mais recentes que conhecemos destes dois programas de arquivo com compressão, PKZIP 1.10 e LHA 2.13, não dispõem dessa possibilidade.
Contudo um outro, relativamente recente, de nome ARJ cuja última versão é a 2.30 está a ter um sucesso bastante grande pois além de permitir obter uma taxa de compressão algumas vezes superior aos acima citados, possui capacidade de arquivo para múltiplas disquetes.
O STACKER cria um ficheiro gigante (com o nome STACVOL.DSK) que é acedível como uma drive lógica. Para quase (atenção ao quase) todos os efeitos esse ficheiro (ou melhor drive) não se distinguirá de uma verdadeira partição que se tivesse efectuado no disco rígido embora nos mereça mais alguns cuidados em termos de manutenção que esta última. O STACKER propõe por *default* para essa drive lógica uma capacidade de armazenamento dupla do espaço ocupado em disco pelo STACVOL.DSK.
A capacidade de armazenamento efectivamente conseguida dependerá de vários factores, podendo eventualmente ser um pouco menor que a proposta original do STACKER, como no caso de muitos dos programas já estarem previamente comprimidos por exemplo com o PKZIP. Através do CONFIG.SYS é instalado um *block device driver* que para além de mentir todo o tempo para o DOS que o STACVOL.DSK é uma verdadeira partição ainda comprime e descomprime em tempo real o conteúdo desse mesmo STACVOL.DSK apresentando ao DOS os ficheiros que este solicita exactamente como se os mesmos nunca tivessem estado comprimidos.
A compressão do STACKER é conseguida de duas maneiras:
a) Uma compressão efectiva baseada nas teorias matemáticas dos professores Lempel, Ziv e Welch. Por razões de não perder desempenho o STACKER obtêm taxas de compressão algo inferiores às que se obteriam com arquivadores-compressores tais como o PKZIP.
b) Um aproveitamento do *slack space* do disco. Se tiver uma partição não comprimida pelo STACKER e essa partição possuir os sectores organizados em *clusters* de por exemplo 2 KB. se guardar no disco um ficheiro de 1 byte ou um outro de 2 KB o espaço ocupado será sempre de 2 KB. O que sobra designa-se por *slack space*. Com o STACKER não há este tipo de perda. José Páscoa».



O SEU PC DE BOLSO HP 95LX

Grafiteria pub

- 512 Kb a 1 Mb de ROM
- *Software*:
  - MS DOS® 3.22
  - Lotus® 1-2-3®, versão 2.2
  - Comunicação de dados
  - Editor de texto
  - Gestão de ficheiros
  - Calculadora H. P.
  - Agenda
  - Lista telefónica
- 512 Kb de RAM expansíveis até 1 Mb
- Ligação a PC, impressora, *modem*, etc.



Contacte:

DIMECRO
Representações Comerciais e Técnicas, Lda.
Rua Joaquim António de Aguiar, 41 - 6º Esqº – 1000 LISBOA
Telefones: 315 76 12/3/4/5 – Fax: 315 76 12

HEWLETT PACKARD
Distribuidor

Computadores Spooler



A **Spooler** escolheu, importou e preparou para si um computador «topo de gama» na categoria 386.

- ✔ **Processador INTEL 80386DX/33 Mhz ou 80486DX/33 Mhz**
- ✔ **4 Mb de RAM**
- ✔ **Memória «Cache» Spooler 386: 64 Kb; Spooler 486: 256 Kb**
- ✔ **Disco rígido de 100 Mb/16 ms**
- ✔ **Drive de disquetas de 3,5" – 1,44 Mb**
- ✔ **Placa SuperVGA de 1 Mb – Monitor SVGA policromático**
- ✔ **Teclado em português**
- ✔ **DOS 5.0 – Versão integral em português com manual**
- ✔ **2 anos de garantia. Assistência Técnica Qualificada**
- ✔ **Preços: Spooler 80386: 380 000$00 + 16% IVA**
  Spooler 80486: 460 000$00 + 16% IVA

FACILITAMOS PAGAMENTO ATÉ 6 MESES

Equipados com componentes de primeira qualidade, devidamente testados, a Spooler oferece-lhe duas máquinas poderosas nas classes 386DX/33 e 486DX/33.

Reserve já o seu computador **Spooler**

Rua Duarte Pacheco Pereira, 69/A – Damaia-de-Baixo – 2700 AMADORA

Telefs.: 474 60 11/41 Fax: 475 22 89

PROGRAMAR EM ASSEMBLY (PARTE II)

> **...um computador pessoal pode ser entendido como um sistema de cinco unidades funcionais...**

Afirmámos no artigo anterior que não existe melhor alternativa para aprofundar o nosso conhecimento sobre um computador do que o estudo da sua linguagem *Assembly*. Uma das razões é que a programação em *Assembly* depende em absoluto, e é indissociável, da realidade física desse computador.

Reservámos o artigo de hoje, exactamente, para fazer uma pequena apresentação desse«sujeito» que dá pelo nome de PC (e entendamos aqui por PC, o IBM PC e seus compatíveis e descendentes) pois muitas das suas peculiaridades são decisivas para se saber como «falar» com ele através do *Assembly*. Se já não tem muitas dúvidas de que é mesmo importante aprender *Assembly*, então encha-se de coragem e avance na leitura.

A ARQUITECTURA DOS PC

1- Visão geral

Muito simplificadamente, um computador pessoal pode ser entendido como um **sistema** de cinco unidades funcionais: a **unidade de entrada**, a **unidade de memória**, a **unidade aritmética**, a **unidade lógica** e a **unidade de saída**.

As unidades de entrada e saída estão em contacto com o mundo exterior ao sistema e esse «mundo» é constituído, ou pode sê-lo, pelo teclado, o monitor, o rato, o disco rígido e as *drives* de disquete, a impressora, o *modem*, uma placa de fax, um CD-ROM e, enfim, qualquer outro **periférico** presente ou que o futuro nos possa trazer.

As unidades de entrada e saída comunicam com o resto do sistema através de **portos** (em inglês diz-se *Input/Output ports*), os quais não devem ser confundidos com as conhecidas «portas» de comunicação série e paralela que todos os PC normalmente possuem e que nos permitem acrescentar ao sistema, periféricos tais como impressoras ou *modems*.

Nos PC as unidades aritmética e lógica estão agrupadas naquilo que se convencionou chamar a Unidade Central de Processamento, abreviadamente **CPU** (do inglês *Central Processing Unit*) mas mais vulgarmente **microprocessador** (ou simplesmente **processador**).

Para o PC nos ser útil necessita de actuar sobre um tipo de informação que dá pelo nome de **programas** os quais são constituidos por comandos que ele entende, e designados **instruções** as quais, quando passadas para o microprocessador de um modo **sequencial** (sequencialmente não significa obrigatoriamente que a instrução anterior esteja ordinalmente posicionada antes), vão actuar sobre outro tipo de informação designada por **dados**.

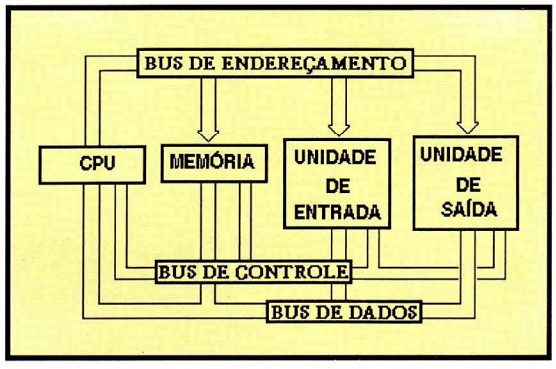
Os programas e os dados são introduzidos no sistema pela unidade de entrada e armazenados na **memória**. Os PC guardam indistintamente na «mesma» memória, tanto os programas como os dados, cabendo ao programador a responsabilidade de fazer o computador diferenciar entre os dois tipos de informação.

Todo este esquema de funcionamento foi idealizado há muitos anos por um senhor chamado Von Neumann, e por isso podemos dizer que um PC é uma máquina classificável como tendo arquitectura de Von Neumann.

Parte da memória designa-se por **RAM** (*Random Acess Memory*) e outra parte designa-se por **ROM** (*Read Only Memory*). RAM é memória que permite leitura e escrita e pode ser também chamada de RWM (*Read/Write Memory*), embora este termo tenha caido um pouco em desuso. ROM é a memória que só permite leitura. O ROM normalmente já vem de fábrica com todos os programas e dados que irá conter durante toda a sua vida (como excepção à regra existe um tipo especial de ROM, designado EPROM, cuja informação pode ser substituída com o auxílio de equipamento especial). O ROM *também é* Random Acess Memory mas não é RAM no mesmo sentido que se dá hoje a este termo!

Tanto a RAM como o ROM são constituídos por uma larga sequência de **localizações** cujo conteúdo é 1 **byte** (8 bits). Cada uma dessas localizações tem um **endereço**. O endereço é expresso por um número que define unívocamente o byte dessa localização, mas também podemos juntar 2 bytes seguidos (nos PC chama-se um **word**) ou 4 bytes seguidos (nos PC chama-se um **doubleword**) e acedê-los simultaneamente com um único endereçamento.

Nos PC as várias unidades são ligadas por um conjunto de circuitos eléctricos designados por **BUS** (há quem diga barramento ou via), os quais são constituídos por **linhas de sinal**, cada uma delas podendo transportar 1 bit de informação... Existe o Bus de Controle, o Bus de Endereçamento e o Bus de Dados.



Qualquer dado que «viaje» do microprocessador para a memória ou para as unidades de entrada e saída, ou entre qualquer das outras partes componentes do sistema, fá-lo sempre através do Bus de Dados.

Mas antes desse dado viajar, o seu endereço, no caso de se tratar de uma célula de memória ou de um porto, é colocado préviamente no Bus de Endereçamento.

Por sua vez, o Bus de Controle ajuda a coordenar «toda essa louca correria» de um lado para o outro. Para toda esta «orquestra» acertar o passo existe também um **metrónomo**, isto é, um gerador de frequências constantes designa-

do *clock generator* (traduzido normalmente por relógio).

O microprocessador e todos os outros componentes trabalham a uma frequência que é sempre uma fracção da frequência do clock.

Naturalmente que todos estes conceitos «dão pano para mangas» e a sua explicação em pormenor daria garantidamente para se escrever um livro bastante volumoso. Porém, se conseguiu apreender o que foi dito até aqui (se é que não sabia já), então possui uma base mínima para poder prosseguir.

Existem ainda outros componentes do sistema com certa importância para o programador, mas vamos deixar a sua abordagem para uma outra ocasião.

Para a iniciação à programação em *Assembly* é essencial, contudo, ter uma ideia um pouco mais aprofundada sobre dois dos componentes do sistema: o CPU (que integra as unidades lógica e aritmética) e o modo como ele está organizado, e a memória e a sua estrutura interna.

No artigo de hoje falaremos precisamente sobre o primeiro deles, o CPU.

2 – O CPU

O CPU ou microprocessador é constituído por pequenos locais de armazenamento designados **registos.** Do ponto de vista do programador o melhor enfoque a dar ao CPU é precisamente o de vê-lo como um conjunto de registos.

O microprocessador do PC original foi baptizado de **8088** e era caracterizado por ter **registos internos de 16 bits**, mas utilizar um Bus de Dados de apenas 8 bits.

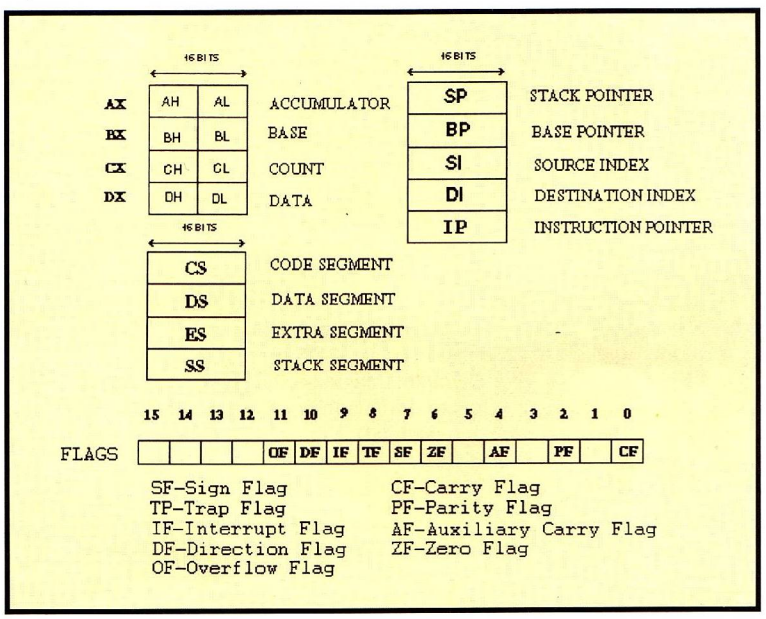
Pouco tempo depois apareceu o **8086** que era muito semelhante ao 8088, mas podia usar um Bus de Dados de 16 bits.

Na altura algumas pessoas referiam-se ao 8088 como um falso processador de 16 bits (o que lembra a polémica dos nossos dias entre o 80386DX e o 80386SX).

Contudo, a verdade é que o 8088 é idêntico ao 8086 a nível de registos internos e deve ser considerado um processador de 16 bits, apesar de a transferência de words de e para a memória ser efectuada através de 2 acessos consecutivos.

Tanto o 8088 como o 8086 têm um Bus de Endereçamento de 20 linhas (numeradas de 0 a 19) e a sua capacidade de endereçar memória é assim de 2 elevado a 20, o que corresponde a 1 megabyte, 1024 kilobytes ou 1048576 bytes.

Os registos do 8088/8086 são mostrados na gravura seguinte:



Na linha de evolução dos microprocessadores 8088 e 8086, e devido ao sucesso dos computadores que os tomaram por base, a Intel desenvolveu o **80286**, o qual também é constituído por registos de 16 bits e um Bus de Dados de 16 bits, mas possui um Bus de Endereçamento de 24 linhas.

O 80286 possui mais alguns registos que o 8088/8086 o que, aliado à capacidade de endereçamento de 16 megabytes, lhe permite trabalhar naquilo que se designa por **modo protegido.**

Contudo, mantém plena compatibilidade a nível dos mesmos registos já existentes no 8088/8086, quando trabalha em **modo real**.

Depois do 80286 apareceram os 80386 e 80486 que são microprocessadores de 32 bits e tem um Bus de Dados de 32 bits para acesso à memória e, eventualmente, também as unidade de entrada e de saída (se em arquitectura EISA), e um Bus de Endereçamento também de 32 bits. O 80386 SX possui um Bus de Dados de apenas 16 bits para acesso à memória e um Bus de Endereçamento de 24 bits, mas considera-se também um microprocessador de 32 bits pois é internamente idêntico aos outros e integralmente compatível com eles.

Todos os modelos 80386 (incluindo o SX) e 80486, e só a titulo de curiosidade, possuem um espaço de endereçamento lógico de até 64 triliões (2 elevado à 46ª potência) de bytes em estrutura segmentada, ou um máximo de 4 biliões de bytes num único segmento. E não é necessário haver correspondência em memória física para esse endereço ser produzido, o espaço em disco também serve, como se vê pelas versões 3 do Windows, porque este «milagre» é conseguido através de um mecanismo assaz complicado e que se designa globalmente por **virtualização de memória.**

Apesar destes microprocessadores melhorarem em muito a capacidade de actuação em modo protegido comparativamente ao 80286 e implementarem ainda um novo modo de trabalho designado por **modo virtual 86** (parecido com o modo real e na prática indiferenciável pelos programas executáveis sobre o MS-DOS, pelo que tudo o que dissermos sobre o modo real também se aplica ao modo virtual), acima de tudo mantiveram também a compatibilidade a nível de registos com os «velhinhos» 8088/8086 quando trabalham no modo real.

O máximo divisor comum de todos os processadores da família é, como já verificou, o trabalho em modo real. E, como nos nossos dias, 95% dos programas existentes foram feitos para trabalhar apenas em modo real, é extremamente importante começar por conhecer como trabalham os microprocessadores desta família nesse modo.

Nos PC as unidades aritmética e lógica estão agrupadas naquilo que se convencionou chamar a Unidade Central de Processamento

Deixemos apenas a salvaguarda que os 80386 e 80486 possuem ainda outros registos internos que podem ser utilizados em modo real e que não existiam nos modelos anteriores; contudo, na prática, os programas que pretendam correr em todos os computadores da família não deverão fazer uso deles.

Para efeito do nosso estudo os registos podem ser divididos em 4 grupos tal como se vê na figura à esquerda, e assim teremos:

a) Registos de utilização geral

Designam-se por **registos de utilização geral** os registos AX, BX, CX e DX.

Cada um deles é por sua vez subdivisível em dois registos, com-

preendendo quer a parte dos últimos 8 bits (parte alta) quer a parte dos primeiros 8 bits (parte baixa) do registo original.

Por exemplo, à parte alta do registo AX dá-se o nome de registo AH (*Accumulator High*) e à sua parte baixa dá-se o nome de registo AL (*Accumulator Low*). E de modo idêntico seria para os outros registos deste grupo.

O programador é relativamente livre de escolher, com alguma arbitraridade, qualquer um destes registos para muitas operações aritméticas e lógicas, mas existem contudo grupos de instruções em que tal já não pode acontecer. Isto significa que, embora haja uma certa flexibilidade, os registos de utilização geral têm uma certa vocação.

Assim o AX é vocacionado para multiplicar e dividir, o BX é o único neste grupo que pode conter um endereço de memória para aceder a dados, o CX é utilizado como contador de iterações em várias instruções e o DX é o único que pode especificar o endereço de um «porto» nas instruções que permitem o seu acesso.

“ Designam-se por registos de utilização geral os registos AX, BX, CX e DX ”

b) Registos Ponteiro e Registos de Indexação

Quando a Intel iniciou o projecto do 8088, partiu da premissa que as linguagens de programação de alto nível com futuro nos PC seriam aquelas que, como o Pascal (a linguagem C também, mas na altura ainda era pouco popular), passam os dados às «subrotinas» através do **stack** (também designado por **pilha**) e que usam também esse stack para a colocação das variáveis locais da subrotina.

Nessa linha de pensamento tinha muita lógica que o microprocessador tornasse a «vida mais fácil» aos compiladores dessas linguagens, disponibilizando registos suficientes para uma correcta manipulação dos dados existentes no stack.

Para quem ande um pouco «arredado destas coisas»: nos PC o stack é uma área da memória que o sistema operativo ou o programa em execução reservou primáriamente para o microprocessador poder guardar os endereços de retorno das subrotinas, mas que pode também servir para as finalidades acima indicadas.

O stack funciona como uma «pilha de pratos que estão para lavar na cozinha». O que chegou em último lugar está no topo da pilha e lavar-se-á primeiro (e o primeiro da pilha pode ter muito que esperar até ser lavado).

Curiosamente, nos PC a pilha está virada ao contrário e o seu **topo** tem, por conseguinte, o endereço de memória mais baixo. Para saber qual é o topo do stack o processador utiliza um registo ponteiro de nome SP (*Stack Pointer*).

E para saber qual é o endereço de uma área dentro do stack que contém os dados passados à subrotina (ou as variáveis locais), o processador socorre-se de outro registo ponteiro de nome BP (*Base Pointer*).

Uma das primeiras acções que muitas subrotinas produzidas em linguagens de alto nível executam (em programa já compilado naturalmente) é passarem para o registo BP o valor do topo do stack, de modo a poderem aceder aos dados no stack, e fazem isso copiando para esse registo o conteúdo de SP.

Mas, muitas outras vezes, nós não queremos manipular dados no stack, mas sim em áreas da memória especificamente reservadas para dados.

Podemos, por exemplo, desejar efectuar uma movimentação de um grupo de dados entre duas dessas áreas. Para essa finalidade existem dois registos classificados como registos de indexação e que são o SI (*Source Index*) e o DI (*Destination Index*).

A razão do nome «indexação» será posta em evidência quando estudarmos as instruções de movimentação de *strings*.

Tanto os registos Ponteiro como os de Indexação trabalham em conjunto com os registo de Segmento, que serão explicados mais abaixo.

Por último, e ainda neste grupo, temos o IP (*Instruction Pointer*) que guarda o endereço da instrução que vai ser executada dentro do registo de segmento de código CS.

O IP faz parte da unidade de controlo do microprocessador e os programas não podem alterar directamente o seu valor, embora possam contudo fazê-lo indirectamente com instruções que alterem a sequência da execução de um programa, isto é, instruções que provoquem saltos para outros locais dentro do programa.

c) Registos de Segmento

Muitas pessoas desistem à primeira do estudo da linguagem *Assembly*, não por qualquer eventual dificuldade no entendimento do conjunto de instruções *Assembly*, que é extraordinariamente simples e muito mais reduzido do que o de qualquer linguagem de alto nível. Também não é sequer pela necessidade de ter de conhecer as directivas que é necessário dar ao Assemblador para que ele entenda bem o que nós pretendemos.

O que muitas pessoas tem mais dificuldade em assimilar é que a memória é tratada pelos processadores desta família como estando dividida em compartimentos de 64 KB (em modo real, pois em modo protegido podem ser maiores ou menores), designados, muito naturalmente, por **segmentos**.

A necessidade dos segmentos radica num facto muito simples.

A Intel pretendia que o microprocessador 8088/8086 podesse endereçar 1 MB (que corresponde a 2 elevado à 20ª potência) de memória, mas este processador dispunha apenas de registos de 16 bits que, no máximo, permitem construir um endereço compreendido entre 0 e 65535 (isto é uma quantidade de endereços correspondentes a 2 elevado à 16ª potência).

Como naquele tempo não se tornava viável a construção de um processador com alguns registos de 20 bits (pelo menos os que endereçam memória), encontrou-se uma solução alternativa e que, embora de «grande engenho e arte», não deixa de ser um pouco confusa mesmo para quem já tenha alguma prática de programação em *Assembly* noutros tipos de microcomputadores.

Nos PC cada posição de memória é determinada por um conjunto de 2 registos, o primeiro dos quais é um dos registos (e existem 4 deles no 8088/8086 e 80286) designados por **registo de segmento** e o segundo é um dos outros que vimos atrás como tendo capacidade para endereçar memória.

Os registos de segmento percorrem a memória a «passadas» de 16 bytes (também se diz **1 parágrafo**) de cada vez, e os outros a passadas de apenas 1 byte.

E para cada localização de memória existe sempre um registo de segmento que nos dá um valor dito **valor de segmento**, e um registo «dos outros» que nos dá um valor dito valor do **deslocamento** (ou **offset**) dentro desse segmento.

Contudo, vê-se facilmente que o mesmo local de memória pode ficar representado por mais de um par de valores segmento/deslocamento, na realidade pode ser representado por até 4096 pares diferentes.

Por exemplo, imagine-se que o registo CS (um registo de segmento designado por Code Segment) tem o valor 16000 (em decimal), e que o registo IP tem o valor 3000 (em decimal).

Isto significa nem mais nem menos que estamos a apontar (neste caso é para a instrução em execução) para o **endereço absoluto de memória** 259000 (16000x16+3000).

Imagine-se agora que CS continha 16001 e que IP

continha 2984. Se fizer as contas obtém também o endereço absoluto de memória 259000.

Uma confusão muito comum é imaginar-se que os segmentos se encontram espaçados de 64 KB, de tal modo que um computador com 640 KB de RAM possui 10 segmentos. Isso não é correcto, pois os segmentos podem estar parcialmente sobrepostos como se viu no exemplo acima. E qualquer segmento pode iniciar-se em qualquer parágrafo (múltiplo de 16 bytes) da memória.

Se neste momento está perfeitamente confuso então considere-se uma pessoa normal, pois os conceitos de segmentação da memória, registos de segmento e deslocamento dentro do segmento não são de imediato intuitivos. Encha-se de coragem e continue em frente, pois se há quem demore apenas alguns minutos a ganhar a percepção correcta destes conceitos, a maioria demora horas ou dias (quando não é mais).

Fique neste momento ciente que nem tudo é mau a respeito da segmentação; na verdade, os processadores 80286 e superiores, quando a trabalhar em modo protegido, segmentam a memória para poderem isolar programas (genéricamente designados em modo protegido por **processos**) e as suas partes umas das outras e assim implementarem a virtualização da memória e a independência dos processos.

Os PC 8088/8086 e o 80286 possuem 4 registos de segmento e são eles o CS (*Code Segment* ou Segmento de Código), o DS (*Data Segment* ou Segmento de Dados), o ES (*Extra Segment* ou Segmento Extra) e o SS (*Stack Segment* ou Segmento da Pilha).

O valor contido em cada um destes registos define uma área de 64 KB.

O CS trabalha conjuntamente com o IP para apontar para a instrução que vai ser executada pelo programa.

Do valor contido no registo CS diz-se que é o valor corrente do segmento de código.

Se o conjunto de instruções de um programa for maior que 64 KB é necessário, pelo menos uma vez, e algures dentro desse programa, alterar o valor de CS. Ao contrário dos outros registos de segmento, o valor de CS é apenas alterável indirectamente (de modo semelhante ao IP), por exemplo por instruções que provoquem um tipo de salto designado FAR (para longe).

O DS contém o valor corrente do segmento de dados.

O ES pode ser considerado também como um segmento de dados mas de carácter auxiliar para possibilitar a execução de certas instruções.

O SS é o segmento do Stack.

Nas instruções *Assembly* que endereçam memória existe sempre um segmento considerado o *default* para essa instrução.

Por exemplo, instruções que endereçam o stack fazem uso dos registos SP ou BP e assumem implicitamente que o segmento *default* é o SS. Isso significa que não é necessário mencionar o nome do segmento na instrução.

A título ilustrativo, a instrução MOV AX,[BP] significa que o registo AX ficará com o conteúdo da posição de memória endereçada pelo valor do registo BP dentro do segmento SS.

Contudo, a instrução pode ser compelida a utilizar outro segmento que não o *default*, através de uma indicação explícita do nome do segmento.

No caso acima, se escrevessemos MOV AX,DS:[BP] o registo AX ficaria carregado com o conteúdo da posição de memória endereçada pelo valor do registo BP dentro do segmento de dados DS.

Diz-se num caso destes que houve uma **derrogação** (em inglês *override*) do segmento *default.*

d) O registo Flags

Todos os processadores desta família reservam um registo cujo objectivo se assemelha ao de um painel de controle.

É constituído por **flags** (sinalizadores) que podem assumir apenas um de dois resultados, 0 ou 1. Cada flag ocupa um bit no registo Flags. As flags cumprem três finalidades genéricas:



Se o conjunto de instruções de um programa for maior que 64 KB é necessário, pelo menos uma vez, e algures dentro desse programa, alterar o valor de CS

Umas servem para prestar informação complementar sobre o que se passou na execução da última instrução (*Status Flags*), outras para controlar o comportamento do microprocessador na execução de certas instruções (*Control Flags*), e ainda outras para auxiliar internamente a operação do processador (*System Flags*).

Nos processadores 8088/8086 e 80286 o registo Flags é de 16 bits.

Como pode verificar pela figura, nem todos os bits são utilizados, pois alguns são considerados reservados para eventual utilização por futuros membros da família. Vamos conhecê-los então um pouco melhor:

CF – É uma *status flag*. Após uma operação aritmética reflecte se houve transporte do último bit, isto é, se *vai um*. Se CF for 1 representa-se essa situação por CY e se for 0 representa-se por NC.

PF – É uma *status flag*. Após uma operação lógica ou aritmética se o resultado tiver número impar de bits 1 este flag será 0 (e representa-se PO, *parity odd*), se for par será 1 (e representa-se por PE, *parity even*).

AF – É outra *status flag*. Indica se houve transporte do bit 3 (isto é o quarto bit, pois eles contam-se a partir do 0). Em caso positivo representa-se por AC e em caso negativo por NA. Este flag é pouco utilizado na prática pelos programas.

ZF – Mais uma *status* flag. Indica se o resultado de uma operação é 0.

Se for 0 o flag apresenta-se com o valor 1 e representa-se por ZR em caso contrário por NZ.

SF – Também uma *status flag* e indica se o resultado de uma operação é positivo ou negativo. Se negativo representa-se por NG e se for positivo por PL.

TF – É uma *system flag*. É utilizada pelo processador quando executa instruções «passo a passo» como por exemplo de dentro do programa DEBUG.

IF – Outra *system flag*. Sinaliza se os *interrupts* podem ser atendidos. O conceito de interrupt (interrupção) será explanado no próximo artigo.

No caso de poderem ser atendidos este flag apresenta o valor 1 e representa-se por EI (*enable interrupts*), e em caso contrário por DI (*disable interrupts*).

DF – Trata-se de uma *control flag*. Se o valor do bit for 0 (e representa-se por UP) então algumas instruções que envolvem os registo de indexação (designadas instruções *string*) incrementam automáticamente esse registos posicionando-os no byte seguinte da *string*. Em caso contrário, isto é se o bit for 1 (e representa-se por DN), os registos de indexação são decrementados.

OF – É um *status flag*. Indica se o resultado da operação aritmética ultrapassou a capacidade do byte ou word que deveria receber esse resultado. Se assim for escreve-se OV (*overflow*) e em caso contrário NV (*not overflow*).

E terminamos aqui a pequena apresentação que queríamos fazer deste componente fundamental do PC que é o CPU.

A etapa de hoje foi crucial para criar as bases necessárias para «chegar á fala» com o PC através do *Assembly* e,embora saibamos que ela se revestiu de grande dificuldade para muitos dos leitores, temos a certeza que irá facilitar em muito a compreensão do que se irá seguir nos próximos artigos.

José Páscoa

ABERTOS AOS SÁBADOS ATÉ ÀS 13 H.

Loja Spooler

Venha visitar-nos. Na Loja Spooler, vai encontrar um pouco de tudo a preços imbatíveis

Só vendemos produtos de qualidade comprovada

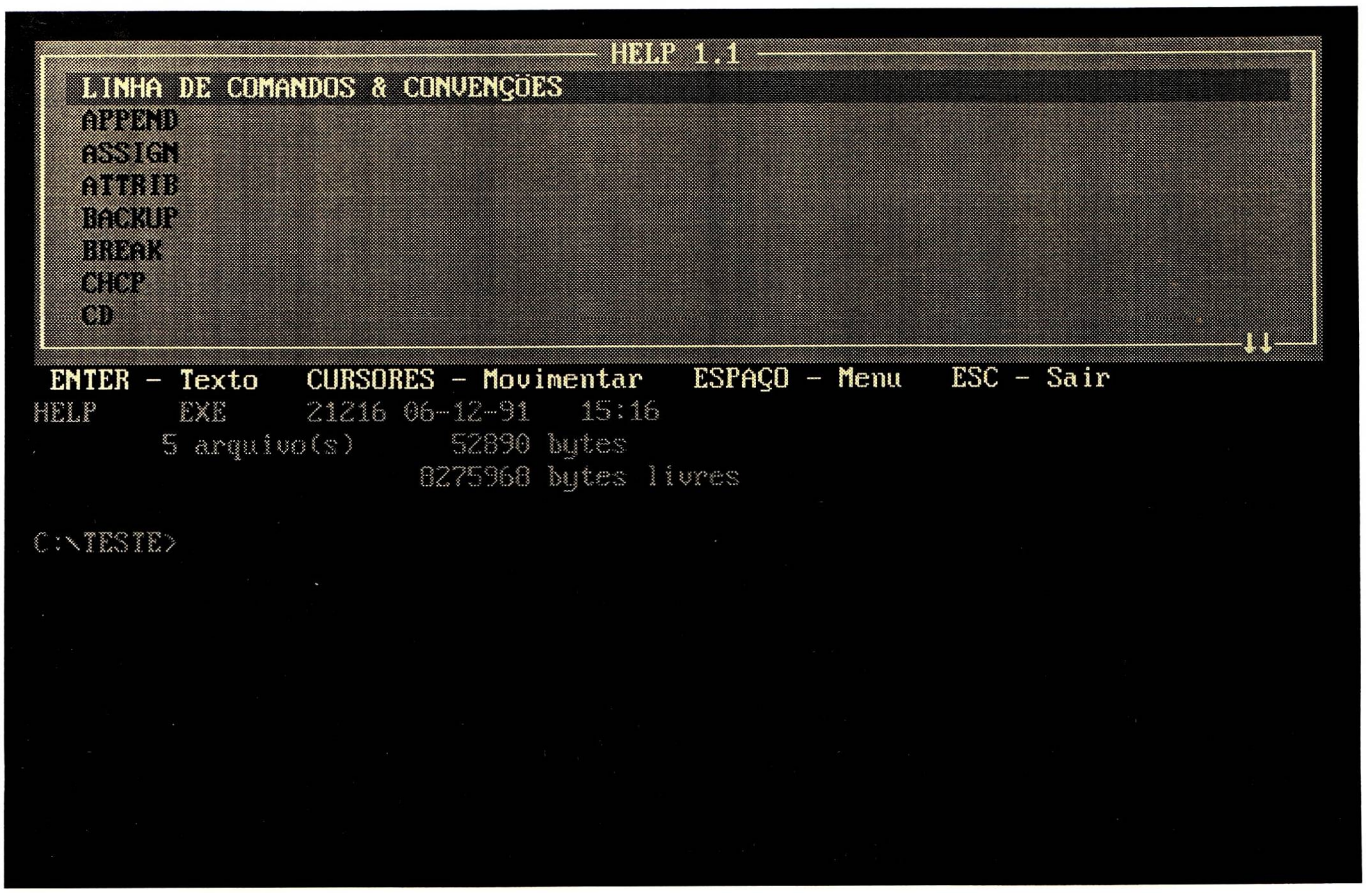
Computadores • Impressoras • Power Packs • Modems • Disquetas • Caixas arquivadoras para disquetas • Ratos • Jogos • Joysticks • Papel contínuo • Tapetes para rato, etc., etc., etc.

Rua Duarte Pacheco Pereira, 69-A – Damaia-de-Baixo – 2700 AMADORA
Telefs.: 475 22 89/474 60 11/474 60 41

DESCONTO 10% – ESPECIAL ASSINANTES

HELP

TRANSFORME O SEU PC NUM LIVRO



Quem nunca viu o seu PC rodeado de manuais de informática? Pois é, esta situação (um pouco desagradável) acontece frequentemente, por exemplo quando se está a programar numa determinada linguagem.

O programa aqui descrito pretende resolver esta situação. Basicamente é um «help» residente em memória, o qual poderá ser visualizado a qualquer momento (mesmo durante a execução de outro programa). O texto de ajuda que compõe o HELP, está num ficheiro que será indicado pelo utilizador no momento da instalação do programa em memória. Este tipo de ficheiros têm o nome de ficheiros de suporte e poderá ser o utilizador a criá-los, devendo ter em atenção a sua estrutura, que será descrita mais à frente.

Depois de instalado em memória, o programa está pronto a ser activado, bastando para isso premir simultaneamente as teclas **<ALT+CTRL+F5>**. É-nos então apresentado um menu com as palavras-chave (um ficheiro de suporte é dividido em blocos de texto que são «apontados» por palavras-chave), sendo-nos permitido escolher uma para visualizar o texto correspondente.

Para além deste HELP, existe também uma tabela ASCII que poderá ser consultada a qualquer momento. A activação desta tabela é feita através das teclas **<ALT+CTRL+F6>**.

ESTRUTURA DOS FICHEIROS DE SUPORTE

Os ficheiros de suporte contêm os dados (palavra chave + texto) utilizados pelo programa, que poderão ser consultados a qualquer momento.

Um ficheiro de suporte pode ser constituído por uma linha com a(s) palavra(s) chave precedida(s) de um "@". Na(s) linha(s) seguintes virá o texto referente a esta(s) palavra(s) chave. Toda esta sequência pode repetir-se ao longo do ficheiro. O número de palavras chave que representa o mesmo texto poderá ser qualquer número desde que não exceda uma linha de texto (uma linha de texto termina com o carácter 13, correspondendo ao ENTER no fim de uma linha). A sua estrutura básica é:

"Depois de instalado, o programa está pronto a ser activado, bastando para isso premir simultaneamente as teclas <ALT+CTRL+F5>"

> **o programa aqui descrito é um TSR, isto é, fica residente em memória depois de terminada a sua execução**

```
@pal1 [@pal2 [@pal3 [...]]]
.....
bla..bla..bla..
.....
@palN [...]
.....
alb..alb..alb..
.....
[@...
[.....]]
```

Número máximo de palavras chave: 200
Tamanho máximo de um ficheiro de suporte: 40.000 bytes

COMO CRIAR FICHEIROS DE SUPORTE

É muito fácil criar um ficheiro de suporte para este programa, pois trata-se de um simples ficheiro de texto. Segue-se um exemplo de um ficheiro criado na linha de comando do DOS:

```
C:\HELP >copy con teste.dat

@ABC@CBA
este texto é o mesmo para as duas palavras
chave ABC e CBA
asasasasasasasasasasasasassasasasasasasa
sasa
asdas s sas as a a s as asasas

fim do texto das primeiras palavras chave
@XYZ
este texto representa só XYZ

 xyzxyzxyzxyzxyzxyz
@A@B@C
abcabc abcabc abcabc abcabc abcabc
@X
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx^Z
        1 File(s) copied

C:\HELP >
```

COMO USAR OS FICHEIROS DE SUPORTE

A utilização dos fichciros de suporte é muito simples. Imagine o ficheiro criado no ponto anterior (que deverá estar na mesma directoria que o programa), para o utilizarmos como ficheiro de suporte basta executar o seguinte comando:

```
C:\HELP\ >HELP TESTE.DAT
```

(o programa fica instalado na memória com os dados contidos em TESTE.DAT).

Sempre que o programa receber o nome de um ficheiro existente como parâmetro, vai escrever esse nome no ficheiro HELP.DAT.

Sempre que não seja especificado o nome do ficheiro de dados na linha de comandos o programa vai ler o nome que está guardado no ficheiro HELP.DAT, e será esse o nome do ficheiro de suporte utilizado.

OS PROGRAMAS TSR

Como de certo já reparou, o programa aqui descrito é um TSR, isto é, fica residente em memória depois de terminada a sua execução. Isto permite que mais tarde a parte residente do programa possa ser activada (mesmo durante o decorrer da execução de outros programas).

O procedimento descrito atrás é talvez, para muitos, um verdadeiro mistério. Sendo a causa de todo este mistério, entre outras razões, a falta de documentação sobre este tipo de programa. Há quem aponte a Microsoft como responsável por esta situação, pois desde as primeiras versões do MS-DOS que esta companhia optou por não documentar certas funções do sistema operativo.

Em alguns casos, de vital importancia para um TSR que pretendesse coexistir com o MS-DOS sem problemas. No entanto, com o decorrer do tempo, houve quem se empenhasse em «caçar» essas funções. O que teve como consequência o aparecimento de vários programas deste tipo. Sendo sem dúvida o SideKick da Borland, o mais famoso.

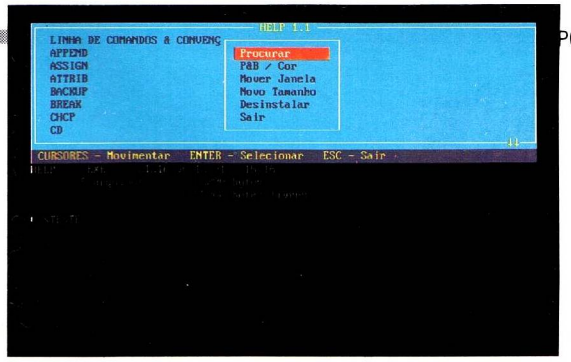
A estrutura básica deste tipo de programa é um pouco diferente dos programas normais. Podem ser divididos em duas partes principais: uma parte que é responsavel pela instalação do programa em memória e outra que contém o código necessário à utilização do programa quando residente em memória. As funções desempenhadas pelo bloco de instalação são básicamente estas:

- Verificar se o programa já está em memória,
- Inicializar algumas variáveis,
- Guardar os vectores das *interrupt* que irão ser capturadas,
- Redireccionar os vectores de *interrupt* usados, para rotinas de *interrupt* definidas no código do TSR (de notar que pelo menos uma destas rotinas deverá ser capaz de activar o programa),
- E finalmente sair para o MS-DOS deixando o programa residente em memória (função 31h da *interrupt* 21h).

Quanto à forma de activação depende muito das necessidades e da imaginação do programador. No processo de activação devem ser tomados vários cuidados, ou a activação poderá resultar num *crash* do sistema. Normalmente a activação de um programa residente em memória é feita através de rotinas de *interrupt*, sendo as seguintes as mais utilizadas:

- **Int 08h** Relógio do sistema (é executada 18,2 vezes por segundo, isto é, a cada 55 mili-segundos. Esta *interrupt* é ideal para activar um programa que tenha de fazer um determinado processamento de fundo regulado por tempo).
- **Int 09h** Teclado (é executada sempre que for pressionada uma tecla. É sem dúvida a mais utilizada, pois actualmente grande parte dos programas residentes em memória são activa dos através de uma combinação de teclas).

No caso de o leitor estar interessado neste tipo de programação, poderá consultar as fontes do programa HELP e, se desejar, poderá consultar os seguintes livros:



Born to Code in C, (orientado para a linguagem C)
Turbo Pascal: Advanced Programmer's Guide,
(orientado para o Pascal)
Editora McGraw-Hill.

(Qualquer destes livros dedica um capítulo aos programas TSR)

AS FONTES DO PROGRAMA HELP

Nota da Redacção: *por absoluta falta de espaço na disqueta, não nos foi possível incluir as fontes aqui referidas. Faremos tudo para as publicar no próximo número.*

A linguagem que utilizei na elaboração deste programa foi o Turbo C 2.0 da Borland International, Inc. Para facilitar a compilação, utilizei um projecto (ver ficheiro HELP.PRJ). Para quem não sabe, um projecto é uma ferramenta muito útil implementada pela Borland nas suas versões da linguagem C. Esta poderosa ferramenta permite ao programador ter vários arquivos fonte que depois de devidamente compilados e linquetados geram o programa. No caso deste programa os vários arquivos fonte são:

help.c	Bloco principal do programa. Neste bloco está a função pincipal que contém a inicialização deste programa.
tsr.c	Rotinas de activação da parte residente do programa.
kernel.c	Bloco central da parte residente.
videotsr.c	Rotinas usadas para manipular o vídeo (escrita de strings, manipulação do cursor, janelas, ...).
utils.c	Várias rotinas de uso geral para o programa.
keys.c	Rotinas de manipulação do teclado.
display.c	Rotinas usadas no menu de palavras chave.

Todos estes ficheiros incluem (#include....) no seu código dois arquivos:

help.mac	Macros defínidas para uso pelo programa.
help.pro	Protótipos de todas funções usadas pelo projecto.

Caso pretenda alterar as fontes para testes, deverá ter em atenção o ficheiro TCCONFIG.TC (também distribuído) que contém as opções usadas para criar o executável (HELP.EXE). Estas opções poderão ser alteradas, embora tomando todo cuidado com as opções do compilador e do linker, pois trata-se de um tipo de programação de «alto risco».

Espero que as fontes enviadas com o programa sirvam para afastar o mistério que parece rodear os programas residentes em memória. Espero também que o programa aqui descrito lhe seja útil.

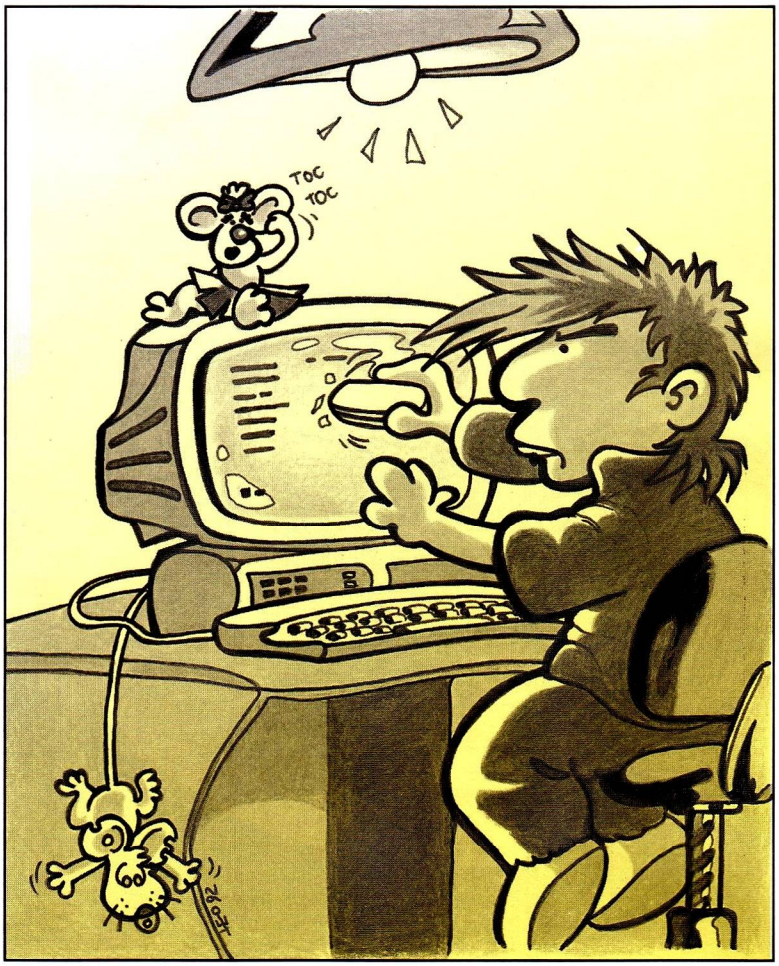
Carlos Ladeira



COLABORAÇÃO DO LEITOR

EXCLUI

Este pequeno utilitário que apresentamos aos nossos leitores foi objecto de algumas alterações, por nós sugeridas, após os ensaios efectuados sobre a primeira versão. Logo de início mostrou-se capaz de executar a função para a qual foi (bem) concebido. E foi útil. E interessante pois, contrariamente ao que é habitual, trabalha por «exclusão». O autor explica-nos como funciona e os cuidados a ter na sua utilização.



EXCLUI

Bolas... Onde é que está o PCTOOLS?. Quero apagar todos os ficheiros deste directório excepto os executáveis. Ou apago um a um ou vou à disqueta da **Spooler** e utilizo o...

EXCLUI 1.2

Esta situação é-vos familiar? A mim ocorre-me frequentemente. Para resolver isto, desenvolvi um pequeno utilitário em C para executar um determinado comando num directório e evitar que alguns ficheiros «sofram» o dito comando. Depois de alguns dias de trabalho surgiu o EXCLUI.

Sintaxe:

EX [-S] <Especificação de ficheiros a Excluir> <Comando a executar>

Explicação dos parâmetros

A especificação dos ficheiros pode ser qualquer uma aceite pelo DOS:

(*.com, *.dat, teste?.??a, etc.)

O comando será qualquer comando interno ou externo do MS-DOS, ou qualquer outro, possuindo os seus parâmetros usuais:

(del, format, pkzip, sst, etc.)

O parâmetro **-S** irá fazer com que o EX não apresente nada no ecrã a não ser os resultados produzidos pelo comando dado pelo utilizador (Modo Silencioso).

Se correr o programa sem parâmetros, ele apresentar-lhe-á uma breve explicação do modo o de utilizar.

Exemplos:

Imagine um directório com os seguintes ficheiros:

- **COMMAND.COM**
- **TESTE.BAT**
- **LIXO.TXT**
- **ABC.CDE**
- **TESTE.COM**

Exercício: Para apagar todos os ficheiros excepto o COMMAND.COM, digite :

EX command.com del *.*

O EX irá funcionar de um modo transparente para com o comando DEL. Como o leitor provavelmente sabe, ao mandar o MS-DOS apagar todos os ficheiros de um determinado directório, ele irá perguntar se tem a certeza. Dentro do EX, o DEL irá funcionar exactamente como o previsto. Depois do comando, volte a ver o directório com o DIR e verificará que todos os ficheiros, excepto o COMMAND.COM, desapareceram.

Outro exemplo mais complexo: (Se o leitor não compreender não se preoucupe):

EX *.com FOR %a in (*.*) do TYPE %a

Se o comando FOR fosse executado na linha de comando do MS-DOS, ele mostraria todos os ficheiros existentes no directório, incluindo os .COM que não são ASCII. Executando este comando, o MS-DOS apresentar-lhe-á sequencialmente os ficheiros TESTE.BAT LIXO.TXT e ABC.CDE

EXPLICAÇÃO TÉCNICA

Para utilizar o EX em ficheiros BATCH, ele devolve vários códigos de erro conforme o que se passou. Quando

os parâmetros estão errados, o programa devolve o valor 1 (ERRORLEVEL). Quando os ficheiros que especificou não são encontrados, ele devolove 2 e quando o segundo passo falha (ver a seguir), ele devolve 3. Se tudo se processar correctamente, o programa devolve 0.

MODO DE FUNCIONAMENTO INTERNO

Para funcionar, o EX esconde os ficheiros especificados na linha de comandó, executa o comando especificado e volta a pôr os ficheiros como os encontrou. O MS-DOS fornece, para mudar os atributos dos ficheiros, um comando: O ATTRIB. Mas, infelizmente, esse comando só modifica os atributos do arquivo e de «só-de-leitura». (MS-DOS versões anteriores à 5.00). Assim, recorre-se à função **0x43** (43 hexadecimal) do **INT 0X21** (*Interrupt* 21 Hexadecimal – Funções internas do DOS), que processa os atributos de um ficheiro. Esta funcão subdivide-se em duas subfunções:

A subfunção 1 altera o atributo. Assim, recolhemos os atributos originais com a subfunção 0 e combinamo-los com o valor que significa «escondido» a nivel de *bits*, utilizando a operação lógica **OR**. Alteram-se os atributos do ficheiro para este novo valor e processa-se o comando especificado. Depois inverte-se o processo, desligando o *bit* que corresponde ao atributo escondido.

Informação adicional

– O erro do 2º passo

O EX funciona em dois passos:

1 – Esconde todos os ficheiros e executa o comando especificado

2 – Recupera os atributos originais do ficheiro.

Se, por qualquer motivo, os ficheiros especificados não puderem regressar ao seu atributo original, verifica-se este erro. Causas prováveis deste erro :

– O comando especificado reconhece a existência de ficheiros escondidos.
– Houve um erro de dados (*Data error Reading*) ou qualquer outro que corrompeu a estrutura lógica dos dados.

CONCLUSÃO

O EX utiliza um algoritmo de separação do directório da especificação fornecida pelo utilizador um pouco complexa. Diga-se apenas que o algoritomo adapta-se a operações fora do directório e *drives* correntes.

Espero sinceramente que este utilitário vos facilite a já difícil tarefa de manusear um sistema informático. Este programa foi desenvolvido em C e foi concebido para ser o mais rápido e fiável possivel. Assim, eu não me posso responsabilizar por qualquer perca de dados que possa advir da má utilização do meu programa. Aconselho-vos sempre a fazerem *backups* dos vossos ficheiros mais importantes. Não estou a dizer que o meu programa irá de certeza causar perca de dados, mas, até o simples comando COPY do DOS pode causar uma situação desagradável.



Fábio Luis Bravo Teixeira da Silva

Executamos todo o tipo de trabalhos gráficos,
desde a concepção à impressão e acabamento.

Utilizamos avançadas técnicas de composição
e paginação computadorizadas aliadas à experiência
e conhecimento dos processos tradicionais.

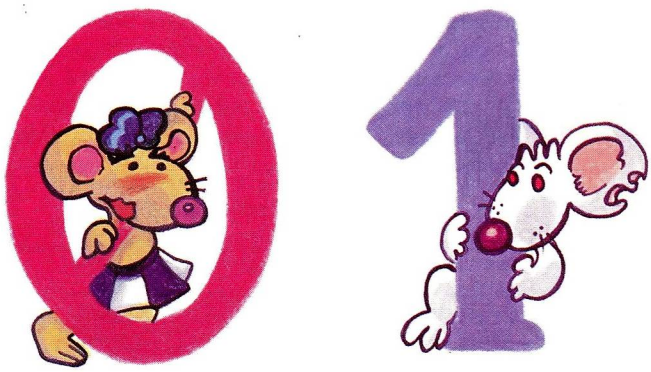
Oferecemos criatividade, competência e alta qualidade.
Cumprimos prazos.

Contacte-nos.

Graficria
Publicidade e Artes Gráficas, Lda.

Rua Duarte Pacheco Pereira, 69-A – Damaia-de-Baixo
2700 AMADORA – Telefs.: 474 60 11/474 60 41 – Fax: 475 22 89

A Construção de Módulos em Formato Binário



> **... estes módulos são uma imagem de código binário directamente executada pelo processador do computador.**

Este artigo ilustrará como produzir módulos **.BIN** para incorporar nos seus programas dBASE III PLUS e dBASE IV.

Como a extensão .BIN indica, estes módulos são uma imagem de código binário directamente executada pelo processador do computador.

Tanto o dBASE III PLUS como o dBASE IV vêm acompanhados de alguns .BIN, mas nada nos impede de fazermos os nossos próprios, pois é relativamente fácil. O esquema de utilização de um módulo .BIN de dentro do dBASE é o seguinte:

1) Carrega-se o módulo com:

LOAD **binfile**

Sendo **binfile** o nome de um módulo .BIN. O dBASE assume a extensão .BIN pelo que não é necessário mencioná-la, excepto no caso de módulos binários com qualquer outra extensão que não .BIN em que se deverá escrever o nome completo. Por exemplo, se tiver o módulo **binfile.sys** deverá escrever:

LOAD **binfile.sys**

2) Executa-se o módulo na altura apropriada com:

CALL **binfile**
ou
CALL **binfile** WITH parâmetros

No caso de haver parâmetros, o utilizador deverá ter o cuidado de respeitar a sintaxe prevista para o módulo em questão pois, de um modo geral, os módulos não se preocupam excessivamente em fazer o controlo e expurgo de erros.

3) Quando o módulo deixar de ser necessário podemos libertá-lo da memória com:

RELEASE **module binfile**

Os módulos BIN são normalmente feitos em *Assembly* embora possam também, mas dentro de certos limites, sê-lo na linguagem 'C'.

Vamos examinar como se produzem em *Assembly* ficando para um próximo número a alternativa da linguagem 'C':

1 – As regras são:

1.1 – O código inicia sempre no *offset 0*, e o Assemblador deverá ficar ciente desse facto pela inclusão da directiva:

ORG 0

no ínicio do programa fonte.

1.2 – A rotina é considerada como **FAR** pelo dBASE, por conseguinte deverá ser elaborada dando-se directiva ao Assemblador que se trata de *procedure far* ou então em alternativa colocar a instrução **RETF** no fim.

1.3 – No final da rotina é necessário repor a pilha como estava no íncio, e o valor do registo de segmento DS terá de ser o mesmo da entrada no módulo.

1.4 – As variáveis, são passados pelo dBASE em **DS:BX.**

E é só, muito simples, como se vê.

Vamos a um pequeno exemplo: o módulo APAGA.BIN que proporciona uma alternativa ao comando CLEAR do dBASE:

A sua sintaxe é: **CALL APAGA WITH CHR(n)**

Sendo **n** o número da cor e está compreendido entre 0 e 255:

Nº da cor = **FOREGROUND + (BACKGROUND x 16) + (BRIGHT x 8) + (BLINK x 128)**

Para **'FOREGROUND' e 'BACKGROUND':**
0=preto, **1**=azul, **2**=verde, **3**=cyan, **4**=vermelho, **5**=magenta **6**=castanho e **7**=branco.
BRIGHT e BLINK pode ser **0** ou **1**.

O **APAGA.BIN** está disponível na disquete **SPOOLER.** Se quiser obtê-lo a partir do código fonte faça:

Com o **MASM:**
MASM APAGA.ASM,,,;
LINK APAGA,,,;
EXE2BIN APAGA

Com o **TASM:**
TASM APAGA
TLINK APAGA
EXE2BIN APAGA

Com o **A86** é só:
A86 APAGA.ASM

APAGA.ASM

```
CODE SEGMENT
ASSUME CS:CODE, DS:CODE
ORG 0

START   proc far

        push bp         ; Estas instrução são típicas dos
        mov bp,sp       ; dos módulos que acompanham
                        ; o dBASE.
                        ; Os valores passados pelo
                        ; dBASE vêm apontados por
        push ds         ; DS:BX e os módulos
        mov al,[bx]     ; não devolvem valores na pilha.
                        ; Contudo no Clipper as duas
                        ; primeiras instruções são
                        ; obrigatórias, por exemplo:
                        ; push bp
                        ; mov bp,sp
                        ; push ds
                        ; lds ax,dword ptr [bp+6]
        push cs
        pop ds          ; Para utilização local vamos
                        ; apontar o segmento de dados
                        ; para o nosso módulo.
                        ; É mais facil trabalhar assim.
                        ; Repare que acima ASSUMimos
                        ; para o assemblador que o
                        ; segmento de código seria igual
                        ; ao segmento de dados, de
                        ; modo que ele tratou de todos os
                        ; detalhes.

        mov COR,al      ; Guardamos o valor da cor no
                        ; endereço do módulo já previsto
                        ; para o efeito.

        call QUAL_VIDEO; Verificar se o sistema é mono ou
                        ; a cores.
                        ; Isto é necessário pois a
                        ; localização da memória video
                        ; em modos texto não é igual em
                        ; mono e cores.

        mov ax,word ptr VID_SEG
        mov es,ax

        call APAGADOR   ; Temos tudo o que necessitamos,
                        ; e vamos então ao trabalho !
        pop ds          ; Limpeza final: Repor a pilha tal
                        ; como estava.
        pop bp          ; Note que como o procedimento
        ret             ; é FAR o assemblador substituirá
                        ; o ret por retf.
START   endp


COR     db (?)
VID_SEG dw 0b800h

QUAL_VIDEO:
        mov ah,0fh
        int 10h
        cmp al,7
      jnz VIDEO_END     ; Se for mono AL chega-nos com
                        ; o valor 7, e temos de substituir o
      mov ax,0b000h     ; valor 'default' que havíamos
        mov VID_SEG,ax  ; considerado.
VIDEO_END:

        ret

CONTADOR db 0

TABELA  db 127,113,109,107,103,101,97,89,83,79,73,
        71,67,61,59,53
        db 47, 43, 41, 37, 31, 29,23,19,17,13,11, 7, 5,
        3, 2, 1

APAGADOR:
        xor ax,ax
        xor bx,bx
        lea bx,TABELA   ; BX fica com o endeço de
                        ; deslocamento da tabela de
                        ; valores acima.
        mov al,contador
        xlat            ; A instrução XLAT carrega para
                        ; AL, o valor de uma tabela
                        ; endereçada por BX cujo índice
                        ; está inicialmente em AL.
        add al,almov bl,al
        dec al
        dec al
        inc CONTADOR
        cmp CONTADOR,20h
                        ; Quando ultrapassar 20h (32 em
                        ; decimal) chegamos ao fim da
                        ; tabela.
        jbe APAGA_PARTE
        mov CONTADOR,0
                        ; Vamos deixar a 0 o CONTADOR
                        ; pois o módulo poderá voltar a
                        ; ser chamado, e os
                        ; valores nele contidos são
                        ; estáticos.
        ret

APAGA_PARTE:
        mov di,ax
        push ax
        mov ah,COR
        mov al,20h
        stosw
        pop ax
        mov cx,300h
DELAY:
        loop DELAY      ; Os computadores são
                        ; excessivamente rápidos em
                        ; algumas situações...
        add ax,bx
        cmp ah,10h      ; Quando AH for 10h,
                        ; ultrapassámos a página activa
                        ; de vídeo e é tempo de parar.
        loopnz APAGA_PARTE
                        ; Trata-se de um loop controlado
                        ; apenas pelo sinalizador 'Zero
                        ; Flag', pois o registo CX é
                        ; sempre não nulo.
        loop APAGADOR   ; Este loop é infinito.

CODE ENDS
END START
```

COLABORAÇÃO DO LEITOR

COMPRIME



Tal como os homens, também os programas não se medem «aos palmos», nem são as «origens» (leia-se «linguagens») que os classificam. Esta filosofia de La Palisse destina-se a introduzir um pequeno conjunto de *batch files* construídos com o propósito de auxiliar os utilizadores de uma das grandes facilidades dos PC's: a compressão e descompressão de ficheiros. Claro que é indispensável possuir e ter instalados dois dos compressores mais divulgados entre nós. E tudo se torna mais fácil com o COMPRIME. Vejam só:

O programa COMPRIME serve para tornar mais simples a compressão de ficheiros. Ele permite a utilização do PKZIP, PKUNZIP, ZIP2EXE e do LHARC através de menus.

É constituido pelos seguintes ficheiros:

COM.BAT	ficheiro principal
MENU.TXT	ecrã do menu de opção
COMPRIME.TXT	ecrã do menu de compressão
ARRANQUE.TXT	ecrã de introdução
TITULO.TXT	ecrã para o título
ESCOLHA.COM	ficheiro que torna possivel a escolha
EXEMPLO.BAT	ensina a funcionar com o programa ESCOLHA

INSTALAÇÃO

1º Criar a directoria COMPRIME na drive C: (no caso de um computador sem disco duro será necessário alterar os caminhos no ficheiro COM.BAT de acordo com a *drive* e a directoria pretendida).

2º Copiar para esta directoria o ficheiro COMPRIME.EXE e corrê-lo para descomprimir este utilitário. Após a descompressão, apagar o COMPRIME.EXE para libertar espaço no disco. É conveniente guardar em disquete o ficheiro comprimido.

3º Colocar no PATH do AUTOEXEC.BAT a *drive* e directoria onde instalou o COMPRIME, o LHARC e o PKZIP, para permitir utilizar estes ficheiros a partir de qualquer directoria ou *drive*.

UTILIZAÇÃO

Para iniciar o utilitário proceder do seguinte modo:

1º Escrever COM <nome do ficheiro>, se estiver situado na directoria a comprimir.

Exemplo: **COM spooler**

2º Caso não esteja situado na directoria desejada, proceder da seguinte forma:

COM <nome do ficheiro> [drive] [directoria]

Exemplo: **COM spooler a: revista** (vai comprimir os ficheiros na directoria **revista** existente na *drive* **A**.

Se fizer **COM spooler a:** o utilitário vai comprimir os ficheiros existentes na ROOT da *drive* **A**.

3º O utilitário apresentar-lhe-á um ecrã de introdução e a seguir um menu de escolha do compressor a utilizar.

4º A partir daqui o utilitário apresenta-lhe um Menu com as seguintes opções:

1 – Comprimir	vai comprimir a directoria escolhida *
2 – Descomprimir	vai descomprimir o ficheiro com o nome dado no início
3 – Auto-descomp	vai tornar o ficheiro comprimido num ficheiro executável
4 – L. ficheiros	vai listar os ficheiros existentes no ficheiro comprimido
5 – Ajuda	vai mostrar um ecrã de ajuda

* No caso de ter escolhido o compressor PKZIP o utilitário vai mostrar o seguinte menu:

1 – Usar compressão rápida
2 – Usar compressão máxima
3 – Comprimir subdirectorias
4 – Ajuda

Nota: No caso de na iniciação do Utilitário escrever apenas COM sem dar o nome do ficheiro desejado, aparece um ecrã de Ajuda.

Um abraço para todos os que tornam possível a nossa revista.

Pedro Miguel Alves dos Santos

TESTAR O «RATO» EM PASCAL

Olá leitores.

Eis que o Pascal regressa à carga!

Desta feita, elaborámos um pequeno programa de teste que interessará sobremaneira a quem pretenda aceder ao rato através de um programa.

Das dezenas de funções disponibilizadas pelo interface MicroSoft, usámos apenas as mais importantes: inicialização, tornar visível o cursor, leitura da posição do cursor e leitura do estado dos botões. Como terão oportunidade de verificar se inspeccionarem a listagem do programa RATO, estas poucas primitivas são, por si só, suficientes para se trabalhar com o rato num programa que não tenha grandes exigências em termos de eficiência.

Para usar este nosso RATO.EXE – e qualquer programa que trabalhe com o rato – é necessário instalar previamente o respectivo *device-driver*. O *driver* tanto pode ser instalado no CONFIG.SYS (se for do tipo NNNNNNNN.SYS) como colocado residente (se for NNNNNNNN.COM ou NNNNNNNN.EXE). O acesso ao *driver* faz-se por intermédio do *interrupt* 33H.

Uma chamada aos serviços do *driver* do rato faz-se colocando o código do serviço no registo AX e chamando o *interrupt* de *software* 33H. Ao terminar o *interrupt*, o registo AX indicará o sucesso ou insucesso da operação; no caso de sucesso, alguns outros registos conterão os resultados da operação efectuada com a chamada ao *driver*.

Este programa de teste foi implementado em Turbo Pascal. Em Turbo Pascal, as chamadas aos *interrupts* de *software* fazem-se por intermédio do procedimento INTR, que se encontra disponível na UNIT DOS:

```
procedure intr(numero: byte; var registos : registers);
```

O parâmetro NUMERO determina qual o *interrupt* a ser efectuado – no nosso caso vai ser efectuado o *interrupt* $33 (o prefixo $ indica que o valor está expresso em notação hexadecimal).

O parâmetro REGISTOS, passado por variável, vai conter a configuração de valores nos registos do CPU no momento da chamada ao *interrupt*. Este parâmetro é do tipo REGISTERS, declarado igualmente na UNIT DOS:

```
TYPE
   REGISTERS=RECORD
      CASE  INTEGER OF
         0:  (AX,BX,CX,DX,BP,SI,DI,DS,ES,FLAGS: WORD);
         1:  (AL,AH,BL,BH,CL,CH,DL,DH:BYTE);
       END;
```

A declaração REGISTERS define uma estrutura do tipo *record* variante: pode aceder-se aos pares de registos (registos a 16 bits) ou aos registos simples (registos a 8 bits). A utilização de um ou outro ramo do CASE é perfeitamente transparente ao programador:

```
VAR A,B:REGISTERS;
 BEGIN
      A.AX:=$AAFF;
      B.AL:=$FF;
      B.AH:=$AA;
      WRITELN(A.AX);        (* imprime 43755 *)
      WRITELN(B.AX);        (* imprime 43755 *)
      WRITELN(B.AH);        (* imprime 170   *)
      WRITELN(B.AL);        (* imprime 255   *)
```

Neste exemplo, coloca-se no par de registos AX o valor hexadecimal AAFF de duas formas diferentes mas com resultados finais iguais.

No nosso programa do RATO, todas as chamadas ao INT 33H foram efectuadas através de procedimentos com a seguinte estrutura:

```
procedure initMOUSE;    (* SERVIÇO 00H - Inicializa o RATO *)
var r:registers;
   begin
      r.ax:=$00;          { Coloca o código do serviço no par AX }
      Intr($33,r);        { Invoca o MOUSE DRIVER  }
      if r.ax=$FFFF then{ O resultado da chamada vem em AX}
         begin
            { Acções no caso de tudo correr bem ...}
         end
      else
         begin
            { Acções no caso de tudo correr mal ... }
         end;
   end;
```

Na sua essência, um programa que use o rato é muito simples, como se pode avaliar pelo algoritmo apresentado:

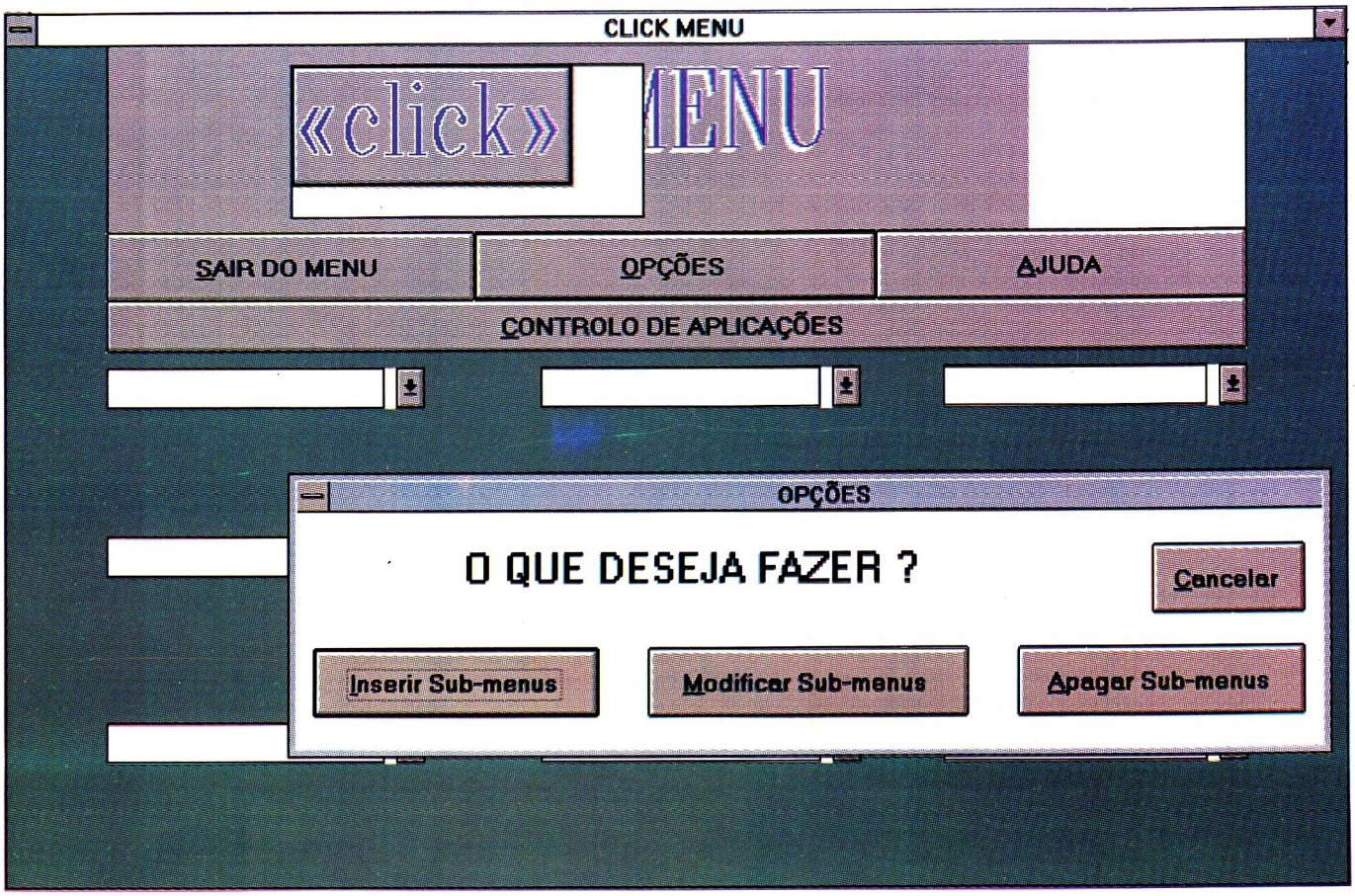
- inicializa o *driver* do rato;
- se tudo correr bem,
- torna o cursor visível;
- recolhe as informações da posição e estado dos botões;
- entra em ciclo de repetição, controlado pelo estado de uma variável – quando for premido ESC ou o botão esquerdo do rato dentro da área indicada como FIM, o ciclo termina;
- dentro do ciclo, recolhe as informações da posição do rato e estado dos botões; se alterar o estado anterior, refresca a informação no ecrã;
- senão (se tudo correr mal), dá mensagem de erro e termina.

Este programa (fonte e executável) encontra-se na disqueta – estão intimados a utilizá-lo para teste de todas as funções disponíveis no *driver*. A listagem das ditas funções encontrá-la-ão no artigo do meu ilustre colega Carlos Azevedo, «História e programação dos roedores informáticos», que tiveram o prazer de acompanhar nos números 15, 16 e 17 da **Spooler**. Tenho dito.

Paulo Bernardo

COLABORAÇÃO DO LEITOR

MENÚ



Eis uma agradável suspresa para os nossos leitores – um programa escrito em VisualBasic para o Windows 3.0! O nosso Tomé Pessegueiro esmerou-se e oferece-nos um interessante utilitário destinado aos «fãs» do Windows 3.0 – um Menu muito *user friendly* para um rápido e fácil arranque de qualquer aplicação. Mas passemos a palavra ao autor:

Existem vários sistemas operativos no mundo inteiro, mas o mais popular é, sem dúvida, o MS-DOS. Um programa que quase substitui esse sistema operativo, é o Windows, um ambiente de trabalho que tem a finalidade de tornar a vida dos utilizadores um pouco mais fácil.

Já na versão 3.0 e com a versão 3.1em versão experimental, foi só agora que ele foi aceite pela grande maioria dos utilizadores.

Pode-se perguntar porquê só agora e para quê um programa deste tipo?

O Windows só agora foi aceite mundialmente devido a várias causas:

– Esta versão é bastante fácil de instalar
– Trabalha com todas as linguas ocidentais
– As arestas foram limadas e o programa foi preparado para que qualquer pessoa pudesse trabalhar com ele (mesmo um principiante)
– Estão na moda as GUI's (Graphical User Interfaces), que são a base de trabalho dos Machintosh, por exemplo.
– A grande baixa de preços nos computadores permitiu a compra de máquinas com uma melhor *performance* e com capacidade de gestão de memória muito maior.

Mas, quanto a mim, o maior avanço desta versão foi o lançamento de ferramentas de trabalho para programadores e a maior facilidade em escrever programas específicos para este ambiente de trabalho.

Com efeito, foram lançados compiladores de C, Pascal, Prolog e até BASIC, especialmente para o ambiente Windows.

A partir de agora já é possivel escrever programas em quase qualquer linguagem e que aproveitem as capacidades deste programa.

O programa que acompanha este artigo, foi escrito em Visual Basic, um compilador de BASIC para o Windows e que permite à grande parte dos utilizadores de computadores, escreverem pequenas aplicações sem grande esforço nem conhecimentos; mas com uma apresentação e uma facilidade de uso não encontradas normalmente. É isso mesmo que caracteriza o Windows e os seus programas – a facilidade de uso, a beleza e a potência.

É provável que muitos digam que é muito incómodo e lento correr qualquer aplicação para o Windows. É claro que não é tão rápido como se fosse directamente no DOS, mas tem grandes vantagens, e só agora é que se está a começar a evoluir neste tipo de ambientes; quem sabe o que nos reserva o futuro?

Este artigo é também uma chamada de atenção a todos aqueles que pensam que o BASIC é uma linguagem para as crianças aprenderem a trabalhar com o computador, pois não é verdade.

O que é verdade é que a facilidade de uso e de aprendizagem desta linguagem não permite que seja tão rápida, nem que os programas compilados sejam tão pequenos como nas outras linguagens.

Mas.... para muitas aplicações isso é grandemente compensado pela facilidade e rapidez no desenvolvimento dessas aplicações.

Por isso mesmo apresento aqui o primeiro programa para Windows feito em língua portuguesa (penso eu!). Isto só foi possível devido à velocidade com que se podem fazer programas em BASIC.

Este compilador, embora dê grandes facilidades, tem algumas limitações:

– É bastante lento, pois o código não é totalmente compilado.

– Ocupa muito espaço, pois o ficheiro VBRUN100.DLL tem obrigatoriamente que acompanhar o executável produzido, o que é uma consequência da compilação em pseudo-código.

– Os programas produzidos só podem correr em máquinas com processador 80286 ou superiores, o que vai acontecer também com a versão 3.1 do Windows.

Os programas executáveis produzidos pelo Visual Basic não são propriamente executáveis, pois quando carrega o programa Click Menu, este vai chamar o .DLL que não é mais do que um interpretador «concentrado» de Visual Basic; por isso os programas ficam algo lentos.

Se possuir a versão 3.0 (ou superior) do Windows, pode executar este Menu. Para o instalar, basta seguir os seguintes passos: (use o botão esquerdo do seu rato para «premir» um botão no ecrã):

– Criar um directório chamado MENU, na raiz da *drive* C: (Não é obrigatório que o programa se situe nesta *drive* ou directório)

– Copiar para este directório o ficheiro compactado existente na disquete **Spooler**, e execute-o;

– Corra o Windows;

– Dentro do Program Manager, abra a janela onde quer que o ícon do programa se situe (ex: Main, Accessories);

– Vá ao menu FILE e escolha a opção NEW;

– Prima o botão OK;

– Prima o botão BROWSE;

– Na parte inferior da janela, verá duas caixas; uma contém os directórios e o outro, os ficheiros existentes. Entre no directório menu da *drive* C:, seleccione o ficheiro MENU.EXE e prima o botão OK;

– Prima o botão OK.

E pronto, já tem o programa instalado. Prima agora esse ícon duas vezes seguidas para o correr.

O ecrã principal do Click Menu, está dividido em duas secções: os botões, e os menus ou caixas. Na primeira secção aparecem três botões:

– Sair do Menu – se premir este botão, os dados são gravados no ficheiro MENU.DAT e o programa é finalizado.

– Opções– Permite-lhe inserir ou modificar um item de um menu.

– Ajuda – Mostra-lhe um pequeno quadro com indicações úteis.

– Controlo de Aplicações – Carrega em memória o gestor de aplicações do Windows que permite navegar entre as varias aplicações em memória.

A segunda Secção é composta por nove caixas com uma seta no lado direito de cada uma delas. Estas caixas são «armários» que podem conter vários items, que por sua vez são os nomes dos programas que quer correr.

Primeiro que tudo, deve dar um nome às caixas que quer utilizar. Por exemplo, temos uma caixa com o nome Windows (para as aplicações do Windows) e DOS (para as aplicações que correm directamente em MS-DOS). Para poder dar um nome a uma caixa, basta mover o seu rato de maneira a que o cursor (a seta que se move no ecrã), fique por cima de uma caixa. Nessa altura o cursor deve mudar de forma. Quando isso acontecer, prima o botão esquerdo do rato; depois é só digitar o nome que desejar.

Estas caixas têm a mesma função que os grupos do Program Manager (ex: Main, Accessories, Games), mas em forma de texto, o que permite uma melhor arrumação dos vários programas.

Experimente premir o botão «Opções» e logo a seguir o botão «Inserir submenus». Na caixa da esquerda verá os nomes das caixas; selccione uma delas e escreva o nome do programa que quer dentro dessa caixa.

Pressione «OK». Agora basta escolher o ficheiro principal, que costuma executar para correr esse programa; se quiser, pode também dar parâmetros necessários ao programa. Seleccione «OK». Pode inserir vários items em cada caixa, até um máximo de 40.

Você deverá estar de novo no ecrã principal do programa. Com o rato pressione a seta da caixa em que meteu o programa. Verá o seu programa pronto a correr, quer corra directamente no Windows, ou em DOS. Leve o cursor até ao item desejado e prima a tecla esquerda do rato para correr esse programa. No ecrã de ajuda pode encontrar designações como Menus, ou submenus que são a mesma coisa que caixas e items dessas caixas, respectivamente.

Se estiver no Menu e quiser trabalhar com uma aplicação que já esteja em memória, não a carregue de novo. Em vez disso, deve premir o botão que tem como título «CONTROLO DE APLICAÇÕES». Se assim o fizer, irá aparecer uma pequena janela com os vários programas já em memória.

Para poder trabalhar com um deles, basta seleccioná-lo com o botão esquerdo do rato e pressionar o botão que diz «SWITCH TO».

E pronto, a sua aplicação estará em realce, à frente de todas as outras janelas de outros programas.

Embora este programa seja uma espécie de Program Manager, penso que tem certas vantagens: a não utilização de ícons, o que permite uma melhor e mais compacta arrumação; maior facilidade na configuração; e mais importante que tudo, é em português.

Espero que gostem deste programa e que agora seja mais confortável abrir janelas, mas... cuidado com as correntes de ar. E, já agora, pensem Português!

Tomé Pessegueiro

“ ... apresento aqui o primeiro programa para Windows feito em língua portuguesa ... ”

PROGRAMAR EM C

(PARTE II)

INTRODUÇÃO

Nesta segunda lição do nosso curso de C, vamos analisar mais alguns aspectos desta interessante linguagem. Entre eles destacam-se:

- Constantes
- Ciclos de repetição (loops)
- Arrays
- Pointers
- Strings

CONSTANTES

Em C, constantes são valores fixos que não podem ser alterados pelo programa. Estas podem ser de qualquer um dos tipos básicos, no entanto a sua representação depende de tipo para tipo. Normalmente, não é necessário especificar o tipo de constante, pois o compilador ajusta-a para o tipo compatível mais pequeno. Por exemplo: 33 é um **int** por defeito, 45000 é um **unsigned** e 75000 será um **long**.

Normalmente, o tipo assumido pelo compilador é o adequado, mas se for necessário, o programador poderá forçar o compilador a usar o tipo pretendido, utilizando um sufixo. Este é colocado depois do número. Os sufixos de que dispomos são: **F** para o tipo **float**, **L** para o tipo **long** ou **long double** e **U** para **unsigned**. Seguem-se alguns exemplos:

TIPO	EXEMPLOS DE CONSTANTES
int	5 574 -78 27845
long int	49654L -894L 125365
short int	34 -56
unsigned int	5499U 50354
float	546.67F 2.98e-2F
double	234.43 15845983 -0.3456
long double	8756.12L

OUTRAS CONSTANTES

Existem ainda mais três tipos de constantes:

– Constantes hexadecimal e octal

Estas são de grande utilidade, pois frequentemente é necessário recorrer às bases hexadecimal (16) e octal (8). A definição de constantes numa destas bases é muito fácil. Para indicar que uma constante está no formato **hexadecimal** precede-se-a com **0x**, para indicar que o formato é **octal** precede-se-a com **0**. Seguem-se alguns exemplos:

```
int numero1 = 0x1B;    /* corresponde a 27 em decimal */
int numero2 = 0543;    /* corresponde a 355 em decimal */
```

– Constantes de string

Uma **string constante** é um conjunto de caracteres envolvidos por aspas. Por exemplo, "SPOOLER Magazine" é uma **string constante**. Convém no entanto salientar que, apesar do C permitir definir **constantes de string**, o tipo **string** não existe. Agora cuidado, pois uma **string constante** é definida com **aspas** e não com **plicas**. **'a'** é uma constante do tipo **char**, enquanto **«a»** é uma constante de **string**. As **string** serão estudadas de uma forma pormenorizada no final desta lição.

– Constantes de \ (back-slash)

Este tipo de constante é de grande utilidade, principalmente quando pretendemos enviar para o ecrã (por exemplo) caracteres de controle. Segue-se uma tabela com todas estas constantes:

Constante	Significado
\b	bakspace
\f	muda de linha e fica na mesma coluna
\r	retorno do cursor à primeira coluna
\n	nova linha (equivalente aos dois anteriores)
\t	tab horizontal
\"	aspas
\'	plica
\0	nulo
\\	o carácter \
\v	tab vertical
\a	sinal sonoro
\N	constante octal (N é a constante)
\xN	constante hexadecimal (N é a constante)

Segue-se um pequeno exemplo:

```
#include <stdio.h>
main(void)
{
 printf("\a\n\"Alerta!\x0F!\"\n");
 return 0;
 }
```

CICLOS DE REPETIÇÃO

Em C tal como em grande parte das linguagens de programação, os ciclos permitem que um dado grupo de instruções seja repetido até que uma determinada condição se estabeleça. Em C existem três tipos de ciclos:

– O ciclo for:

Este é dotado de uma flexibilidade muito grande. A sua forma geral é a seguinte:

```
for(inicialização; condição; incremento) instrucção;
```

ou

```
for(inicialização; condição; incremento){ instrucções }
```

Normalmente, a inicialização é uma afectação da variável de controle do ciclo. A condição é responsável pela repetição do ciclo, que é feita enquanto a condição for verdadeira. O incremento define a forma como a

variável de controle do ciclo é alterada de cada vêz que o ciclo é repetido. Todos o componentes do ciclo (inicialização, condição, incremento e instrução) são opcionais. Quando o ciclo chega ao fim (condição falsa), a execução passa para a primeira instrução depois do **for**. A sua «mecânica» é a seguinte:

(1) inicialização
(2) verifica a condição e sai do ciclo se ela for falsa
(3) instrução / instrucões
(4) incremento
(5) salta para 2

Segue-se um pequeno exemplo que imprime no ecrã os números de 0 a 50:

```
#include <stdio.h>
main(void)
{
   int i;   /* variavel de controle do ciclo */

   for (i=0; i<51; i++) printf("%d ", i);
   return 0;
}
```

– Variações possíveis do ciclo for

A discução anterior descreve a forma mais simples do ciclo **for**. No entanto são permitidas variações na sua forma, daí a sua flexibilidade e «potência». Uma das variações mais comum é o uso do operador «,». Exemplo:

```
for (i=0, j=100; i<j; i++, j—) faz_qualquer_coisa();
```

– O ciclo while

Este é talvez o segundo ciclo mais usado em C, depois do **for**. A sua forma geral é a seguinte:

```
while (condição) instrução;
```

ou

```
while (condição) { instruções }
```

Neste ciclo **instrução** é executada enquanto a **condição** for verdadeira. Quando a condição for falsa a execução passa para a primeira instrução depois do **while**. No próximo exemplo o ciclo é executado até a tecla **f** ser premida:

```
#include <stdio.h>
#include <conio.h>   /* conio contem a declaração de getch() */
main (void)
{
   char ch=0;

   printf("\nPressione a tecla <f> para sair do programa ...");
   while (ch!='f' && ch!='F') ch=getch();
   /* getch() lê uma tecla do Standard Input (teclado) */
   return 0;
}
```

No ciclo **while** tal como no ciclo **for** é possível o uso da vírgula para separar expressões, por exemplo:

```
#include <stdio.h>
#include <conio.h>   /* conio contem a declaração de getch() */
main (void)
{
   char ch;

   printf("\nPressione a tecla <f> para sair do programa ...");
   while (ch=getch(), ch!='f' && ch!='F');
   return 0;
}
```

– O ciclo do-while

Este ciclo é bastante idêntico ao **while**. Exepto num ponto: enquanto que o **while** faz o teste à condição antes de executar as instruções que lhe estão associadas, o **do-while** só faz o teste à condição depois de executadas as instruções que lhe estão associadas. Assim, o ciclo **do-while** poderá tirar vantagem sobre o **while** quando o bloco de instruções que lhe está associado tem de ser executado pelo menos uma vez. A sua forma geral é:

```
do {
instrucção ou instrucções
} while (condição);
```

Apesar das chavetas não serem necessárias quando o ciclo só tem uma instrução, convém usá-las sempre, pois torna o programa mais legível. Segue-se um exemplo:

```
#include <stdio.h>
#include <conio.h>   /* conio contem a declaração de getch() */
main (void)
{
   char ch=0;

   printf("\nPressione a tecla ESCAPE para sair do programa");
   do {
      printf("%c\n—> ", ch);
      ch=getch();
      } while(ch!=27);   /* 27 valor ASCII da tecla ESCAPE */
   return 0;
}
```

ARRAYS

Um **array** pode ser definido como sendo uma sequência de elementos de um determinado tipo, podendo estes ser acedidos através do mesmo nome. Cada elemento de um **array** é acedido através de um «índice», ficando todos seguidos na memória, sendo o primeiro colocado num endereço mais baixo que o último.

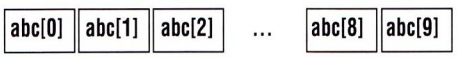
– Arrays de uma dimensão

A forma geral de declarar um **array** é:

```
tipo nome[tamanho];
```

tipo determina o tipo de cada um dos elementos do **array**, **tamanho** é o número de elementos do **array**. Por exemplo, para declarar um **array** de nome "abc" com 10 elementos do tipo **int**, proceder-se-ia da seguinte forma:

```
int abc[10];
```



O primeiro elemento do **array** é acedido através de **abc[0]**, para aceder ao último usa-se **abc[9]**. Em C a verificação de acesso dentro dos limites definidos na declaração do **array** não é feita, é-nos dada total liberdade. No entanto este tipo de programação pode ser perigosa, pois ao «mexermos» em zonas fora do **array**, vamos estar muito provavelmente a «mexer» noutras variáveis. Quanto à memória ocupada por um **array** de uma dimensão, pode ser calculada, multiplicando o número de elementos do **array** pelo número de bytes que ocupa o seu tipo base.

Segue-se um exemplo do uso de um **array**:

```
#include <stdio.h>
main (void)
{
      int abc[10];
      char i;

      for (i=0; i<10; i++) abc[i]=(i+1)*7;
      printf("\nTabuada dos 7\n");
      for (i=0; i<10; i++) printf("\n7 X %d = %d", i+1, abc[i]);

      return 0;
}
```

– Arrays de duas ou mais dimensões

Em C tal como em muitas outras linguagens, é possível definir **arrays** de duas ou mais dimensões. A sua forma geral é a seguinte:

tipo nome[a][b]...[n];

No entanto, **arrays** de três ou mais dimensões não são normalmente utilizados porque, para além de ocuparem muita memória, o seu uso não é muito solicitado. Por tudo isso, vamos restringir o nosso curso a **arrays** de uma e duas dimensões. Segue-se um exemplo da declaração de um **array** de duas dimensões:

char tabela[3][4];

tabela[0][0]	tabela[0][1]	tabela[0][2]	tabela[0][3]
tabela[1][0]	tabela[1][1]	tabela[1][2]	tabela[1][3]
tabela[2][0]	tabela[2][1]	tabela[2][2]	tabela[2][3]

E agora, um exemplo:

```
#include <stdio.h>
main(void)
{
   int tabela[10][10];
   char lin, col;

   for (lin=0; lin<10; lin++)      /* preenche o array */
      for (col=0; col<10; col++)
         tabela[lin][col]=col*lin;

   for (lin=0; lin<10; lin++)      /* mostra o conteúdo do array */
      for (col=0; col<10; col++)
         printf("\ntabela[%i][%i]=%i", lin, col, tabela[lin][col]);

   return 0;
}
```

– Arrays e a sua inicialização

Em C é possível inicializar um **array** no momento da sua declaração. A forma geral da inicialização é a seguinte:

tipo nome[tamanho 1]...[tamanho n] = { lista de valores };

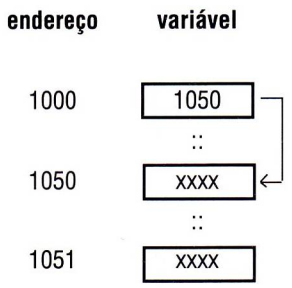
A lista de valores é composta por uma lista de constantes separadas por vírgulas. As constantes da lista tem de ser compatíveis com os tipos que vão inicializar, sendo a primeira da lista atribuída ao primeiro elemento do **array**, a segunda ao segundo elemento, e assim sucessivamente. Seguem-se exemplos:

```
char i[8]={56, 87, 34, 57, 32, 0, 12, 69};

int t[4][3]={
 45, -65,   0,          /* linha 0, colunas 0..2 */
  3,  2,   -1,          /* linha 1, colunas 0..2 */
 56, -89,  3243,        /* linha 2, colunas 0..2 */
 78, -23, -1567         /* linha 3, colunas 0..2 */
 };
```

POINTERS

Os **pointers** são talvez um dos aspectos mais importantes da linguagem C. Sem eles o C perderia muita da sua «potência». Por outro lado, o seu uso por programadores inexperientes pode causar muitas dores de cabeça, isto porque é suficiente usar um **pointer** não inicializado para gerar um *crash* do sistema (claro que nem sempre é este o resultado, poderá ainda ser pior). Os *bugs* causados por **pointers** são, geralmente, os mais difíceis de encontrar. Basicamente, um **pointer** é uma variável que contém um endereço de memória, normalmente este endereço é a localização de outra variável na memória. Se uma variável contém o endereço de outra, diz-se então que a primeira aponta para a segunda. A figura abaixo ilustra bem esta situação:



Se uma variável vai ser declarada como sendo um **pointer**, então é necessário especificar o tipo de variável que ela vai apontar. A declaração é idêntica à declaração de uma variável normal, com a diferença que antes do nome da variável (neste caso, do **pointer**) é colocado o carácter *. Vejamos um exemplo:

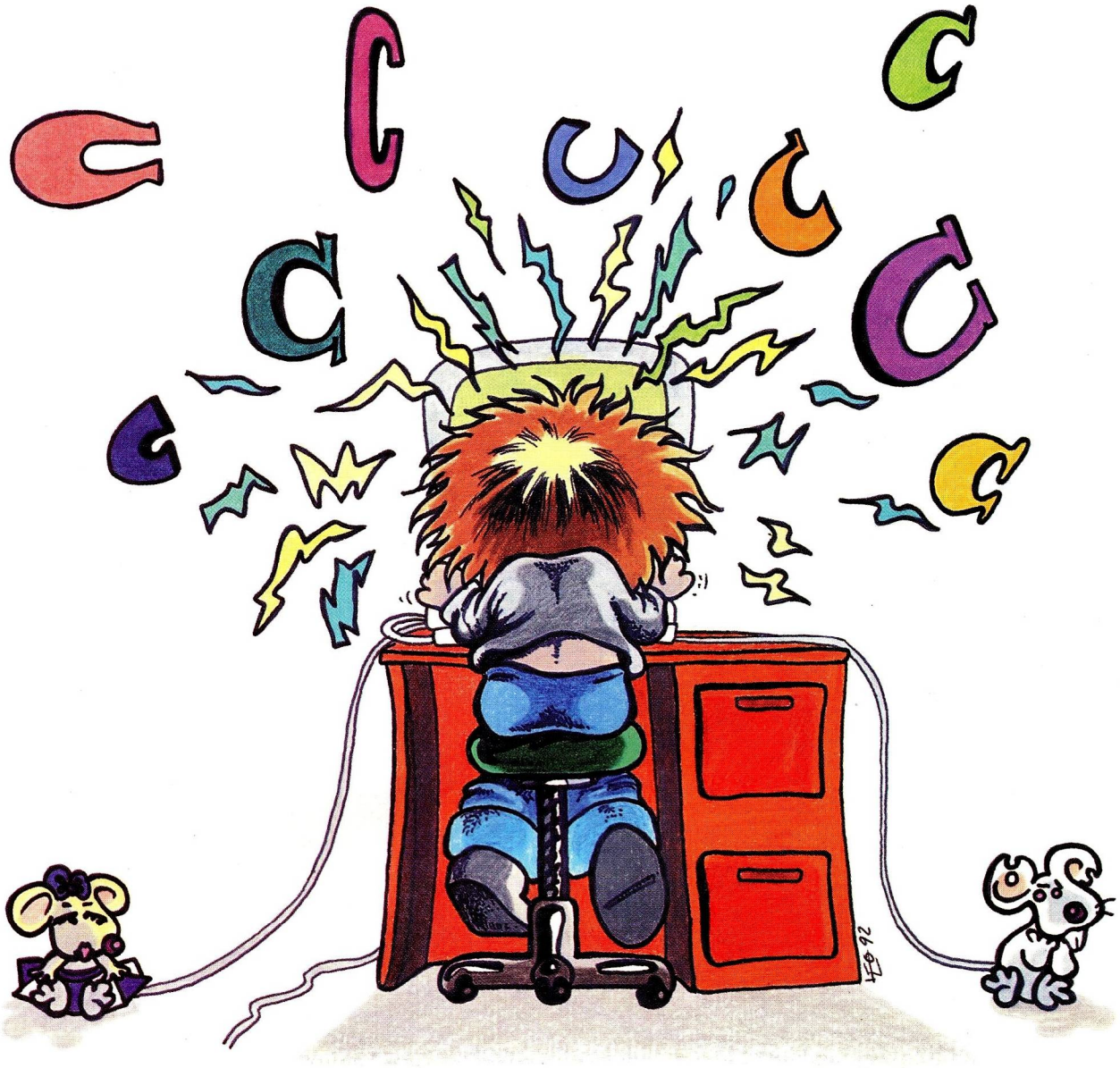
```
int *aponta_para_int;    /* tipo *nome; */
```

Na última lição foram mencionados os operadores de **pointer**, no entanto nada sobre eles foi dito. São apenas dois (* e **&**), o **&** é um operador unário e retorna o endereço de memória do seu operando. Por exemplo:

```
int *ptr, numero=100; /* é declarado um pointer para int e um int */
ptr=&numero;          /* faz-se ptr apontar para número */
```

Neste caso imagine que o endereço de memória da variável **numero** é 2000, na afectação seria esse o valor colocado em **ptr** e não 100 (valor inicial da variável numero). Quanto ao operador *, este é um complemento do operador **&**. É um operador unário e retorna o valor apontado pelo seu operador. Isto é, imagine o exemplo anterior: se executássemos a seguinte chamada à função printf: **printf("%i", *ptr)**, ser-nos-ia mostrado «100», pois **ptr** aponta para a variável **numero** que contem 100.

IMPORTANTE: Todos os **pointers** tem de ser inicializados antes de serem utilizados, caso contrário



corremos o risco de estar a mexer num local da memória desconhecido e, como é fácil compreender, isto não é nada bom.

Veja o seguinte exemplo:

```
main(void)
{
        int *p;

*p=100;  /* o valor 100 é colocado no local apontado por p, como p */
 .       /* aponta para um local desconhecido, o valor 100 poderá estar*/
 .       /* a ser colocado num local qualquer do programa, inclusive */
 .       /* por cima do código do próprio programa */

return 0;
}
```

– A afectação de pointers

A afectação de **pointers** é como qualquer variável. O próximo exemplo mostra como é:

```
#include <stdio.h>
main(void)
{
  int *p1, *p2, x;

  x=20;    /* afecta x com 20 */
  p1=&x;   /* p1 fica a apontar para x */
  p2=p1;   /* p2 fica a apontar para o local onde p1 aponta, isto é, para x */
  *p2=50;  /* afecta x com 50, pois p2 aponta para x */
  printf("%i", *p1);   /* envia para o ecrã o valor de x, p1 aponta para x */

  return 0;
}
```

– Aritmética de pointers

Tal como as variáveis normais, os **pointers** também se podem somar, subtrair, etc., embora com umas regras um pouco diferentes. Imagine que tem um **pointer p** que aponta para o tipo **int**, com o valor 2000. Assumindo também que um inteiro ocupa **2 bytes**. Depois da expressão:

```
p++;
```

O conteúdo de **p** será 2002, e não 2001. Isto porque no incremento de **pointers**, é-lhes sempre adicionado o número de bytes que ocupa o tipo para o qual eles apontam (determinado no momento da declaração). Assim, cada vez que um **pointer** é incrementado/decrementado,

ele vai apontar para a localização de memória que contém o próximo/anterior elemento do seu tipo base. O mesmo se aplica a operações do tipo:

```
p=p-5;
```

Neste caso **p** fica a apontar para o quinto elemento do tipo **int** antes do que ele apontava anteriormente (10 bytes atrás). Se o tipo base do **pointer** for **char** então tudo se passa como se fosse aritmética normal, pois um **char** só ocupa um byte.

– Comparação de pointers

É perfeitamente possível comparar dois **pointers** numa expressão relacional. Imagine que tem dois **pointers p1 e p2**, a próxima expressão é perfeitamente válida:

```
if (p1<p2) printf("\n p1 aponta para um endereço mais baixo que p2 ");
```

– Arrays e pointers

Existe uma relação muito próxima entre **arrays** e **pointers**. Considere o exemplo que se segue:

```
int st[100], *p;

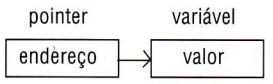
p=st;     /* p aponta para o primeiro elemento de st */
```

A linha **p=st** é equivalente a **p=&st[0]**. Imagine agora que pretende aceder ao sétimo elemento do **array**:

```
st[6]  ou  *(p+6)
```

– Indirecção múltipla

É ainda possível ter um **pointer** que aponta para outro **pointer** que por sua vez aponta para o valor desejado. Na figura abaixo poderemos ver como funciona:



Indirecção simples



Indirecção múltipla

A sintaxe da declaração de um **pointer** para **pointer** será:

```
tipo **nome;
```

Segue-se um exemplo:

```
#include <stdio.h>
main(void)
{
   int a, *p, **pp;

   a=69;    /* coloca o valor 69 em a */
   p=&a;    /* p aponta para a */
   pp=&p;   /* pp aponta para p que por sua vez aponta para a */
   printf("\n%d", **pp);    /* escreve no ecrã 69 */
   return 0;
}
```

STRINGS

Pode parecer estranho, mas em C não existir o tipo **string**, ou qualquer coisa que se assemelhe. Em C, uma **string** é composta da seguinte forma: uma sequência de caracteres seguidos do carácter '\0' (valor zero). Como o tipo **string** não está definido, são usados **arrays** do tipo **char** para guardar **strings**. O número de elementos que estes deverão ter será o número de caracteres máximo da **string** que vão comportar, mais um para o carácter '\0' (fim de **string**). Seguem-se alguns exemplos:

```
char st[9]={83, 80, 79, 79, 76, 69, 82, 46, 0};
char st[9]={'S', 'P', 'O', 'O', 'L', 'E', 'R', '.', '\0'};
char st[9]="SPOOLER.";
char st[]="SPOOLER.";
```

Todos os exemplos atrás vão inicializar o **array st** com «SPOOLER». No entanto, o último exemplo é sem duvida o que mais fácil torna a declaração, pois no acto da declaração, o **array** não é dimensionado: será então o compilador que, ao encontrar esta expressão, vai auto-dimensionar o **array** para o tamanho da **string** que está a inicializar. Este tipo de situação só é possível se o **array** for inicializado. Outro exemplo também muito utilizado é um **array** de ponteiros para **char**. Por exemplo:

```
char *mensagens[ ]={
    "olá",                  /* mensagens[0] */
    "bom dia",              /* mensagens[1] */
    "SPOOLER magazine",     /* mensagens[2] */
    "linguagem C",          /* mensagens[3] */
    "098gjkfjlbt80",        /* mensagens[4] */
};
```

PROGRAMA EXEMPLO (VERSÃO 2)

Poucas alterações foram feitas em relação a primeira versão deste programa, isto porque só será possível dar um certo avanço quando forem dadas as funções (na próxima lição). As alterações mais significativas foram feitas mais a nível da estrutura. Os labels/gotos foram todos substituídos por ciclos e foram usados pointers para uma maior simplificação. A fonte do programa exemplo poderá ser encontrada no ficheiro «CALC1.C».

NA PRÓXIMA LIÇÃO

Na próxima lição serão focados os seguintes tópicos:

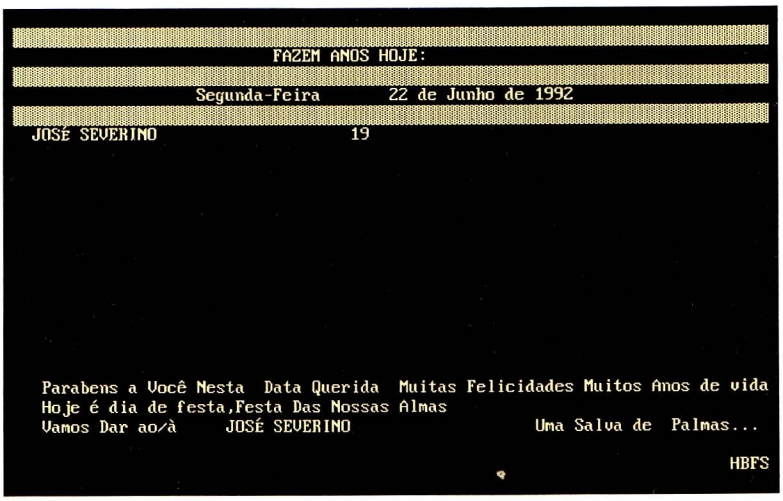
- Structs
- Unions
- Funções
- Recursividade
- As funções mais importantes da Standard Library
- Versão 3 da calculadora

Carlos Ladeira

COLABORAÇÃO DO LEITOR

GESTÃO DE ANIVERSÁRIOS

Decididamente, a imaginação do nosso fiel leitor Herberto Silva, da Madeira, não pára de nos presentear com novos e interessantes utilitários. Desta vez dedicou-se aos muito ocupados (ou distraídos) mas que não querem ficar mal vistos perante os seus parentes e amigos. Quem tiver este programa instalado no seu computador, dificilmente passará pela vergonha do esquecimento de um aniversário.



INSTALAÇÃO

Da mesma forma que os outros programas, penso que este também se apresentará numa forma compactada na disquete da **Spooler**. Depois de executar a descompactação para uma disquete ou mesmo para o disco, deverá criar um directório chamado «ANOS» e copiar para lá os ficheiros descompactados. Como alternativa, crie primeiro esse directório e faça a descompactação directamente para ele.

Após a instalação completada, caso pretenda uma execução do programa todas as vezes que ligar o computador, deve acrescentar no fim do ficheiro AUTOEXEC.BAT as seguintes linhas:

```
cd \anos
anos
cd \
```

INTRODUÇÃO

Este programa foi desenvolvido em Clipper e serve para detectar os aniversariantes na data corrente, pelo que após a sua execução, caso aconteça existir aniversariantes, o programa dirá quem faz anos e idade correspondente, tocando e escrevendo no ecrã a música e a letra do «Parabéns a Você».

Após a apresentação do ecrã principal o programa sugere que regresse ao DOS premindo a tecla **S** ou premindo qualquer tecla passar para o menu de gestão do programa.

TRATAMENTO DE DADOS

A base de dados possui a seguinte estrutura:

1	NOME	Character	25
2	DATA	Date	8

NOTA: Utiliza-se a tecla ESC para regressar à opção ou situação anterior. Para escolher determinada opção do menu, move-se com as teclas do cursor (cima,baixo) e prime-se ENTER após a escolha seleccionada.

INSERIR: Esta opção permite inserir novos aniversariantes, introduzindo o nome até 25 caracteres e a respectiva data de nascimento no tipo (dd-mm-aa) - dia, mês e ano.

ALTERAR: Permite alterar o nome ou a data de nascimento de determinado aniversariante, digitando o nome pretendido.

ELIMINAR: Apaga do ficheiro determinado aniversariante. Para isso, introduz-se o nome respectivo, e para apagar digita-se o número do aniversariante que se encontra a piscar, premindo ENTER de seguida, como confirmação.

CONSULTA: Permite uma consulta rápida de todos os aniversariantes listando o nome e a data de nascimento correspondente. Caso não apareçam todos os aniversariantes, liste os restantes premindo as teclas do cursor (baixo, cima). A listagem apresenta-se ordenada alfabéticamente.

FIM : Regresso ao DOS.

R: Reindexar o ficheiro – permite reindexar (ordenar novamente o ficheiro dos aniversariantes), no caso de acontecer algum erro na ordenação do próprio ficheiro.

Penso que com estas explicações não haverá problema em utilizar este pequeno programa. Felizes Aniversários!

Herberto B.F.Silva

Programar em Clipper (PARTE II)

Para organizar uma base de dados, são necessários vários procedimentos: inserir, alterar, anular e visualizar dados. São estas operações básicas que aqui vamos descrever, não esquecendo a criação da própria base de dados. Vamos ainda escrever um pequeno programa para ilustrar os comandos necessários a este controlo. O programa (uma agenda, para não variar!) apenas tem algumas opções muito simples, reservando as mais elaboradas para o próximo número (como por exemplo: impressão de etiquetas postais, listagem de aniversariantes, etc).

MANUTENÇÃO DE UMA BASE DE DADOS

Criar uma Base de Dados

Uma base de dados é constituida por campos e registos. Existem cinco tipos de campos em dBase:

TIPO DE DADOS		TAMANHO MÁXIMO
CHARACTER	(carácter)	254
NUMERIC	(numérico)	19
DATE	(data)	8
LOGICAL	(lógico)	1
MEMO	(memo)	65 500 (64 Kb)

O tipo CHARACTER, pode conter qualquer carácter (letras, números ou sinais). O tipo NUMERIC, pode ser submetido a cálculos aritméticos e conter números com ou sem casas decimais. O tipo DATE permite armazenar datas e o formato padrão é mm/dd/aa. Estas datas podem ser somadas e subtraídas entre si. O tipo LOGICAL contém apenas um carácter, representando verdadeiro (T ou Y) ou falso (F ou N). O tipo MEMO é utilizado para conter blocos de informação textuais. Estes blocos são armazenados num ficheiro auxiliar (de extensão DBT) e pode conter qualquer tipo de informação.

O comando CREATE define uma nova base de dados. Para utilizar este comando, deve indicar o nome do ficheiro (até oito caracteres), o nome dos campos (até dez caracteres), o tipo dos campos e o tamanho dos mesmos. O tamanho dos campos diz respeito à quantidade de informação que cada campo deverá conter (armazenar). Nos campos numéricos, o tamanho inclui o número de casas decimais e o ponto decimal. O número máximo de campos que o dBase permite definir, são 128.

EXEMPLO:
Para criar uma base de dados, chamada AGENDA, contendo informações relativas ao NOME, MORADA, TELEFONE e DATA DE NASCIMENTO, devemos fazer o seguinte na *prompt* do dBase:

CREATE agenda (ENTER)

No menu CREATE indicamos a estrutura da base de dados:

NOME	Character	40
MORADA	Character	40
C. POSTAL	Character	40
TELEFONE	Character	10
DATA_NASC	Date	8

Para finalizar a inserção de campos, utilize em simultâneo as teclas Ctrl-W.

Ao executar o comando DIR na *prompt*, aparecerá o ficheiro AGENDA.DBF, não contendo ainda qualquer registo.

A estrutura da base de dados pode ser alterada, executando o comando MODIFY STRUTURE, no entanto o seu uso deve ser cuidadoso pois é possível perder informação caso não se utilize correctamente.

Adicionar Registos

Para inserir registos na base de dados utiliza-se o comando APPEND, cuja sintaxe é:

```
APPEND [BLANK]
APPEND FROM  <ficheiro> [FOR / WHILE <condição>]
             [DELIMITED [WITH BLANK / <delimitador>]
```

Ao digitar apenas APPEND na *prompt*, o dBase insere um novo registo e permite-lhe preenchê-lo com informação. Utilizando a opção BLANK, apenas é inserido um registo em branco no fim da base de dados.

O comando APPEND FROM copia os registos de um ficheiro de dados (.DBF) para o final do ficheiro. Com DELIMITED, é possivel copiar dados de um ficheiro de texto para a base de dados em uso. Caso os dados tenham algum delimitador (p.ex.: " ou '), este deve indicar-se (por defeito, o dBase utiliza as aspas), caso contrário digitar a palavra BLANK.

Este comando permite ainda inserir registos, segundo uma determinada condição, como por exemplo:

```
APPEND FROM ficheiro.txt FOR nome="MARIA" DELIMITED WITH "
```

Neste caso, apenas os nomes que comecem por MARIA serão inseridos na base de dados em uso.

Alterar Registos

Os registos podem ser alterados usando o comando EDIT com a seguinte sintaxe:

```
EDIT [<registos>] [FIELDS <campos>] [FOR / WHILE <condição>]
```

A opção <registos> estabelece em que registo ou registos será aplicado o comando e pode ser um seguimento das frases:

RECCORD <n>	– o número de um registo simples;
NEXT <n>	– <n> registos começando no registo posicionado;
REST	– começando no registo posicionado e terminando no fim do ficheiro;
ALL	– todos os registos da base de dados.

Digitar apenas EDIT na *prompt* do dBase, permite alterar o registo corrente. A opção FIELDS, determina quais os campos que se querem alterar. Por exemplo, o seguinte comando altera os campos NOME e MORADA, para os nomes que comecem pela letra A, em toda a base de dados.

```
EDIT FIELDS nome,morada FOR nome="A"
```

Apagar Registos

Para apagar registos, existem duas etapas:

1 - Eliminação lógica (marcar o registo para ser removido da base de dados);
2 - Eliminação física (eliminar todos os registos marcados para remoção).

A primeira etapa, apenas marca o registo com um asterisco (*) e é utilizado o comando DELETE com a seguinte sintaxe:

```
DELETE [<registos>] [FOR / WHILE <condição>]
```

Para anular esta operação, executa-se o comando RECALL, que tem a mesma sintaxe que o comando DELETE.
A segunda etapa consiste em eliminar definitivamente os registos marcados para remoção, através do comando PACK. Depois de utilizado este comando, é impossível recuperar os registos que tenham sido apagados.
Por exemplo, se pretendermos eliminar todos os registos cujos nomes comecem por MARIA ou os telefones por 943, digitamos o seguinte:

```
DELETE FOR nome="MARIA" .OR. telefone="943"
```

Para recuperar os registos que foram marcados, mas apenas aqueles em que o telefone comece por 943:

```
RECALL FOR telefone="943"
```

Na sequência destes comandos, ao executar PACK, só os registos cujos nomes comecem por MARIA é que serão definitivamente apagados.
Para apagar todos os registos de uma base de dados, executa-se o comando ZAP. Este comando é equivalente ao comando DELETE ALL seguido do comando PACK, mas muito mais rápido.

Avançar/Recuar registos

Para passarmos de um registo para outro, utiliza-se o comando SKIP:

```
SKIP [<nº de registos>]
```

Este comando permite avançar ou recuar um determinado número de registos, conforme o número indicado como parâmetro seja positivo ou negativo. Executado sem qualquer número, avança um registo.
Quando se atinje o fim da base de dados (End Of File) ou o início (Begin Of File), as funções EOF( ) e BOF( ) devolvem o valor verdadeiro.

```
SKIP 6
SKIP -3
SKIP
```

Abrir uma Base de Dados

Para abrir uma base de dados já criada, utiliza-se o comando USE:

```
USE [<ficheiro.dbf>] [INDEX <ficheiro.ndx>][ALIAS <alias>]
```

Este comando abre um arquivo (de extensão DBF, caso não seja indicada outra) e opcionalmente o ficheiro de indíces (NDX), e ao mesmo tempo fecha a base de dados que estiver aberta na mesma área de trabalho. ALIAS é um nome que se dá à base de dados, para que a sua identificação seja mais simples (Ex.: USE xyz_382.l2i ALIAS fich).

No seguinte exemplo, vamos abrir a base de dados AGENDA, dar-lhe o nome de AG e usar o ficheiro de indíce F_NOME.NDX:

```
.USE agenda ALIAS ag INDEX f_nome
```

USE sem qualquer tipo de opções, fecha o ficheiro DBF em uso.

ORGANIZAÇÃO DE UMA BASE DE DADOS

Para mais rapidamente localizar os dados necessários à execução de uma determinada tarefa, é conveniente ter as bases de dados organizadas.

Existem dois processos de organização:

1 - Organização lógica (comando INDEX);
2 - Organização física (comando SORT).

Estas duas formas de organização são de utilização comum em programas desenvolvidos em dBase.
A organização lógica cria um ficheiro auxiliar (de extensão NDX) onde guarda apenas os dados referentes ao campo «indexado», mantendo-se os restantes campos na base de dados activa. Este tipo de indexação pode ser activado ou desactivado.
A organização física cria uma nova base de dados com um nome diferente e copia toda a informação do ficheiro antigo para o novo, mas organizado quer por ordem alfabética, numérica ou cronológica. Este tipo de organização não pode ser desactivado e exige mais tempo a ser realizada que a organização lógica, além de ocupar substancialmente mais espaço em disco. Com efeito, o comando INDEX apenas cria um ficheiro de índices, enquanto que o comando SORT irá criar um novo ficheiro contendo toda a informação da base de dados original, mas ordenada. A inserção de um novo registo irá exigir nova ordenação, enquanto o comando INDEX prescindirá desta operação se estiver activado.

Comando INDEX

Para indexar um determinado campo, executa-se o comando INDEX que tem a seguinte sintaxe:

```
INDEX ON <campos> TO <ficheiro de indice>
```

Por exemplo, para ordenarmos a nossa Agenda por ordem alfabética de nomes, executamos o seguinte comando:

“Tanto a ordenação lógica como a física, são usadas para verificar e organizar bases de dados com o objectivo de uma melhor utilização da informação guardada”

```
.INDEX ON nome TO nome_ord
```

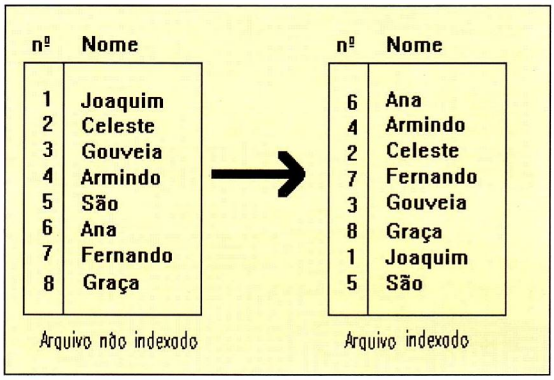
Também poderíamos organizar a base de dados segundo vários campos:

```
.INDEX ON nome+morada+telefone TO nome_ord
```

Se existirem dois registos com o mesmo NOME, o dBase irá verificar o campo MORADA para determinar como posicionar os registos no ficheiro de índices. Para activar um ficheiro de índices que já tenha sido criado, usamos o comando SET INDEX:

```
SET INDEX TO <ficheiro de índice>
```

Se não precisar de ter o ficheiro de índices aberto, utilize o comando CLOSE INDEX. Este comando fechará qualquer ficheiro de indexação que esteja aberto. O dBase pode ter até sete ficheiros de índices abertos ao mesmo tempo, embora apenas um possa estar activo. Se o índice estiver aberto, qualquer alteração efectuada na base de dados fonte será actualizada no arquivo de índice.



Comando SORT

O comando SORT tem como função criar um duplicado organizado da base de dados activa e a sua sintaxe é:

```
SORT TO <novo ficheiro> ON <campo> [/A] [/C] [/D]
      [,<campo2> [/A] [/C] [/D]...] [<registos>]
      [FOR / WHILE <condição>]
```

Copia um ficheiro (.DBF) ou parte dele, para um novo ficheiro (.DBF) que pode ficar em ordem alfabética, numérica ou cronológica de acordo com a especificação do campo.

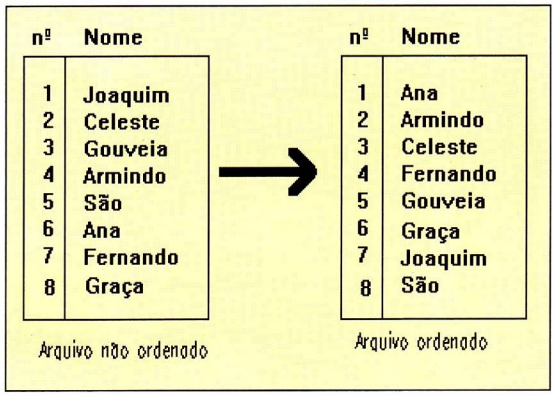
As opções /A e /D permitem a ordenação em ordem ascendente ou descendente, respectivamente. Omitindo tais opções, a ordem será ascendente (sistema padrão). A opção /C ignora a distinção entre letras maiúsculas e minúsculas.

O seguinte exemplo cria uma base de dados ordenada por ordem alfabética de nomes, por ordem descendente e contendo apenas os registos cujos telefones comecem por 943:

```
.SORT TO fich_db ON nome /D FOR telefone="943"
```

Ordenar lógicamente ou físicamente?

Tanto a ordenação lógica como a física, são usadas para verificar e organizar bases de dados com o objectivo de uma melhor utilização da informação guardada, através de programas escritos em dBase. A ordenação física permite que os arquivos possam ser ordenados em ordem crescente ou decrescente, enquanto a ordenação lógica só pode ser realizada em ordem crescente. A grande vantagem da indexação é a reorganização imediata e automática da base de dados, enquanto que a organização física apenas é actualizada quando se executa o comando.



Salvaguardando algumas situações específicas, aconselhamos a ordenação lógica (indexação).

PROGRAMA AGENDA I

Por absoluta falta de espaço na disqueta deste número, não foi possível incluir a primeira parte do programa AGENDA.

NO PRÓXIMO NÚMERO...

- Listagem de uma base de dados.
- Novas opções para a Agenda: etiquetas, aniversários, procura de informação, etc.
- Introdução ao CLIPPER.

Vitor Manuel Monteiro



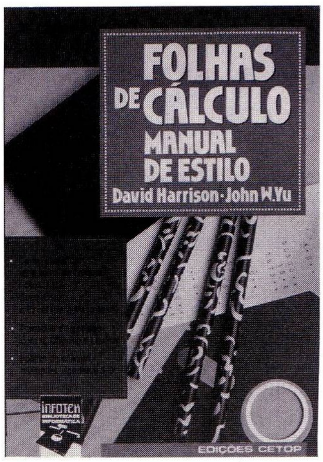
Livros



INTRODUÇÃO À PROGRAMAÇÃO EM PASCAL

Nesta obra ensina-se programação prática baseada em Pascal, a linguagem franca da ciência dos computadores. Serve também de introdução metódica à linguagem Pascal, tanto para iniciados como para peritos noutras linguagens de programação. Esta edição é compatível com o padrão ISO proposto para Pascal. A obra contém muitos exemplos e exercícios no final de cada capítulo, alguns dos quais com as respectivas soluções. Vários apêndices complementam o tratamento dos vários capítulos.

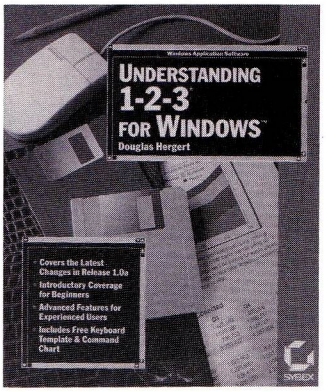
Título: Introdução à Programação em Pascal
Idioma: Português
Autores: William Findlay e David A. Watt
Tradução: Ana Cristina Calaço Dias
Edição e Distribuição: Edições Cetop – colecção Infotek
I.S.B.N.: 972-641-048-7
Preço: 3 200$00



FOLHAS DE CÁLCULO – MANUAL DE ESTILO

Um manual de estilo para a criação de *layouts*, paginação e edição electrónica de folhas de cálculo, gráficos e macros, para PC e Macintosh.
Destinada aos utilizadores de computadores que pretendam resultados imediatos, esta obra oferece:
– «Dicas» simples e práticas para a concepção e execução gráfica de folhas de cálculo e gráficos.
– Regras básicas para construir macros a partir de várias folhas de cálculo e gráficos.
– Um sistema gráfico para criar folhas de cálculo tecnicamente correctas e de fácil utilização.
– Inúmeros exemplos ilustrados de aplicações de todos os tipos.
Os autores, premiados com o «Computer Press Award», são profissionais experimentados na adaptação de material técnico acessível ao leitor comum.

Título: Folhas de Cálculo – Manual de Estilo
Idioma: Português
Autores: David Harrison e John W. Yu
Tradução: Jorge manuel da Silva Neves
Edição e Distribuição: Edições Cetop – Colecção Infotek
I.S.B.N.: 972-641-156-4
Preço: 2 090$00



UNDERSTANDING 1-2-3 FOR WINDOWS

Esta obra é um guia completo e actualizado do Lotus 1-2-3 para o Windows 3.0.
Douglas Hergert, autor já de outras obras de sucesso no âmbito do *software*, aborda neste livro temas tais como:
– Introdução ao ambiente Windows 3.0 para os novos utilizadores deste novo *interface* gráfico.
– Lições passo-a-passo dedicadas a folhas de cálculo, acessíveis aos iniciados ao Lotus 1-2-3.
– Dezenas de exemplos ilustrados, práticos e simples de modo a permitir-lhe conceber folhas de cálculo verdadeiramente úteis e bem estruturadas.
– Dos princípios básicos aos tópicos médios e avançados, da*«linkagem»* das folhas de cálculo à gestão de bases de dados e utilização de macros para automação do seu trabalho, este livro leva-o, de modo gradual e simples, a tomar conhecimento de todas as capacidades do Lotus 1-2-3 para Windows.

Título: Understanding 1-2-3 for Windows
Idioma: Inglês
Autor: Douglas Hergert
Edição: Sybex
Distribuição: Lidel – Edições Técnicas, Lda.
I.S.B.N.: 0-89588-845-9
Preço: –



FUNDAMENTAL DO QUATR0 PRO 3

Embora o «Quatro Pro» seja uma aplicação vasta e sofisticada, a longa experiência do autor desta obra, permitiu-lhe elaborar este livro dando a um tema complexo, uma simples e fácil compreensão, na qual se procura transmitir os conhecimentos sobre o que faz, para que serve e como se manipula uma folha de cálculo.
Pode dizer-se que o «Quatro Pro» é, essencialmente, uma «porta de entrada» na aplicação da Borland, salientando-se:
– Como conhecer o cenário do «Quatro Pro»;
– Como introduzir e manipular dados;
– Como elaborar macros;
– Como trabalhar no banco de menus.
Além dos temas referidos este livro contém muito mais informação. Assim, e para se tornar um livro de fácil consulta o seu autor teve o cuidado de o dividir em catorze capítulos.

Título: FUNDAMENTAL DO QUATRO PRO 3
Idioma: Português
Autor: Renato Reis
Editora: FCA - Editora de Informática, Lda
Distribuidora: Lidel - Edições Técnicas, lda.
ISBN: 972-722-001-0
Preço: 1350$00

Estes e outros livros sobre informática editados pela **Sybex, QUE, Ziff, Editorial O Livro, FCA, Edições Cetop** e **Graficria**, podem ser encomendados através da **Montra Spooler**.
Para o efeito utilize o cupão respectivo.
Os assinantes da **Spooler** beneficiam de 10 % de desconto.

COLABORAÇÃO DO LEITOR

LIVRE
FAST

O nosso amigo Nelson Pires enviou-nos quatro utilitários, três escritos em Pascal e um em *Assembly*. Neste número da Spooler só podemos publicar dois deles: LIVRE e FAST. Os restantes, TECLADO e QCD, serão publicados no próximo número, em conjunto com as fontes dos quatro utilitários. O clássico problema do espaço na disqueta...

Este pequeno programa indica o número de bytes livres num disco (ou disquete), o seu número total, e percentagem de espaço livre. Através da sua utilização, não precisamos estar a ver toda a listagem de ficheiros e directorias que ocorre quando se executa um DIR. A sua utilização é do tipo:

LIVRE [drive]

Se o *drive* não for especificado, então é assumido o *drive* actual. Embora as funções **disksize** e **diskfree** existam na Unit Dos, optei por escever as minhas próprias versões destas funções, para assim não ter de incluir a livraria Dos, e desta forma, «poupar» mais alguns bytes.

O LIVRE coloca um ponto de três em três números para facilitar a leitura dos valores, o que no caso de números grandes (os do disco rígido) ajuda um pouco. Alguns dos procedimentos do LIVRE foram escritos em *assemby*, utilizando os fantásticos recursos que esta versão do turbo pascal permite aos seus utilizadores. Devo referir que o algoritmo para a função CONVERTE (é a que coloca os pontos nos resultados), levou-me mais de duas horas a descobrir, visto que foi feito por «tentativas».

FAST

Junto segue também um pequeno ficheiro, o FAST.COM, que altera o teclado para os valores máximos permitidos. Este ficheiro é para colocar no AUTOEXEC.BAT, e é por isso que é tão pequeno.

Nelson Pires

Gest Pro
Software de Gestão Profissional
Gestão modular de Clientes, Facturação, Encomendas de Clientes, Cobranças, Fornecedores, Encomendas a Fornecedores, Stocks, Etiquetagem (Código de Barras Inclusivé), Produção, Bancos, Caixa, Letras, Vendedores, Comissões e Mailing .
Consulte-nos, peça uma demonstração.
SITAC
SITAC - Sistemas Informáticos, Tecnologia, Assistência e Consultadoria, Lda.
Rua Jacinto Nunes, 18 r/c Dº - 1100 LISBOA - Telefs.: 815 51 39/815 50 11 - Fax: 815 42 12

A DINASTIA INTEL

(PARTE II)

A CAMINHO DO FUTURO: 80286

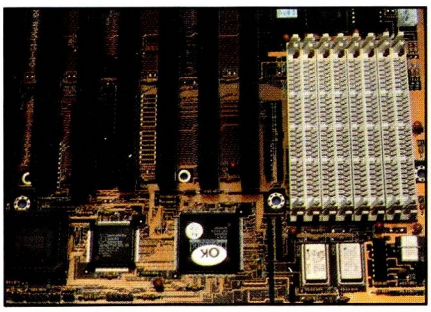
Em 1982, precedendo em apenas alguns meses a introdução do 80286, a INTEL apresentou o 80186 o qual, possuido uma arquitectura idêntica ao 8086, incluia na mesma pastilha de silício mais alguns componentes. O uso deste processador não se tornou prática corrente em compativeis PC, na altura ainda inexistentes, nem nos «azuis» oficiais, dada a introdução, pouco depois, do 80286. O 186 viu, no entanto, uma boa aceitação no mercado dos controladores integrados e, mais recentemente, em placas de expansão para PC, onde é um componente importante da arquitectura EISA (*Extended Industry Standard Architecture*).

No entanto, o 286 foi mais que uma evolução: foi uma «revolução». Apresentando um desempenho quase 3 vezes superior a um 8088 à mesma velocidade de relógio, proporcionou aos utilizadores um aumento significativo de produtividade. Porém, mais importantes mas sem dúvida menos conhecidas, foram as alterações efectuadas na sua arquitectura.

A IBM introduziu este processador na sua linha PC/AT, a qual apresentava melhorias significativas, tais como barramento de expansão de 16 bits, monitores EGA, disquetas de 5,25" de alta capacidade, um BIOS substancialmente expandido e melhorado e o DOS Versão 3. Além disso, ao mesmo tempo prometeu um novo sistema operativo que aproveitasse as novas capacidades do 286, o que viria, mais tarde, a tornar–se realidade sob a forma do OS/2. Quais são então as novidades introduzidas pelo 286 que justifiquem o abandono de um sistema operativo, e a criação de outro a partir do zero?

O 286, apesar de ser integralmente compatível com o seu predecessor, era capaz de funcionar de um modo substancialmente diferente – o chamado modo protegido – ao contrário do modo real em que é uma veloz réplica de um 8086.

Em modo protegido, o 286 faz a gestão de memória de um modo radicalmente diferente, apesar de se basear ainda em segmentos. Os registos de segmento, em vez de possuirem o endereço a dividir por 16, como em modo real, contêm um índice para uma de duas tabelas, onde cada entrada descreve um segmento. Além de conterem o endereço inicial do segmento, guardam o tamanho deste, que pode ser variável, e um conjunto de atributos, tal como a proibição de escrita. Para aceder agora a um endereço, o processador consulta o registo de segmento através do qual acede à tabela, depois adiciona o deslocamento contido no outro registo. Por fim



verifica se o endereço é «legal», isto é, se está contido no segmento e se respeita os atributos.

A existência de duas tabelas de tradução de endereços reflecte a vocação multitarefa do 286, sendo a primeira a GDT *(Global Descriptor Table)* que é única e comum a todos os processos, e as LDT *(Local Descriptor Table),* cada uma particular ao processo a que está associada. Os endereços estão assim virtualizados: vários processos podem escrever numa mesma posição de memória, sem que se atropelem mutuamente, pois os mesmos endereços virtuais estão em LDT's diferentes atribuídos a endereços absolutos distintos.

Ao mesmo tempo, um programa é impedido de danificar a área de memória pertencente a outro, por dois motivos: ou os endereços do outro programa não figuram na sua LDT, ou se figuram na GDTque, como já vimos, é partilhada, estão marcados como proibidos, o que vai gerar um interrupção, a qual é adequadamente tratada pelo sistema operativo.

No Windows 3.0, que em processadores 286 corre em modo protegido, a mensagem *Unrecoverable Application Error* é o resultado da invasão por parte de um programa de um espaço de memória que não lhe pertence. Se corrermos a mesma aplicação num 8086, este erro não aparece, porque o processador não é capaz de o detectar, o que pode levar a um estoiro do sistema, caso a «invasão» tenha danificado alguma zona crítica.

Este método de gestão de memória abre o caminho ao que é conhecido como «memória virtual», a qual, quando necessária, pode ser copiada para disco. Ao mesmo tempo marcam–se como «inválidas» na LDT e/ou na GDT (um segmento pode ser acedido ao mesmo tempo por vários selectores com atributos diferentes) as entradas correspondentes a essa zona de memória. Quando um programa tenta aceder a esse segmento, o processador «vê» que é um segmento inválido e gera uma interrupção que o sistema de memória virtual captura. (Note–se que em modo protegido as interrupções funcionam de um modo diferente, embora a filosofia básica seja a mesma). Esta rotina pode, então, examinar o selector que provocou o problema e carregar do disco os dados em questão, devolvendo de seguida o controlo ao programa que nem sequer suspeita que foi interrompido.

Em modo protegido, o processador também fiscaliza as instruções, executadas de acordo com o CPL *(Current Privilege Level),* o nível de privilégio, que é um valor entre 0 e 3. O nível 0 é o mais privilegiado e é onde se executa o núcleo do sistema operativo. As aplicações executam–se em nivel 3, onde têm menos privilégios, para não poderem fazer estoirar o sistema através de um comportamento indevido. No nível 1 encontra–se o resto do sistema operativo e no nível 2 *software* intermédio tal como*device drivers* e*debuggers*. Os segmentos de memória podem ser marcados com um nível de privilégio e, assim, este só pode ser acedido pelos programas correndo nesse nível ou inferior. (É preciso ter em conta que quanto mais privilegiado é um nível, menor ele é.)

Um programa a correr em nível 3 necessita, obviamente, de recorrer aos serviços do sistema operativo, que corre num nível inferior ao seu. Esta transição de nível pode ser efectuada através de vários mecanismos bastante complexos, pois necessitam de proteger as camadas mais profundas do sistema, do exterior menos priveligiado e potencialmente menos bem comportado.

O 286 também protege os portos de entrada/saída através de uma mapa de permissões, impedindo que os programas de aplicação recorram directamente ao *hardware* – situação indispensável a uma multitarefa segura. O CPL também comanda as instruções que podem ser executadas. Algumas só podem ser invocadas a partir de nível 0. Entre estas encontram–se as que modificam a MSW *(Machine Status Word)*, que é um registo especial que controla o processador, ou as que alteram a LDT ou GDT corrente.

O modo protegido do 286 revela também a sua vocação multitarefa através de um tipo especial de segmento de memória, o TSS *(Task Switch State)* que é particularmente adequado para guardar um processo congelado e que serve de suporte aos sistemas operativos multitarefa. A mudança de contexto *(task switch)* é também particularmente complexa, pelo que não cabe aqui uma análise em pormenor. Pode resumir–se, no entanto, como a salvaguarda de todos os registos de uso geral do processador tal como se apresentam de momento, e na sua reposição a partir do TSS de outro processo congelado. Esta mudança várias vezes por segundo cria a ilusão

de vários processos a executar simultaneamente. Sistemas operativos sofisticados como o OS/2 ou o Unix utilizam elaborados algoritmos para repartir, de forma adequada, o tempo entre os vários processos e garantir um bom desempenho geral.

Outra das inovações do 286 em relação ao 8086 foi a capacidade de gerir até 16Mb de memória física, através do alargamento do barramento de endereços para 24 bits. Mais uma vez esta característica só era utilizável em modo protegido, pois dependia da nova gestão de memória.

No entanto, a opção pelo modo protegido, com todas as vantagens que acarreta, não pôde ser tomada, o que ditou o relativo fracasso do OS/2 até aos nossos dias, pois era inconcebível o investimento necessário para reescrever todo o *software* existente.

Quando os engenheiros da INTEL projectaram o 286 nos princípios da década de 80, não podiam prever o estrondoso sucesso do DOS; estavam pois convencidos que rapidamente seria adoptado o modo protegido e que o modo real serviria apenas para inicializar o sistema. Assim, o processador tinha uma maneira muito simples de entrar em modo protegido, mas uma vez instalado, não havia retorno fácil para o modo real, pois era necessário recorrer a *hardware* externo. Desta forma, o modo protegido viu–se relegado a um papel secundário no DOS, onde servia apenas para aceder aos Kb extra que muitos PC/AT começaram a trazer de origem. O modo protegido permitiu trazer o Unix para os PC's, sob a forma do Xenix/286 o qual, no entanto, ficaria grato se o libertassem da memória segmentada.

Ultimamente, o 286 pôde começar a ser utilizado no seu potencial completo, pois as necessidades de memória dos novos programas não cabiam nas 20 linhas de endereço do 8086. Desta forma nasceram as tecnologias DOS *Extender* , que permitem a programas DOS serem facilmente transportados para modo protegido. O Windows 3.0 é também um DOS *Extender* quando corre em modo *Standard* ou *Enhanced* , embora, como todos sabem, seja bastante mais do que isso.

Os 286 foram fabricados pela INTEL para correrem com frequências de 6,8,10 e 12MHz. No entanto, outras companhias aperfeiçoaram o desenho original e conseguiram levá–los até 16 e 20MHz, tornando–se as principais produtoras deste integrado.

Hoje em dia os 286 começam a apresentar as marcas da idade, e já não são escolha quando o desempenho é a preocupação fundamental, pois a sua velocidade pelos padrões actuais não é de modo nenhum fenomenal e, o que ainda é mais importante, não conseguem correr a verdadeira avalanche de *software* 386 que começa a inundar o mercado. Representam, no entanto, ainda um considerável número de máquinas a trabalhar e podem ser considerados como uma boa compra se o que se pretende é um sistema de baixo custo e entrada da gama. O preço dos 286 nos dias que correm tornam a compra de um 8086 já um erro.

Os 286, nas suas versões de baixo consumo, dominam ainda no mercado dos *notebooks*, computadores portáteis extremamente compactos e com autonomia para várias horas, mas também este pode ser considerado o derradeiro sucesso: o canto do cisne após quase 10 anos de idade.

OS 32 BITS: FINALMENTE!

Quando o 8086 ainda era um rei, já a Motorola, com o seu 68000 (o qual, embora externamente ainda fosse de 16 bits), fazia conhecer ao mundo as maravilhas dos 32 bits, deixando a INTEL bastante para trás em termos de desempenho. Em 1985 foi então apresentado o 80386, o primeiro da linha 80x86 a possuir uma largura de palavra de 32 bits. Estava lançado o maior sucesso comercial da INTEL até então.

As novidades do 386 em relação ao seu predecessor foram os 32 bits e um novo método de gestão de memória. Foram introduzidas novas instruções que, entre outras coisas, suportam os novos tipos de dados de 32 bits. Foram acrescentados dois novos registos de segmento de uso geral, GS e FS, e expandidos os restantes para 32 bits passando a chamar–se EAX em vez de AX, ESP em vez de SP e assim sucessivamente. Note–-se que os registos de segmento permanecem de 16 bits, o suficiente para acolher um selector de modo protegido.

A MSW do 286 deu lugar a 4 registos de controlo CRx, (onde x é um inteiro entre 0 e 3) o primeiro dos quais efectivamente a contém. É através destes registos que é possivel controlar o complexo sistema de gestão de memória.

Foram ainda adicionados mais 8 registos de nome DRx, (onde x é um inteiro de 0 a 7) e que suportam novas facilidades de *debugging* , bastante mais evoluídas do que as dos seus predecessores.

Com o aumento do tamanho dos registos cresceu igualmento o espaço endereçável que é agora de 232–1 bytes, ou seja 4096Mb: 4Gb! Este total está ainda fora do alcance dos PC's actuais, mas promete espaço suficiente para crescer. Este admirável número também é o tamanho máximo de um segmento.

As entradas na GDT e LDT foram expandidas de modo a suportar endereços de 32 bits e outras características novas, como segmentos capazes de crescer, que servem para alojar a *stack* (pilha). Assim, deixa de ser necessário reservar memória para a *stack*, sem haver o perigo de esta encher.

Com o 386 também se pode dizer adeus ao famigerado sistema de gestão de memória segmentada, pois se criarmos um segmento a partir do endereço absoluto 0, e com o tamanho da memória física do sistema, podemos pôr de lado os registos de segmento e usar apenas os de deslocamento que, como são de 32 bits, cobrem a memória toda.

Este modo é o ideal para sistemas como o Unix, que se começaram a vulgarizar na plataforma PC, havendo mesmo quem preveja que acabarão por dominar o mercado.

Quando se trabalha num ambiente de memória linear como o que foi descrito acima, torna–se necessário algum mecanismo que forneça a protecção intertarefa indispensável. O 386 consegue–o através de um mecanismo que se designa por «paginação».

A tradução de um endereço virtual num endereço físico, num 386 com o mecanismo de paginação activado, passa por uma fase adicional em relação ao 286, logo depois de descodificado o par segmento/deslocamento. Nesta fase o espaço de memória físico é visto como uma sucessão de páginas contíguas de 4Kb. Através de um sistema de tabelas, cuja gestão é ainda mais complexa que a das GDT e LDT, qualquer página física pode ser «vista» através de qualquer página lógica. Ou seja, poderemos atribuir um endereço linear virtual a qualquer endereço virtual físico. Com este método podem mover–se dados de um sítio para o outro sem mover efectivamente um único byte: basta alterar o seu endereço. Cada uma destas páginas possui ainda um conjunto de atributos, destinados a realizar a protecção e através dos quais se pode implementar um sistema de gestão de memória virtual idêntico àquele que podia ser conseguido através da segmentação, mas mais eficiente, dado o tamanho reduzido e fixo das páginas.

A incapacidade do 286 de lidar com os programas existentes e aproveitar ao mesmo tempo as suas características avançadas não passou despercebida aos arquitectos do 386, pois, além de ser capaz de entrar e sair do modo protegido sem o recurso a *hardware* externo (pela simples alteração de um bit num dos registos de controlo), tem mais uma mão cheia de características interessantes. A mais importante é o chamado modo«8086–virtual». Como já sabemos, o 286 era capaz de trocar rapidamente entre várias tarefas em modo protegido. O 386 é capaz de alternar transparentemente entre processos 16 bits, como no 286, os novíssimos processos 32 bits, e entre processos que simulam o modo de endereçamento do 8086, ou seja, no modo real. Basta criar um sistema operativo que depois intercepte as excepções (interrupções) levantadas pelo programa a correr na máquina virtual 8086 ao tentar aceder directamente ao *hardware* e as traduza em acessos E/S do modo protegido. Um exemplo deste tipo de programas é o Windows 3.0 a correr em modo *Enhanced* , ou o DESQview/386. Podemos, numa máquina 386 com um destes programas, correr diversas aplicações DOS em simultâneo, sem o mínimo problema.

Outra aplicação do modo 8086–virtual são os gestores de memória tais como o QEMM/386 e 386–To–The–Max da Quarterdeck e da Qualitas, respectivamente. Estes utilitários levam o processador para modo protegido e correm uma sessão DOS. Ao mesmo tempo activam o mecanismo de paginação para «encher» de memória todos os buracos livres no espaço de endereçamento de 1Mb do 8086 e que simulam memória expandida EMS 4.0.

Os DOS *Extender* que, no 286, proporcionavam 16Mb de memória, são em versão 386 ainda mais úteis, porque permitem aos programas libertar–se da segmentação do modo real ao mesmo tempo que passam a usar os 32 bits em pleno. O aumento de desempenho pode ser espectacular, como o demonstrou o AutoCAD/386, em relação ao AutoCAD 10.

Estas aplicações, apesar de serem boas novas, não conseguem esconder o facto

dramático de a maioria dos 386 existentes estarem a correr em modo real, servindo como 8086 rápidos. Os programas 8086 dão-se hoje em dia a grandes trabalhos para ultrapassar as dificuldades impostas por um espaço de memória de apenas 1Mb e segmentado. É cómico que os 386 tenham que «fingir» possuir as mesmas limitações que um 8086, para depois as poderem contornar, perdendo tempo valioso nos dois processos...

O popularização do Unix/386 e o aparecimento para breve do OS/2 Versão 2, um sistema operativo que aproveita as potencialidades 32 bits e gestão de memória do 386, mostram alguma luz no fundo do túnel, mas o *software* DOS ainda está para lavar e durar...

O 386 é fabricado pela INTEL nas frequências de 16,20,25 e 33Mhz, estando as duas mais baixas já um pouco desactualizadas pelo aparecimento do...

SX: 32 BITS PARA TODOS?

Tendo perdido por completo o controlo da fabricação do 286, a INTEL , num genial golpe de *marketing,* decidiu «matá-lo». Com uma vasta campanha na imprensa apresentou o 386SX, um processador 386 completo, mas com o barramento externo reduzido a 16 bits, um pouco à maneira como o 8088 nasceu do 8086. Este *chip* permitia produzir computadores a preços semelhantes aos dos 286, mas com um desempenho semelhante à mesma velocidade de relógio. Possuia, no entanto, a «magia» dos 32 bits que lhe permitia correr todo o *software* do seu irmão mais velho: o 386 agora baptizado de 386DX.

Foi o fim do mercado dos 286 de alta *performance,* a 16 e 20MHz, produzidos apenas por terceiros. Curiosamente, são estas duas as frequências de relógio do SX.

Mais recentemente, apanhada pela onda dos portáteis, a INTEL anunciou uma nova evolução do 386SX, de nome 386SL, preparado para correr à velocidade de 20MHz. Este processador, além de possuir todas as características dos seus predecessores, é especialmente adaptado para a construção de placas minúsculas e com um consumo irrisório. Inclui na mesma pastilha de silício o controlador de *cache* , até então um integrado separado. Acompanhado do seu *chip* «irmão», o 82368SL, pode-se implementar um PC completo com apenas mais uma meia dúzia de integrados. Além disso inclui uma série de características orientadas para a gestão de alimentação, conseguindo um consumo sem precedentes em sistemas deste nível de desempenho, pois o SL consegue ser ainda mais rápido que o SX equivalente. Ainda são raros os computadores com este *chip* como coração, mas sem dúvida que se lhe pode prever um sucesso estrondoso.

Recentemente, depois de uma batalha jurídica desesperada por parte da INTEL para impedir a produção de processadores compativeis com o 386, a AMD *(Advanced Micro Devices)* conseguiu trazer à luz do dia as suas versões do 386, entre as quais as mais interessantes são uma versão de baixo consumo que, embora não tão sofisticada como o SL, pode produzir resultados idênticos e uma versão a 40MHz, colocando um não-INTEL no topo do desempenho 386. As primeiras máquinas estão a aparecer no mercado...

O CÉU É O LIMITE: 486 E A SUBIR...

A acompanhar todos os processadores da linha 80x86 desde sempre existiu a linha 80x87 de coprocessadores matemáticos, capazes de aumentar espectacularmente o desempenho dos 80x86 em aplicações intensivas do ponto de vista numérico, tais como CAD e folhas de cálculo. A partir do 386 começou a sentir-se a necessidade de uma memória *cache* que acelerasse os acessos à preguiçosa RAM, portanto o 386 teve como companheiro, quase sempre, um controlador de *cache* e um punhado de Kb's de memória estática (SRAM), muito mais rápida que a DRAM normal.

Para os anos 90 a INTEL apresentou o i486, uma pastilha de silício que contém 1.200.000 transistores, em comparação com os 29.000 do 8086 original! Este processador inclui, além da unidade de processamento compatível com o 386, um processador matemático (pelo que não há nem vai haver 487), um controlador de *cache* e 8Kb de memória rápida. Tudo isto agregado na mesma pastilha, produziu um efeito espectacular: 3 vezes o desempenho de um 386 a correr à mesma velocidade de relógio! Fabricado inicialmente para a velocidade de 25MHz e depois para 33MHz, muito simplesmente explodiu as marcas dos bancos de teste, quaisquer que estas fossem. O 8086 está a anos-luz de distância.

Não houve, no entanto, alterações profundas na arquitectura básica em relação ao 386, apenas um punhado de instruções e de alterações nos registos de controlo.

O 486 é isso mesmo, a prova que o todo é maior que a soma das partes. Não se faz nada num 486 que não se possa fazer num 386 com os integrados auxiliares. Faz-se, no entanto, substancialmente mais depressa.

No 486 a INTEL aplicou alguns dos paradigmas da tecnologia RISC, tal como a implementação das instruções mais comuns num único ciclo de relógio; porém é um CISC e assim continuará a linha 80x86, uma vez que a INTEL já se comprometeu com a compatibilidade retroactiva.

Para melhorar ainda mais o seu desempenho, o 486 suporta um modo de «rajada» *(burst mode)* no acesso à memória. Esta característica permite ao processador realizar vários acessos à memória num único ciclo, com o objectivo de encher rapidamente a *cache* interna com dados novos, sempre que for necessário.

À semelhança do que aconteceu com o 386, a INTEL anunciou o 486SX, porém desta vez a fórmula é diferente. Não faria sentido cortar o barramento do 486 para metade, portanto, o que a INTEL fez foi desactivar o coprocessador matemático dentro do 486 e reduzir-lhe a velocidade para apenas 20MHz, que (por coincidência) origina um desempenho semelhante ao 386 a 40MHz da AMD.

No entanto, a INTEL afiança que o (seu) 486SX será mais rápido que o 386 (da AMD) quando se encontrar a correr código 32 bits, para o qual está especialmente optimizado, não sendo portanto os programas DOS (logo, de 16 bits), uma boa medida de comparação.

Este processador situa-se, assim, entre o 386 a 33MHz e o 486 a 25MHz. Para complementar o 486SX, é necessário, de novo, um processador matemático, o 487SX, que é nada mais que um 486 verdadeiro a correr apenas a 20MHz. Depois de instalado o 487SX, é mesmo possivel retirar o 486SX da placa do sistema... Mais uma vez a INTEL recua tecnologicamente para aquilo que muitos já vêem como um sucesso comercial tão grande como o 386SX! Brilhante.

Para aumentar ainda mais os rumores de que a INTEL estar a tentar «assassinar» o seu *best-seller,* o 386 (como tinha feito com o 286), o advento do 486SX foi acompanhado da notícia de que o 386 não evoluirá mais, e da redução dos preços de toda a sua linha. O SX permanecerá nos 20MHz e o DX nos 33MHz. Mais uma vez nos diz que o futuro está no que a INTEL e só a INTEL fabrica: o 486.

...ATÉ AOS 100 MILHÕES DE TRANSÍSTORES!

Podemos esperar para a próxima década desenvolvimentos tão ou mais interessantes que os da década de 80. Para começar, a INTEL já prometeu o 486 a 50MHz, que se espera que quebre a barreira do som! Já nas «bancas» em 92. Mostrou também a interlocutores priveligiados o 486 a 100MHz: isso mesmo! Nunca mais o Windows será lento...

Anunciou também a intenção de construir o 486SL, segundo a linha do 386SL. Quem quer um portátil 8086?

Os primeiros pormenores do i586 devem ser conhecidos já no próximo ano, e podemos ter a certeza que não serão apenas um aumento de velocidade. Processador gráfico dedicado e paralelismo estão na lista de possibilidades. Só nos resta esperar...

Para o ano 2000 já temos prometido o i786, que deve correr a 250MHz, e conter numa pastilha 100 milhões de transistores. Fala-se na possibilidade deste processador incluir 4 unidades de processamento idênticas a um 486 actual, um *cache* de 2Mb, uma unidade de gráficos de qualidade equivalente à da norma de TV de alta definição e dois processadores vectoriais, uma técnica emprestada pelos super-computadores e já usada pela INTEL no seu RISC maravilha: i860. Por outro lado, o i786 também referido como o Micro-2000, poderia conter grande parte da electrónica necessária à construção de um PC do próximo milénio, incluindo o reconhecimento de escrita ou de fala!

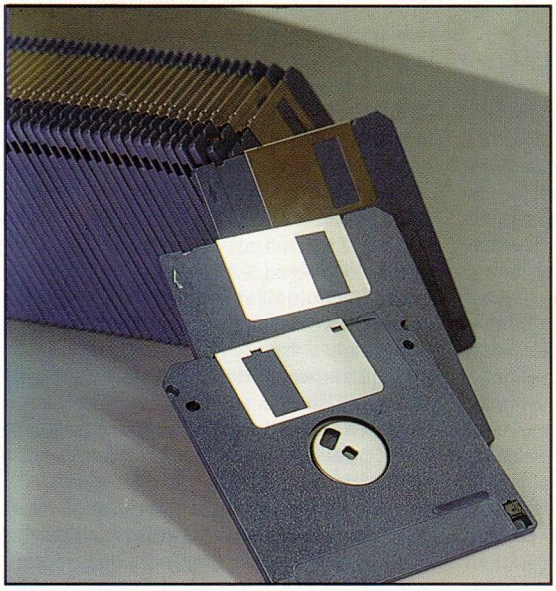
O desempenho deste processador seria da ordem dos 1000 milhões de instruções por segundo (MIPS), contra os cerca de duas dezenas de MIPS do 486 actual.

Comecem a juntar as vossas economias...

José Orlando Pereira

MONTRA Spooler

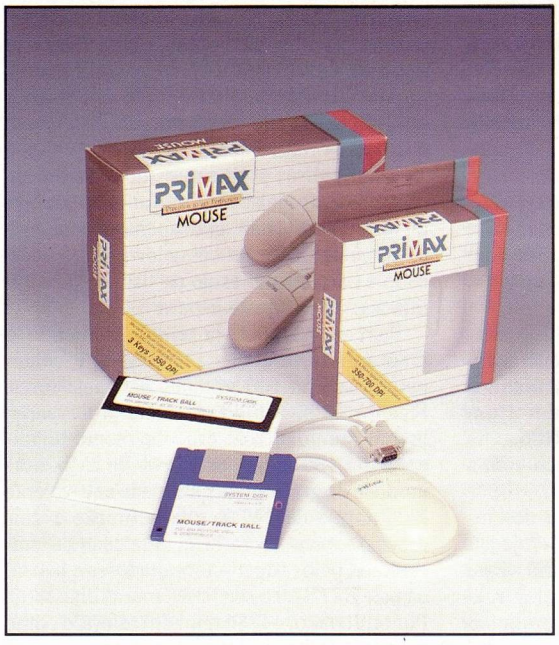
DISQUETAS BULK (sem marca)



50 DISQUETAS 3,5" DS/DD
Refª. D040 **Preço: 6 032$00**

50 DISQUETAS 3,5" DS/HD
Refª. D041 **Preço: 10 406$00**

RATOS PRIMAX



«RATO» COMPATÍVEL MICROSOFT E MOUSE SYSTEM
350-700 DPI – DYNAMIC RESOLUTION – 3 TECLAS

Embalagem com o «rato» e software de instalação
Refª PR01 **Preço: 5 052$00**

Embalagem com o «rato», software, tapete, caixa e ficha adaptadora 9/25 pinos
Refª PR02 **Preço: 6 485$00**

10% de desconto
para assinantes
em todos os produtos da Montra

TODOS OS PREÇOS APRESENTADOS JÁ INCLUEM O IVA.

SOUND BLASTER PRO



SISTEMA PROFISSIONAL PARA RECOLHA, EMISSÃO E TRATAMENTO DE SOM, COMPATÍVEL DOS 3.0 E WINDOWS 3.0 (E SUPERIORES).

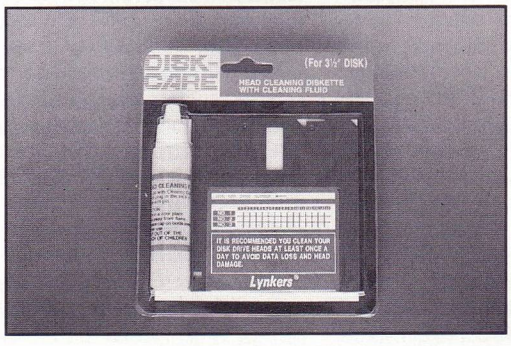
Gravação estéreo; compatibilidade mono com AdLib e Sound Blaster; Twin FM chips proporcionando 22 vozes; misturador estéreo digital/analógico FM, DAC, line-in, microfone, CD e altifalante interno de PC; software para regulação de fade, pan, e volumes direito e esquerdo, mistura com microfone; cabo Midi *standard*; *Interface* CD-ROM com conector interno para CD-Audio; porta para *Joystick*; Amplificador de microfone com AGC (Automatic Gain Control); 4 W por canal; Controlo manual de volume na placa; etc...

A embalagem Sound Blaster Pro contém: Carta Sound Blaster Pro 16 bit; Cabo RCA Audio; Cabos Midi; Disquetas de 3,5" e 5,25"; Software SPJ Sequencer; Editor de voz II; SBTALKER (software texto/voz); Software FM (orgão inteligente); reprodutor CD; demonstração MMPLAY; *drivers* para DOS e Windows.

Requisitos em Hard e Software: PC AT compatível (286 ou superior), mínimo de 512 Kb RAM, DOS 3.0 ou superior, carta EGA ou VGA (VGA recomendada)

Refª SB01 **Preço: 51 272$00**

KITS PARA LIMPEZA DE DRIVES

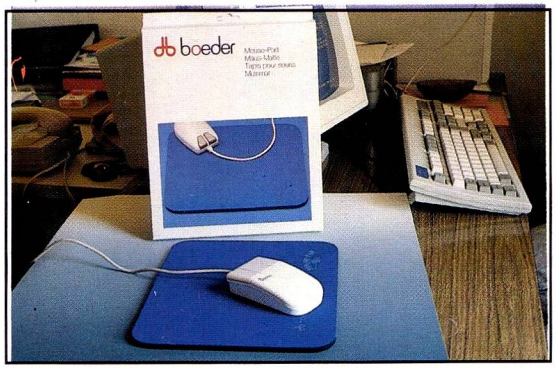


Ref.ª G001
Disqueta 3 ½
602$00

Ref.ª G002
Disqueta 5 ¼
602$00

Kits de limpeza de drives incluindo disqueta e frasco com o líquido de limpeza. Qualidade comprovada.

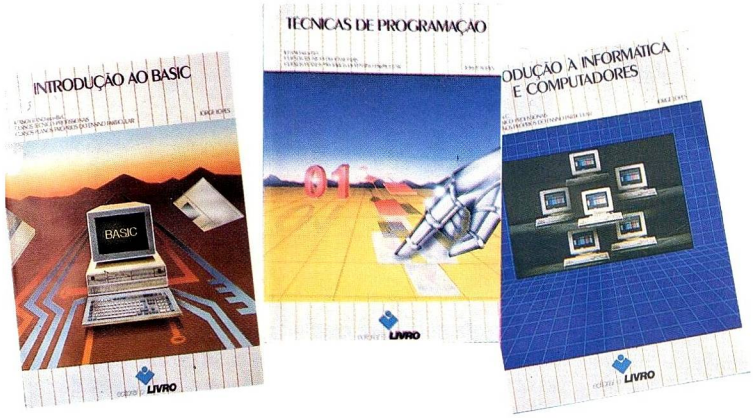
TAPETE PARA MOUSE



Tapete para Mouse de elevada qualidade. Anti-estático. Permite-lhe uma maior precisão e facilidade no seu trabalho.

Ref.ª G005 **PREÇO: 871$00**

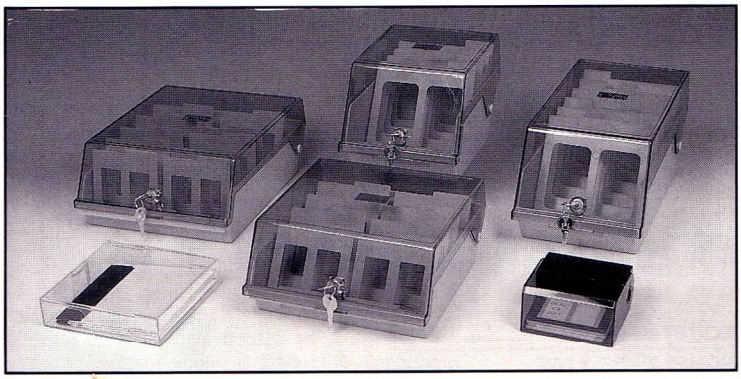
LIVROS EDITORIAL O LIVRO (Em português)



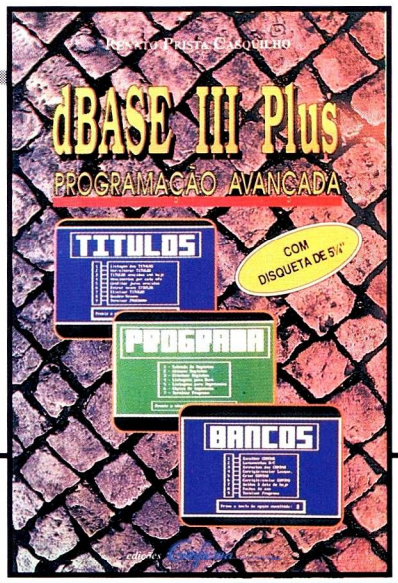
– INTRODUÇÃO À INFORMÁTICA E COMPUTADORES
REF.: L001 **PREÇO: 1 806$00**

– TÉCNICAS DE PROGRAMAÇÃO
REF.: L002 **PREÇO: 1 575$00**

– INTRODUÇÃO AO BASIC
REF.: L003 **PREÇO: 2 352$00**



– 10 disquetas 3,5"	Ref.: H004	Preço:	272$00
– 10 disquetas 5,25"	Ref.: H005	Preço:	344$00
– 50 disquetas 5,25"	Ref.: H006	Preço:	1 131$00
– 100 disquetas 5,25"	Ref.: H007	Preço:	1 328$00
– 50 disquetas. 3,5"	Ref.: H008	Preço:	1 192$00
– 80 disquetas 3,5"	Ref.: H009	Preço:	1 356$00
– 100 disquetas. 3,5"	Ref.: H010	Preço:	1 500$00



DBase III Plus
PROGRAMAÇÃO AVANÇADA

Refª. GR01 COM DISQUETA DE 5,25" **1 890$00**

Refª. GR02 COM DISQUETA DE 3,5" **1 995$00**

INVISUAIS: Quando solicitado forneceremos em disqueta suplementar o texto integral do livro.

POWER PAK – AT



Unidade de Estabilização e Manutenção de energia eléctrica, indicada para protecção da configuração completa de PC AT.
Bateria selada, sem manutenção.
Potência máxima: 550 VA – Autonomia: 25 minutos.

GARANTIA DE 1 ANO

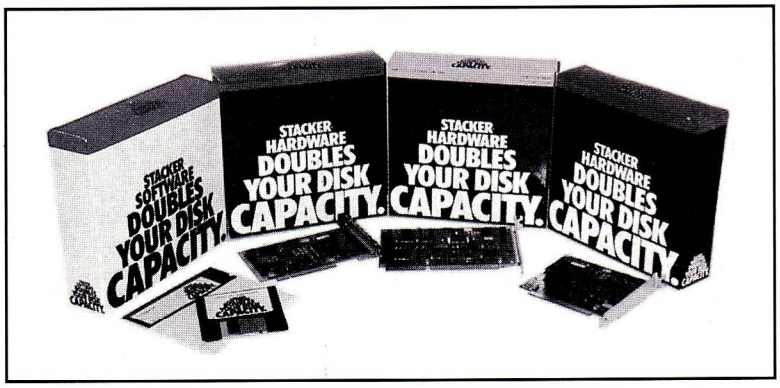
Refª. PK01 **PREÇO: 52 003$00**

PAPEL CONTÍNUO

Embalagem de 500 folhas de papel contínuo (zebra americana) 12x9,5"

Refª. P020 **Preço: 785$00**

STACKER



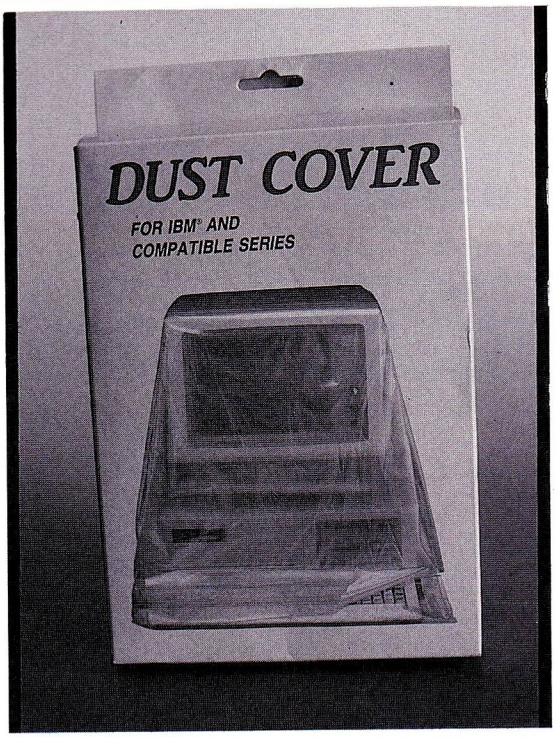
STACKER – PARA DUPLICAR A CAPACIDADE DO SEU DISCO E INCREMENTAR O DESEMPENHO DO SISTEMA

O que obtém com o Stacker:
- O dobro da capacidade do seu disco (de qualquer tipo)
- Rápida compressão em tempo real
- «Cache» que aumenta consideravelmente o desempenho do sistema
- 100% de compatibilidade
- Instalação rápida e fácil (basta abrir a caixa do seu PC, colocar a placa **Stacker** numa *slot* livre e «correr» o software de instalação com instruções em português)
- 1 ano de garantia da placa

Conteúdo da embalagem:
- Placa **Stacker**
- Sofware **Stacker** (em disquetas de 3,5" e 5,25")
- Certificado de garantia
- Manual de instruções (em inglês)

Refª ST01	**Só software**	**26 100$00**
Refª ST02	**Placa + Software para processadores XT**	**43 848$00**
Refª ST03	**Placa + Software processadores AT (286/386/486)**	**49 300$00**

CAPA PROTECTORA PARA PCs E IMPRESSORAS



Capa protectora para computadores em plástico especial reforçado para a completa protecção do seu equipamento contra fumos, poeiras, humidade e sujidades de toda a espécie.
Adaptável a qualquer tipo de PC e impressoras.

Ref.: G006 – Capa para PC	**1 674$00**
Ref.: G007 – Capa p/ imp. 80 c.	**913$00**
Ref.: G008 – Capa p/ imp. 132 c.	**913$00**

REEDIÇÃO DOS NÚMEROS ESGOTADOS DA SPOOLER (4, 5 e 6)

Depois de esgotada a reedição dos números 1, 2 e 3 da Spooler e para satisfazer a procura manifestada por inúmeros leitores vamos tornar a imprimir os números 4, 5 e 6, agrupados num só volume.

Esta edição tem uma tiragem reduzida e não será distribuida.

O único modo de possuir um exemplar é reservá-lo, desde já, preenchendo o cupão anexo (ou fotocópia) e remete-lo para os nossos serviços:

SPOOLER MAGAZINE
Rua Duarte Pacheco Pereira,
69-A – Damaia-de Baixo
2700 AMADORA

CUPÃO DE RESERVA DA EDIÇÃO ESPECIAL SPOOLER – NÚMEROS 4, 5 E 6

NOME .....

ENDEREÇO .....

.....

C. POSTAL ..... LOCALIDADE ..... TELEF. .....

DESEJO RECEBER A EDIÇÃO ESPECIAL «SPOOLER 4 - 5 - 6»

☐ **COM 3 DISQUETAS DE 5 1/4" ..... 1 200$00**
☐ **COM 2 DISQUETAS DE 3 1/2" ..... 1 200$00**

Não mande qualquer pagamento agora. A edição especial ser-lhe-á enviada à cobrança.
Aos preços indicados serão acrescidos os portes de envio e taxa de cobrança dos C.T.T.

Assinatura

COMPUTADOR SPOOLER 386/33

VEJA NA PÁGINA 9
A MÁQUINA
QUE A SPOOLER
ESCOLHEU E PREPAROU
PARA SI

WINDOWS 3.0 (EM PORTUGUÊS)

Não vale a pena possuir cópias piratas. Além da ilegalidade o utilizador está sempre privado dos «upgrades» que vão surgindo e das vantagens que um Manual (neste caso em português) lhe oferece para optimização do seu sistema.

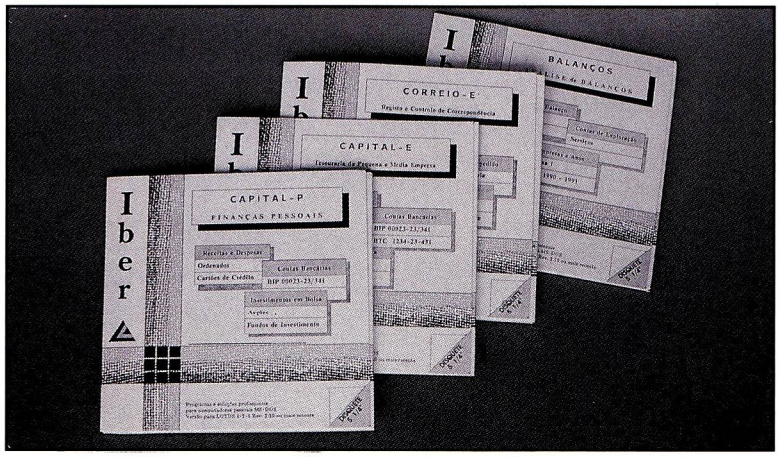
Refª. W001	COM DISQUETAS DE 5,25"	39 324$00
Refª. W002	COM DISQUETAS DE 3,5"	39 324$00

DOS 5.0 (EM PORTUGUÊS)

Conheça as novas e poderosas potencialidades do sistema operativo MS-DOS 5.0. Tudo em português.

Refª. MS01	COM DISQUETAS DE 5,25"	20 300$00
Refª. MS02	COM DISQUETAS DE 3,5"	20 300$00

SOFTWARE IBER



Programas e soluções profissionais para compatíveis MS-DOS. Não requerem conhecimentos de Lotus 1-2-3 ou de MS-DOS.

Refª IB01*	CAPITAL-P – FINANÇAS PESSOAIS.
Refª IB02*	CAPITAL-E – TESOURARIA DE PEQUENA E MÉDIA EMPRESA.
Refª IB03	CORREIO-E – REGISTO E CONTROLO DE CORRESPONDÊNCIA..
Refª IB04*	BALANÇOS – ANÁLISE DE BALANÇOS.
Refª IB05	BIBLOS – GESTÃO DE LIVROS, VIDEOS E DISCOS.
Refª IB06	CATÁLOGO – APOIO A COMPRAS – PRODUTOS FORNECEDORES – PREÇOS

Em disquetas de 3,5" (720 Kb) PREÇO: 3 620$00

***Versões para Lotus 1-2-3 – Rev 2.10 ou mais recente**

CUPÃO MONTRA SPOOLER

Recorte ou fotocopie o cupão e remeta-o para:

SPOOLER MAGAZINE
Rua Duarte Pacheco Pereira, 69-A – Damaia
2700 AMADORA

ESTIMADO LEITOR:
Tentamos entregar as suas encomendas o mais rápido possível.
Quando um número da **Spooler** sai ao público, somos «bombardeados» com muitos milhares de pedidos.
Não desespere se não receber a sua encomenda no dia seguinte e **por favor não a devolva.** Os preços por nós praticados (quase isentos de lucro) não permitem o encargo adicional de ter que pagar os portes do retorno da encomenda. Comunique-nos se pretender, à *posteriori*, desistir da encomenda.

CUPÃO DE ENCOMENDA – MONTRA SPOOLER

Cliente nº.:
A PREENCHER PELOS NOSSOS SERVIÇOS

Desejo receber à cobrança (com os **custos acrescidos de portes e taxas**) os seguintes produtos:

Quant.	Refª.	Designação	Preço
		TOTAL	

NOME .....
MORADA .....
C. POSTAL ..... LOCALIDADE .....
TELEFONE ..... ❑ NÃO SOU ASSINANTE ❑ SOU ASSINANTE Nº .....

Spooler 18

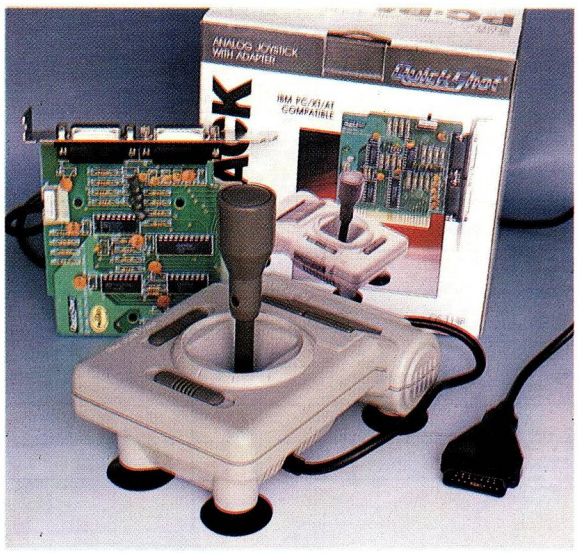
QUICKJOY M-5



Compatível para PC/XT/AT este joystick de *design* ergonómico é o ideal para os seus jogos de acção.
A placa pode ser adquirida em separado.

Refª G004 **Preço: 4 289$00**

QUICK SHOT – JOYSTICK + PLACA



Joystick para compatíveis PC/XT/AT e placa. Óptimo desempenho e *design* atractivo. Um novo dinamismo nos seus jogos predilectos. Um preço incrível!

Refª G003 **Preço: 6 825$00**

DISQUETAS PLATINUM



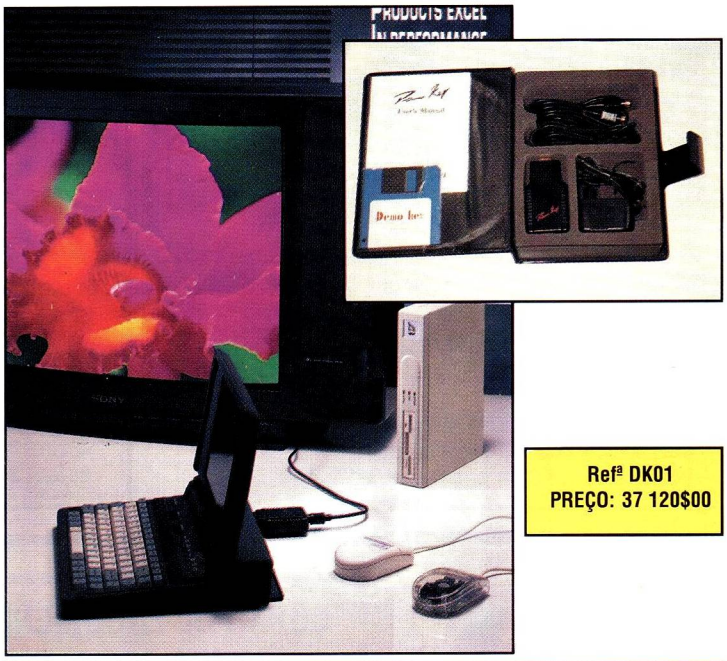
Caixas com 10 disquetes:			
– 5,25" DS/DD	Ref.: PL01	Preço:	905$00
– 3,5" DS/DD	Ref.: PL02	Preço:	1 463$00

PLACA QS-120 PARA JOYSTICK

Placa para joystick

Refª G04A **Preço: 3 861$00**

DEMOKEY (VGA TO VIDEO)



Refª DK01
PREÇO: 37 120$00

Dispositivo que permite transformar em sinal vídeo qualquer saída VGA de computador.
Ideal para os possuidores de portáteis ou *notebook*, ou de monitores VGA monocromáticos, que assim podem visualizar os seus programas confortavelmente no monitor colorido do seu receptor de televisão.
Também indicado para demonstrações e exibições, uma vez que o sinal VGA do computador pode ser «desdobrado» e visualizado em receptores de televisão montados em cadeia. Outra possibilidade é a de permitir registar as imagens do seu computador no seu videogravador.

- Resoluções até 640x480, 256 cores
- Saída vídeo NTSC ou PAL
- Portas de saída AV Vídeo e S-VHS Vídeo

A prática embalagem do Demokey contém:

- O Demokey
- Transformador de alimentação de 9 V
- Cabos de ligação para AV Vídeo e S-VHS
- Manual e disqueta de instalação (em inglês)

CONVERSOR TV PARA VGA



O Conversor TV permite transformar o seu monitor VGA num receptor de televisão.

É tão simples quanto isto: você liga o cabo de antena à tomada do Conversor TV, liga a ficha VGA do seu PC ao conversor e a ficha vídeo do conversor ao seu monitor VGA. E pronto, em qualquer momento e sem interferência no trabalho que estava a executar no seu PC, você pode visionar (e ouvir) o seu programa de televisão favorito.
O Conversor TV também pode ser ligado ao videogrador. O manual de utilização (em castelhano) explica tudo muito bem explicadinho.
À semelhança dos modernos receptores de TV, o Conversor TV dispõe de sintonia e busca automática de canais, 40 pré-sintonias, regulações de nível de som, contraste, luminosidade e saturação de côr, etc.

Características técnicas:
Sistema de TV CCIR • Sistema de côr PAL • Selector de banda VHF e UHF • Pré-sintonias: 40 • Amplificador de potência e altifalante incorporados • Entrada RF coaxial de 75 W • Alimentação 229 V 50 Hz A. C. • Saída RGB analógica: 31,5 KHz horiz.; 50 Hz vert. • Consumo: 15 W • Dimensões: 400 x 140 x 82 mm • Peso: 1 700 g aprox.

Refª TV01 **Preço: 52 200$00**

MODEMS BEST DATA



Modems de alta qualidade a preços notáveis.

Refª BD01 Modem interno – 2400 bps W/MNP 2-7 por software V.22 e V.22 bis) **Preço: 20 880$00**

Refª BD02 Modem externo– 2400 bps W/MNP 2-7 por software V.22 e V.22 bis) **Preço: 33 408$00**

Refª BD03 Fax/Modem interno – 9600 bps Fax – 2400 bps Modem W/MNP 7 por software V.22 e V.22 bis) **Preço: 41 760$00**

Refª BD04 Fax/Modem externo – 9600 bps Fax – 2400 bps Modem W/MNP 7 por software V.22 e V.22 bis) **Preço: 48 024$00**

Nota: *estes Modems aguardam registo de homologação*

RATO MIC - MI-260

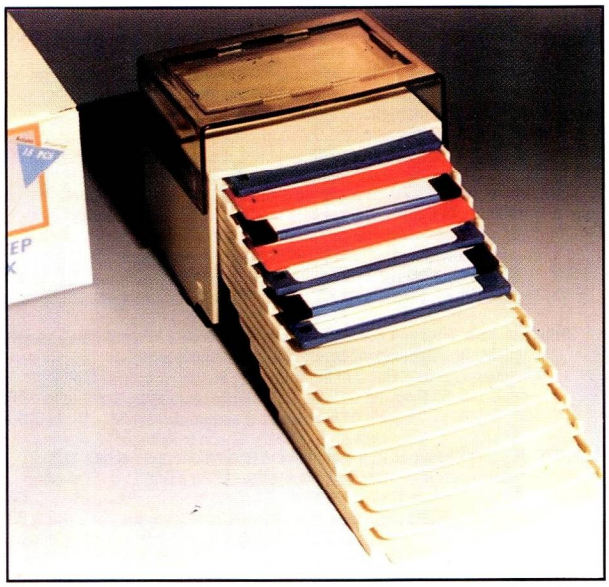


Regressado ao nosso convívio cá está o rato MIC, com nova embalagem e muito, muito mais barato!

Características: 260-2600 DPI, Compatível Microsoft e Mouse System, com comutador no corpo do rato. 3 teclas. Inclui: Software de instalação e teste, ficheiros de ajuda (em inglês) – Ficha adaptadora 9/25 pinos.

Refª MIC1 **Preço: 2 500$00**

CAIXA ARQUIVADORA - 15 DISQ. 3,5"



Caixa arquivadora «elástica» para o acondicionamento perfeito das suas disquetas de 3,5". Versátil e compacta.

Refª BX01 **Preço: 950$00**

FREE-SOFT E SHAREWARE

O valor indicado não paga os direitos destes programas, mas sim, apenas as próprias disquetas e o trabalho de cópia e distribuição. Será o utilizador dos programas tipo «shareware» a ter a responsabilidade de efectuar o respectivo pagamento, directamente aos autores, nos termos e condições por eles propostos. A Spooler limita-se a distribuir esses programas, devidamente testados e na sua versão integral.

A86 – Assemblador e desassemblador – Eric Issacson (86-87-88)
Documentação em francês e inglês
Refª. SH01 – 1 disqueta de 5,25" ..... 500$00
Refª. SH02 – 1 disqueta de 3,5" ..... 600$00

QModem SST – Versão 4.2E – Utilitário de comunicações
– The Forbis Project Inc. – Doc. em inglês
Refª. SH03 – 3 disquetas de 5,25" ..... 1 500$00
Refª. SH04 – 2 disquetas de 3,5" ..... 1 200$00

Thedraw – Versão 3.40 – Utilitário de desenho, animação e tratamento da imagem em modo texto – Doc. em inglês – The Soft Prog. Serv.
Refª. SH05 – 1 disqueta de 5,25" ..... 500$00
Refª. SH06 – 1 disqueta de 3,5" ..... 600$00

Pkzip – Versão 1.10 – Compactador e descompactador de ficheiros
– Pkware Inc. – Doc. em inglês.
Lha – Versão 2.11 – Compactador e descompactador de ficheiros
– Haruyasu Yoshizaki – Doc. em inglês.
Refª. SH07 – 1 disqueta de 5,25" ..... 500$00
Refª. SH08 – 1 disqueta de 3,5" ..... 600$00

List – Versão 7.5 E– O famoso List com respectivos utilitários
– Vernon D. Buerg – Doc. em inglês.
VPIC – Versão 4.2 – Leitor de Gifs – Bob Montgomery – Doc. em inglês
Picem – Versão 2.1 – Leitor de Gifs – John Bridges – Doc. em inglês
Refª. SH09 – 1 disqueta de 5,25" ..... 500$00
Refª. SH10 – 1 disqueta de 3,5" ..... 600$00

Boxer – Versão 3.03 – Editor de texto muito completo e versátil
– David R. Hamel – Doc. em inglês.
Refª. SH11 – 1 disqueta de 5,25" ..... 500$00
Refª. SH12 – 1 disqueta de 3,5" ..... 600$00

Utilitários para DOS – Selecção de diversos utilitários para DOS.
(Manipulação ecrã, rotinas para impressora, caracteres medievais para VGA, utilitários para Batchs, gestores de memória, etc...)
Diversos autores – Doc. em inglês.
Refª. SH13 – 1 disqueta de 5,25" ..... 500$00
Refª. SH14 – 1 disqueta de 3,5" ..... 600$00

Utilitários para DBase – Selecção de diversos utilitários para DBase.
(drivers para impressora, etiquetas, exemplos de base de dados relacional, etc....) – Diversos autores – Doc. em inglês.
Refª. SH15 – 1 disqueta de 5,25" ..... 500$00
Refª. SH16 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Clipper – Selecção de diversos utilitários para Clipper
(Janelas para Clipper, rotinas para o rato, escudos por extenso, dicas e truques, editor de campo memo, animação, etc...)
Diversos autores – Doc. em inglês.
Refª. SH17 – 1 disqueta de 5,25" ..... 500$00
Refª. SH18 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Windows 3.0 – Diversos utilitários para Windows 3.0
(Almanaque, Ascii, Biorritmo, File Manager (aceita zips), etc...)
Diversos autores – Doc. em inglês.
Refª. SH19 – 1 disqueta de 5,25" ..... 500$00
Refª. SH20 – 1 disqueta de 3,5" ..... 600$00

CD-MAN – Jogo estilo «Pac-man» com um grafismo fora de série.
– Creative Dimensions – Doc. em inglês – EGA / VGA
Refª. SH21 – 1 disqueta de 5,25" ..... 500$00
Refª. SH22 – 1 disqueta de 3,5" ..... 600$00

Dark_ages – Jogo de arcada/aventuras. Apogée Software Productions
– Doc. em inglês. – EGA / VGA.
Refª. SH23 – 1 disqueta de 5,25" ..... 500$00
Refª. SH24 – 1 disqueta de 3,5" ..... 600$00

Vampyr – Jogo de aventuras. Género medieval/fantástico.
Brian Weston/Victor Shao – Doc. em inglês – EGA / VGA.
Refª. SH25 – 1 disqueta de 5,25" ..... 500$00
Refª. SH26 – 1 disqueta de 3,5" ..... 600$00

GIFS – Imagens digitalizadas
(Para visionar estas imagens necessita de um editor de GIFs. Disponível nas referências SH09 e SH10).

– Imagens de desenhos animados
Refª. SH27 – 1 disqueta de 5,25" ..... 500$00
Refª. SH28 – 1 disqueta de 3,5" ..... 600$00

– Imagens do mundo animal
Refª. SH29 – 1 disqueta de 5,25" ..... 500$00
Refª. SH30 – 1 disqueta de 3,5" ..... 600$00

– Garotas. Belas imagens
Refª. SH31 – 1 disqueta de 5,25" ..... 500$00
Refª. SH32 – 1 disqueta de 3,5" ..... 600$00

– Paisagens
Refª. SH33 – 1 disqueta de 5,25" ..... 500$00
Refª. SH34 – 1 disqueta de 3,5" ..... 600$00

– Imagens de filmes
Refª. SH35 – 1 disqueta de 5,25" ..... 500$00
Refª. SH36 – 1 disqueta de 3,5" ..... 600$00

– Diversos. Raquel Welch, cão punk, carro antigo, etc.
Refª. SH37 – 1 disqueta de 5,25" ..... 500$00
Refª. SH38 – 1 disqueta de 3,5" ..... 600$00

-Paisagens diversas, um quadro «à Van Gog», S.Francisco, Chicago, a Guerra das Estrelas, uma águia e o pato Dodger. .
Refª. SH55 – 1 disqueta de 5,25" ..... 500$00
Refª. SH56 – 1 disqueta de 3,5" ..... 600$00

Utilitários para DOS
– COPYIT: um copy muito melhorado, com pedido de mais disquetas, se necessário. LO: temporizador de teclado com envio de dados para o ecrã. XEQ: agrupa executáveis num único ficheiro, podendo chamar-se qualquer um isoladamente. CHANGE: muda atributos, data e hora. YAY: utilitário com editor, calendário, tabela ASCII, pesquisa de palavras e SHELL. RUN: corre programas a partir de outra drive e/ou directório. 1PASS: diskcopy melhorado.
Refª. SH39 – 1 disqueta de 5,25" ..... 500$00
Refª. SH40 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Clipper
– CLIPCALC: simulação de uma calculadora. COPYPROT: proteção contra cópias. NANFORLIB: diversas bibliotecas para Clipper. SB: utilitário para execução de menus, caixas, textos, colorir, etc.
Refª. SH41 – 1 disqueta de 5,25" ..... 500$00
Refª. SH42 – 1 disqueta de 3,5" ..... 600$00

Aplicações para Windows
– DBFBROWSER: visualização e tratat. de Bases de Dados. DICE: jogo de dados. ICONDRAW: para desenhar icons. FIREDRAW: simulação de fogo de artifício, FISH3: simulação de aquário, HUNTER: pesquisa de ficheiros em qualquer drive. WINEZ: cria nova janela de comando para execução de aplicações.
Refª. SH43 – 1 disqueta de 5,25" ..... 500$00
Refª. SH44 – 1 disqueta de 3,5" ..... 600$00

Utilitários para C
– CBD: ferramentas para desenvolvimento de bases de dados para programadores de C, compatível com Microsoft C e Turbo C. CLOCK: rotina para inclusão de um relógio digital TSR. DBLOOK: visualização de .DBF's do dBASE. EXEC: fontes para a função «EXEC» que permite chamar programas externos, opcionalmente trocando a memória de imagem para EMS, XMS ou para um ficheiro. Compatível com Turbo C (V. 1.x, 2.x e C++ 1.0), Borland C V. 2.0, Microsoft (V. 5.1 e 6.0), Watcom (V. 8.0), Turbo Pascal (V. 4.x e 5.x). MOUSEC: rotinas para rato em Microsoft C ou Turbo C. PROTOC: execução de protótipos de funções. TRAVDIRS: executa um comando específico em qualquer directório.
Refª. SH45 – 1 disqueta de 5,25" ..... 500$00
Refª. SH46 – 1 disqueta de 3,5" ..... 600$00

Turbo Toolkit V.5.0
– Fontes e Manual. Construção de janelas, menus, entradas do utilizador, formação de cadeias de caracteres, listagem de directórios, etc. Compatível com Turbo Pascal V. 4.0 e 5.0.
Refª. SH47 – 1 disqueta de 5,25" ..... 500$00
Refª. SH48 – 1 disqueta de 3,5" ..... 600$00

Keen – Jogo para EGA e VGA.
Refª. SH49 – 1 disqueta de 5,25" ..... 500$00
Refª. SH50 – 1 disqueta de 3,5" ..... 600$00

Kroz e Quest – Jogos (todas as placas)
Refª. SH51 – 1 disqueta de 5,25" ..... 500$00
Refª. SH52 – 1 disqueta de 3,5" ..... 600$00

Paga1 – Jogo (todas as placas)
Refª. SH53 – 1 disqueta de 5,25" ..... 500$00
Refª. SH54 – 1 disqueta de 3,5" ..... 600$00

Utilitários Diversos. Diversos autores. Documentação em inglês.
QFIND: Pesquisa de qualquer ficheiro no sistema. Inclui utilitário para Windows 3.0. CLEAN84 e SCANV84: As versões V 7.9V84 do famoso «Viruscan» da McAfee. VCOPY82: Um DOS COPY com detecção automática de vírus V 0.7V82 da McAfee. KEYLOCK: Um eficiente cadeado para o teclado.

PUTPASS: Umpassword para aceder a ficheiros .COM.
PULLDWN: Um cómodo e eficiente «menu» para gerir aplicações
Refª. SH57 – 1 disqueta de 5,25" ..... 500$00
Refª. SH58 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Windows 3.0
PCASE – CAD para Windows: Versão de demonstração.
UNICOM: Programa de comunicações.
WUNZIP: «Zipar» e «Dezipar» dentro do Windows.
ZIPAPER: «Wall-paper» comprimido pronto a utilizar.
Refª. SH59 – 1 disqueta de 5,25" ..... 500$00
Refª. SH60 – 1 disqueta de 3,5" ..... 600$00

Utilitários e jogos para Windows 3.0
CUBIC: O famoso jogo dos cubos.
NEWICONS: Biblioteca de ícones.
WINFRA20: Visualização de fractais.
XVTDRAW: Utilitário para desenho de precisão.
Refª. SH61 – 1 disqueta de 5,25" ..... 500$00
Refª. SH62 – 1 disqueta de 3,5" ..... 600$00

Utilitários diversos
EMSTOOL2: O famoso «ToolKit» da Intel para programadores de C, Assembly e Pascal.
HRAM1: Um gestor para memória alta.
DOSMAP2: Apresenta o mapa da memória.
Refª. SH63 – 1 disqueta de 5,25" ..... 500$00
Refª. SH64 – 1 disqueta de 3,5" ..... 600$00

Utilitários diversos
VGACOPY: Copiador de disquetas em VGA
GIFDOT: Imprime GIF's em vários formatos
CODEP: Converte codepage 850 em 860 e vice-versa
DRVINST: Instalador de Device-drivers
HELPSPEC: Conjunto residente de caracteres especiais
PCCALC: Calculadora TSR
BREAKIUP: Parte ficheiros grandes em vários pequenos
SILENCE: Silencia o altifalante
VGADISK: Instala um disco virtual a partir da memória do VGA
Refª. SH65 – 1 disqueta de 5,25" ..... 500$00
Refª. SH66 – 1 disqueta de 3,5" ..... 600$00

V860V321
O famoso package de José Páscoa, destinado à adequada instalação do code page português, poupando precioso espaço na memória convencional e substituindo as obsoletas instruções do MS-DOS. Inclui ainda as seguintes capacidades:
– Acesso a todas as teclas do teclado AT extendido a nível do DOS
– Clique do teclado configurável
– Redefinição do typematic rate e delay rate do teclado
– Possibilidade de desinstalação do programa da memória
– Alarme musical configurável
Tem mais os seguintes utilitários:
MODO.EXE: Altera o modo gráfico utilizando os serviços do DOS
GEN.EXE: Gera e captura fontes para utilização com FONTES.EXE
PLACA.EXE: Identifica a placa gráfica
Refª. SH67 – 1 disqueta de 5,25" ..... 500$00
Refª. SH68 – 1 diqueta de 3,5" ..... 600$00

A/B – HIPERTEXTO
BlackMagic: Processador em «Hypertext», versão 1.2. O sistema chamado de «hipertexto» permite expandir textos ligando palavras-chave e blocos de informação a outros textos.
Refª. SH69 – 2 disquetas de 5,25" ..... 1 000$00
Refª. SH70 – 1 disqueta de 3,5" ..... 600$00

Utilitários
SHEZ: Versão 6.3 do famoso «Shell» altamente especializado para manipulação de compressões. Corre em Windows 3.0 e suporta o rato.
COLORIT: Utilitário para colorir textos ASCII.
PKLITE: Compressor de ficheiros .EXE e .COM mantendo-os executáveis.
HDM4: Gestor do disco rígido com password.
Refª. SH71 – 1 disqueta de 5,25" ..... 500$00
Refª. SH72 – 1 disqueta de 3,5" ..... 600$00

Utilitários e jogos para Windows 3.0
DISKLOOK: Informação permanente sobre os seus discos.
KLZ209: O Tetris para Windows 3.0
MAGIC110: Um screen-saver com password e diversas configurações
BG06: O jogo do Gamão para Windows 3.0
PUZZLE: Um puzzle para Windows 3.0
WINWEEL: Roda da Sorte para Windows 3.0
IEDIT: Editor de ícones para Windows 3.0
WIN3CLOC: Um relógio configurável para Windows 3.0
CHANGCUR: Altere o cursor do rato em Windows 3.0
Refª. SH73 – 1 disqueta de 5,25" ..... 500$00
Refª. SH74 – 1 disqueta de 3,5" ..... 600$00

Utilitários para DOS
MATCH201: Compara documentos linha a linha. Útil para programadores.
AGENDAP: Calendário residente desde 1901 até 2089. CNFMT16: Formata disquetas enquanto se executa outra tarefa. ARJ221: A versão V2.21 do célebre compressor. SHERIFF: Um administrador de segurança para o sistema.
EEMRAM21: Aumenta a memória disponível até 736K com recurso à LIM.
Refª. SH75 – 1 disqueta de 5,25" ..... 500$00
Refª. SH76 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Windows 3.0
METZW30 – CONJUNTO DE UTILITÁRIOS PARA WINDOWS 3.0
DN.EXE – Desktop Navigator: Acesso rápido a drives, directórios e ficheiros. Funções de gestão de directórios e ficheiros.
DT.EXE – Desktop Manager: Sistema de criação de menus e vários utilitários para acesso a aplicações.
FM.EXE – Metz Freemem: Apresentação digital da memória convencional e expandida, colocada em qualquer ponto do ecrã.
LK.EXE – Metz Lock: Utilitário de segurança com password.
RNR.EXE – Metz Runner: Utilitário para execução rápida de aplicações e ficheiros.
TM.EXE – Metz Timmer: Relógio/Calendário digital colocado em qualquer ponto do ecrã.
PSHOP – PaintShop. Manipulação e conversão de imagens digitalizadas.
GIF2BMP – Converte qualquer GIF para «papel de parede».
GIF2ICON – Converte qualquer GIF para ícon.
NEKO – Coloque um «Pussy-cat» no seu Windows 3.0
Refª. SH77 – 1 disqueta de 5,25" ..... 500$00
Refª. SH78 – 1 disqueta de 3,5" ..... 600$00

Utilitários e jogos para Windows 3.0
WINAV – Utilitário destinado à «navegação» através de ficheiros, directórios e informações contidos em discos ou disquetas. Executa programas, descompacta, edita ficheiros de texto, etc.
MLINK – Programa de comunicações
ZIPMGR – Comprime/descomprime/lista ZIP, LZH, ARC, etc.
IMSTR – Desenhar ícons e integrá-los nos programas.
BLITZER – Jogo de guerra com helicópteros
Refª. SH79 – 1 disqueta de 5,25" ..... 500$00
Refª. SH80 – 1 disqueta de 3,5" ..... 600$00

Utilitários para Clipper e dBASE
AECLIPT – Bibliotecas para gráficos em Clipper.
DB3TPL – Construção de ecrãs em dBASE III Plus
C5AN01 – Anomalias da versão Clipper 5.0
CLIPPMOU – Rotinas em Assembly e em C para o rato
CLRSET – Configurar as cores
SHADOW – Fazer menus com sombras
DBTIME – Transformar intervalos de tempo em horas e minutos
Refª. SH81 – 1 disqueta de 5,25" ..... 500$00
Refª. SH82 – 1 disqueta de 3,5" ..... 600$00

Colecção Pintura : Toulouse Lautrec(5¼")
ANGLAISE.GIF – AU_SALON.GIF – CLOWNESS.GIF – GOULUE.GIF
GOUL_ENT.GIF – JANE.GIF – MAIL.GIF – YVETTE.GIF
Refª. SH83 – 4 disquetas de 5,25" ..... 2 000$00

Colecção Pintura : Toulouse Lautrec (3,5")
ANGLAISE.GIF – AU_SALON.GIF – CLOWNESS.GIF – GOULUE.GIF
YVETTE.GIF – GOUL_ENT.GIF – JANE.GIF – MAIL.GIF – SUZANNE.GIF
Refª. SH84 – 3 disquetas de 5,25" ..... 1 800$00

Utilitários e fontes para C
CPTUTOR1/2 – Tutor para a linguagem C
ASYCLS10 – Rotinas em C++ para comunicações
PCTSR – Demonstração e fontes para residentes
TRAVDIRE – Fontes e utilitário para pesquisa de ficheiros
CRITERRC – Rotinas para manipulação de erros
DOSWIND – Rotinas para construção de janelas no DOS
CMOUSE – Rotinas para integração do rato
Refª. SH85 – 1 disqueta de 5,25" ..... 500$00
Refª. SH86 – 1 disqueta de 3,5" ..... 600$00

GIF's Alta resolução
SHORE – Beira-mar (640x400x256)
WORLDS – Os planetas e seus satélites (640x480x256)
PHOBOS2 – Fotografia espacial de Fobos (640x480x256)
Refª. SH87 – 1 disqueta de 5,25" ..... 500$00
Refª. SH88 – 1 disqueta de 3,5" ..... 600$00

GIF's Alta resolução
MNTCLO – Paisagem (640x480x256)
ASIATIQU – Criança oriental (640x512x16)
Refª. SH89 – 1 disqueta de 5,25" ..... 500$00
Refª. SH90 – 1 disqueta de 3,5" ..... 600$00

Utilitários para DOS
BENCH3D – Avaliação do desempenho de sistemas com VGA
TESTCOM – Teste às placas série
HW20 – Configuração do sistema
MAM107 – Análise da memória ocupada
MDR30A – Diagnóstico aos Modems
RX10A – Aumentar a memória convencional nos 386
CARDTEST – Afinação dos tons em mono e policromático
XDISK252 – Alterar a dimensão das RAM-Disco
DRX097 – Manipulação de ficheiros .EXE e .COM comprimidos
Refª. SH91 – 1 disqueta de 5,25" ..... 500$00
Refª. SH92 – 1 disqueta de 3,5" ..... 600$00

Jogos
SA1ASP – «Secret Agent» da Apogee
CAVES – As« Caves» da Apogee
Refª. SH93 – 1 disqueta de 5,25" ..... 500$00
Refª. SH94 – 1 disqueta de 3,5" ..... 600$00

PROGRAMA PARA FACTURAÇÃO «FACTURA»



Este programa de fácil utilização, instalação simples, textos de ajuda claros e precisos, menus *pulldown*, obedece a todos os requisitos legais em vigor e permite-lhe:

- Criar e manter actuais dados sobre clientes e artigos;
- Imprimir listagens de clientes e artigos de acordo com variadas opções de selecção;
- Manter actualizado o movimento e inventário de stocks;
- Produzir e imprimir propostas, facturas, notas de crédito, notas de débito e guias de remessa.
- Contas correntes, tabelas de preços por artigos e clientes.

REQUISITOS DE HARD E SOFTWARE:

- MS-DOS 3.1 ou superior; PC 100% compatível; 640 RAM ; 2 unidades de disquetas ou 1 unidade disquetas + disco rígido.

Com garantia e assistência técnica ON-LINE. Produzido por XT-Software, Lda.

Refª FT01	**Preço: 17 400$00**

FITAS, TONNERS E TINTEIROS PARA IMPRESSORAS

PARA TODOS OS TIPOS DE IMPRESSORAS (E MÁQ. DE ESCREVER) MARCAS ORIGINAIS E COMPATÍVEIS.

Indique no cupão de encomenda a referência que pretende (ou a marca e modelo da impressora)

TEMOS OS MELHORES PREÇOS.

DISQUETAS SKC



ALTA QUALIDADE EM DISQUETAS

Caixas com 10 disquetas	
Refª SKC1 3,5" DS/DD - 720 Kb	**Preço: 1 640$00**
Refª SKC2 3,5" DS/HD - 2 Mb	**Preço: 2 750$00**
Refª SKC3 5,25" DS/DD - 360 Kb	**Preço: 1 430$00**
Refª SKC4 5,25" DS/HD - 1,2 Mb	**Preço: 1 720$00**

ASSINATURAS

Spooler

Recorte ou fotocopie o cupão, preencha-o com letra bem legível e remeta-o, acompanhado do respectivo pagamento em cheque ou vale de correio, para:

SPOOLER MAGAZINE
R. Duarte Pacheco Pereira, 69-A
Damaia – 2700 AMADORA

*Os números anteriores da **Spooler** podem ser encomendados à cobrança utilizando o cupão da **Montra Spooler**. Ao preço de capa serão acrescidos os custos dos portes e taxas de cobrança.*
Se preferir disquetas de 3,5", isto custar-lhe-á mais 140$00 por exemplar.

NOTA: Os números 1 a 6 e o 10 e 11 da **Spooler** encontram-se esgotados. A reedição especial dos números 1, 2 e 3, também!

Spooler 18

CUPÃO DE ASSINATURA

Assinante nº:
A PREENCHER PELOS NOSSOS SERVIÇOS

NOME .....
DATA DE NASCIMENTO* ____ / ____ / ____
PROFISSAO* .....
TRABALHA COM COMPUTADORES?* SIM☐ NÃO ☐
SE SIM ESPECIFIQUE A MARCA E MODELO* .....
ENDEREÇO .....
.....
C. POSTAL ..... LOCALIDADE ..... TELEF(S) .....

DESEJO ASSINAR A SPOOLER A PARTIR DO Nº. 19 (inclusive) E DURANTE 12 NÚMEROS.

☐ **MODALIDADE COM DISQ. DE 5 ¼" ..... 5 500$00**
☐ **MODALIDADE COM DISQ. DE 3 ½" ..... 6 000$00**

PARA PAGAMENTO DA ASSINATURA ENVIO :
☐ O CHEQUE Nº. ..... S/ O BANCO .....
☐ O VALE DE CORREIO Nº. .....

** A resposta aos itens assinalados com um (*) é opcional.*

THE COMPLETE COLOUR SOLUTION

73 500$ + IVA

«The Complete Colour Solution» proporciona-lhe tirar todo o partido da sua placa VGA!
Você pode capturar imagens a cores de alta resolução a partir do seu vídeogravador caseiro, camâra ou qualquer outra fonte de sinal vídeo.
As imagens (não retocadas) que ilustram esta página foram digitalizadas a 256 cores. São um pequeno exemplo do que você pode obter em apenas alguns segundos.

DÊ LARGAS À SUA IMAGINAÇÃO E CRIATIVIDADE



CONTEÚDO DA EMBALAGEM
- Vidi-PC (Carta para a captura e digitalização de imagens)
- VidiChrome-PC (software para tratamento da cor em alta resolução)
- Vidi-RGB (separador electrónico de cores)
- PC Paintbrush IV Plus (provavelmente o melhor software do mundo para arte e grafismos informáticos)
- Picture Box (Conjunto de demonstração)

SOFTWARE:
Disponível em disquetes de 3,5" ou 5,25"

CARACTERÍSTICAS
Resoluções de imagem possíveis:
- 320x200/256 cores
- 640x400/256 cores
- 640x480/256 cores
- 800x600/256 cores
- Suporta todas as resoluções de 256 cores até 800x600 (EVGA)
- Captura até 4096 cores em menos de 1 segundo
- Avançado algoritmo para selecção de paletes
- Zoom interactivo
- Converte imagens negativas em positivas
- Imprime em todas as impressoras *standadrd* mono ou policromáticas
- Grava as imagens nos formatos Gem, PCX ou Tiff.

APLICAÇÕES:
- Apresentação de produtos ou serviços
- Demonstração e vendas
- DTP a cores (edição computorizada)
- Ensino e formação profissional
- Animação
- Concepção de catálogos e literatura diversa



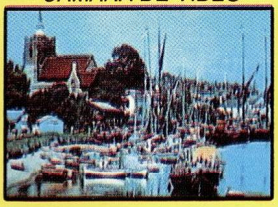
CANON ION

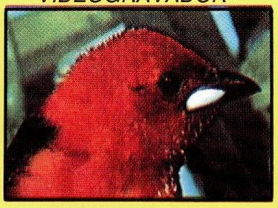


CAMÂRA DE VÍDEO

VIDEOGRAVADOR







Para obter estes resultados, apenas necessita possuir os seguintes equipamentos:
- Computador compatível AT ou XT
- 640 Kb de RAM
- Monitor VGA policromático
- Disco rígido
- Rato
- Uma fonte de vídeo (câmara, videogravador, etc.)

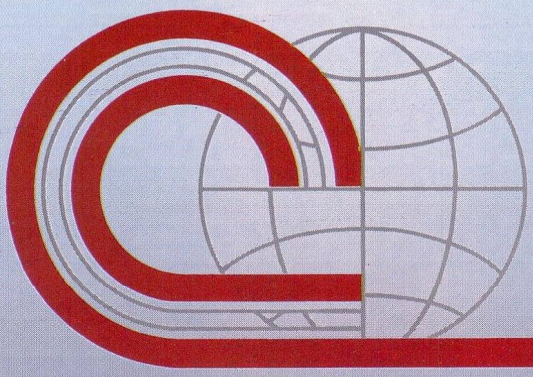
Game® informática

DISTRIBUÍDOR EM PORTUGAL DA ROMBO Limited

Loja 1 – R. Elias Garcia, 362-D – C. C. Babilónia, Loja 70 – 2700 AMADORA – Tel. 492 14 32
Loja 2 – **Departamento Profissional** – Rua D. Pedro IV, Lote 13, Loja C – 2745 QUELUZ
Telef.: 436 15 38 – Tel./Fax: 436 10 08

Graficria pub adapt.

Graficria PUB



Consuldata

SERVIÇOS DE INFORMÁTICA, LDA.

UM MUNDO EM CONSUMÍVEIS PARA INFORMÁTICA

AS MELHORES MARCAS
O MELHOR ATENDIMENTO
A ENTREGA MAIS RÁPIDA
OS MELHORES PREÇOS!!!

DISQUETES DE 3", 3,5", 5,25" E 8"

DATA CARTRIDGES (STREAMTAPES)
DE 10, 20, 45, 60, 150, 250, 320,
525 MEGABYTES E 1,3 GIGABYTES

DISK CARTRIDGES E DISK PACKS
PARA GRANDES SISTEMAS

PASTAS PARA ARQUIVO DE FORMULÁRIOS
DE 9,5x12, 12x9,5, 12x15,5 E 15,5x12
E RESPECTIVOS SEPARADORES

FILTROS ANTI-MAGNÉTICOS E ANTI-ESTÁTICOS
EM REDE E VIDRO PARA TODOS OS TIPOS
DE ECRÃS DE COMPUTADOR

MOBILIÁRIO PRÓPRIO
PARA CENTROS DE INFORMÁTICA

TAPE CARTRIDGES PARA IBM 34800
INCLUINDO CARROS DE TRANSPORTE
E MÓVEIS ESPECIAIS DE ARQUIVO

D. D. S. - 4 MM (1,3 E 2,0 GIGABYTES)
- 8 MM (2,3 GIGABYTES)

E MUITO MAIS!
PARA MAIS INFORMAÇÕES CONTACTE-NOS
OU SOLICITE O CATÁLOGO CONSULDATA

Consuldata - Serviços de Informática, Lda.
Rua Quinta do Almargem, 7 A/B - 1300 LISBOA
Telef.: 364 98 00 Fax: 364 91 08