

TURBO

C

GUIA DE REFERENCIA
RÁPIDA DE TODAS AS

FUNÇÕES
INTERFACE
JANELAS
GRÁFICAS
MATEMÁTICAS
CONTROLE

TURBO C
Guia de Referência Rápida
de Todas as Funções



Ricardo Trevisan e Anselm H. Lehmann

Este é o guia de referência para o software TURBO C. Ele explica como usar o software para escrever programas em C. O software TURBO C é uma versão completa do C com todos os recursos da linguagem. Ele inclui uma biblioteca de funções que pode ser usada para escrever programas complexos. O software também inclui uma interface gráfica para facilitar a programação.

TURBO C

Guia de Referência Rápida

de Todas as Funções

ANO: 1997 96 95 94 93 92 91 90

EDIÇÃO: 9 8 7 6 5 4 3 2 1

LIVROS ÉRICA EDITORA LTDA

TODOS OS DIREITOS RESERVADOS. Proibida a reprodução total ou parcial, por qualquer meio ou processo, especialmente por sistemas gráficos, microfílmicos, fotográficos, reprográficos, fonográficos, videográficos. Vedada a memorização e/ou a recuperação total ou parcial em qualquer sistema de processamento de dados e a inclusão de qualquer parte da obra em qualquer programa juscibernético. Essas proibições aplicam-se também às características gráficas da obra e à sua editoração. A violação dos direitos autorais é punível como crime (art. 184 e parágrafos, do Código Penal, cf. Lei n.º 6.895, de 17.12.80) com pena de prisão e multa, conjuntamente com busca e apreensão e indenizações diversas (artigos 122, 123, 124, 126, da Lei n.º 5.988, de 14.12.73, Lei dos Direitos Autorais).

LIVROS ÉRICA EDITORA LTDA.
Rua Jarinu, 594 - Tatuapé - São Paulo
Fone: 294-8686 - C.G.C. 50.268.838/0001-39
Caixa Postal 15.617

DEDICATÓRIA

Dedicamos este livro às nossas esposas e filhas, que nos incentivaram e apoiaram com sua compreensão, dedicação e carinho, sem os quais não teria sido possível concretizar esta obra.

SUMÁRIO

Terminologia	1
Funções	2
Variáveis Globais	116
Tabela de Cores	121
Agrupamento das Funções por Header	125
Agrupamento das Funções por Categoria	134
Funções de Entrada e Saída	139

INTRODUÇÃO

A elaboração deste manual de REFERÊNCIAS RÁPIDAS PARA TURBO C , teve como meta principal fornecer um apanhado geral das funções do - TURBO C - o mais destacado compilador da atualidade, suprindo desta forma uma necessidade dos programadores nesta linguagem no sentido de lhes permitir uma ampla visão das suas possibilidades, reunindo de forma compacta e de rápida consulta suas funções.

A linguagem C cada vez mais tem conquistado espaço, pois seu poderio e velocidade a colocam em lugar de destaque tornando-a preferida pelos programadores que realizam programas com características profissionais.

Desta forma tornava-se crucial alguma literatura que permitisse a estes programadores uma forma versátil de consultar dados relacionados com as funções , que foram obtidas através de pesquisas, e normalmente não fazem parte de livros específicos deste assunto.

Este manual não tem a pretensão de ensinar LINGUAGEM C, no entanto ele é imprescindível tanto aos PROGRAMADORES EXPERIENTES quanto aos INICIANTES que tenham a intenção de rapidamente se familiarizarem com a linguagem da década.

Indicamos também a aqueles que desejam aprender detalhes de programação outros livros correlatos desta mesma editora.

*RICARDO TREVISAN
ANSELM H. LEHMANN*

TERMINOLOGIA

No decorrer deste MANUAL tentamos torná-lo o mais claro e direto possível, desta forma foram omitidos conceitos que podem ser aqui generalizados, assim temos:

- 1- Funções cujo retorno é um inteiro (int) fica assumido como indicação de erro o valor -1.
- 2- Funções que tratam de arquivos cujo retorno é um inteiro (int) que significa um "file handle" fica assumido como indicação de erro o valor -1 e em operação normal o valor retornado será o próprio "file handle".
- 3- Funções cujo retorno é um pointer (*) fica assumido como indicação de erro o valor NULL.

Quando estas regras acima não forem válidas será informada a situação real na propria função.

A medida do possível tentamos realizar este manual em português no entanto para não fugir muito dos termos utilizados por programadores alguns termos foram mantidos em inglês.

A disposição normal das funções será:

nome_da_função

prototype_da_função

- Esclarecimentos sobre a função
- Estrutura previamente definida → Quando alguma estrutura já definida nos "header files" for utilizada
- Retorno → Quando necessário

Nos prototypes onde existem opcionais, estes foram colocados entre colchetes ([])

A passagem de uma estrutura como parâmetro deve ser feita através de seu endereço da seguinte forma:

função (& estrutura)

abort

void abort (void);

- Termina o processamento de modo anormal

abs

int abs (int num);

- Retorna o valor absoluto de num

absread

*int absread (int drive, int num_de_setores, int primeiro_setor,
void * buffer);*

- Lê num_de_setores do disco para buffer
- Retorno → 0 indica sucesso, -1 indica erro

abswrite

```
int abswrite (int drive, int num_de_setores, int primero_setor,  
             void * buffer);
```

- Escreve num_de_setores do buffer para o disco
- Retorno → 0 indica sucesso, -1 indica erro

access

```
int access (char * path, int mode);
```

- Se mode = 0 determina se o arquivo existe
- mode = 2 determina se é possível escrever no arquivo
- Retorno → 0 indica sucesso, -1 indica erro

acos

```
double acos (double x);
```

- Retorna o arco (em radianos) cujo coseno é x

allocmem

```
int allocmem (unsigned num_de_bytes , unsigned * segp);
```

- Aloca num_de_bytes de 16 bits na memória do DOS
- Retorno → -1 indica sucesso

arc

```
void far arc (int x, int y, int angulo_inicial, int angulo_final,  
              int raio);
```

- Desenha um círculo com centro em x,y

asctime

```
char * asctime (estrutura);
```

- Retorna pointer para string na qual foi convertida data e hora para ASCII
- Estrutura previamente definida → "tm"

asin

```
double asin (double x);
```

- Retorna o arco (em radianos) cujo seno é x

assert

```
void assert (int teste);
```

- Envia mensagem a stderr e termina processamento se teste = 0

atan

```
double atan (double x);
```

- Retorna o arco (em radianos) cuja tangente é x

atan2

```
double atan2 (double y, double x);
```

- Retorna o arco (em radianos) cuja tangente é x/y

atexit

```
int atexit (void * função);
```

- Determina uma função a ser executada ao se sair do programa pela função exit
- Retorno → 0 indica sucesso, não zero indica erro

atof

```
double atof (char * string);
```

- Converte string para número de ponto flutuante

atoi

*int atoi (char * string);*

- Converte string para int

note

atol

*long atol (char * string);*

- Converte string para long

note

bar

void far bar (int esquerdo, int topo, int direito, int base);

- Desenha uma barra bidimensional

note

bar3d

*void far bar3d (int esquerdo, int topo, int direito, int base,
int profundidade, int flag_de_topo);*

- Desenha uma barra tridimensional

note

bdos

int bdos (int função_dos, unsigned dosDX, unsigned dosAL);

- Executa uma função do DOS (INT 21H)
- Retorno → dosAX

bdosptr

*int bdosptr (int função_dos, void * argumento, unsigned dosAL);*

- Idem bdos, para ser usado nos modos compact, large, e huge

bioscom

int bioscom (int cmd, char byte, int port);

- Executa uma comunicação com RS-232
- cmd = 0 - envia byte para o parametro de comunicação
 - 1 - envia byte para transmissão
 - 2 - recebe um caracter (retorno)
 - 3 - retorna status

port é 0 para COM1, 1 para COM2 etc.

- Retorno → MSB 8 bits possuem o status
 - LSB 8 bits cmd = 2 - caracter recebido
 - 3 - status recebido

biosdisk

```
int biosdisk (int cmd, int drive, int head, int trilha, int setor, int num_de_setores, void * buffer);
```

- Executa o acesso de disco através do BIOS (INT 13H)
cmd = 0 - inicializa o disco
 - 1 - retorna o status da última operação
 - 2 - lê disco
 - 3 - escreve no disco
 - 4 - verifica um ou mais setores
 - 5 - formata uma trilha
- Retorno → 0 indica sucesso

bioequip

```
int bioequip (void);
```

- Retorna quais equipamentos estão conectados no computador (INT 11H)

bioskey

```
int bioskey (int cmd);
```

- Leitura de teclado (INT 16H)
cmd = 0 - retorna o valor lido do teclado
 - 1 - retorna 0 indica que não existe tecla a ser lida
 - 2 - retorna um bit para cada tecla de controle

biosmemory

int biosmemory (void);

- Retorna quantidade de memória total em kbytes (INT 12H)

biosprint

int biosprint (int cmd, int byte, int port);

- Executa uma saída para a impressora (INT 17H)
- cmd = 0 - transmite byte
- 1 - inicializa o printer port
- 2 - lê o status do printer port

biostime

long biostime (int cmd, long nova_hora);

- Lê ou escreve no relógio (INT 1AH)
- cmd = 0 - lê
- 1 - escreve

brk

*int brk (void * addr);*

- Altera a alocação do espaço de data-segment
- Retorno → 0 indica sucesso, -1 indica erro

bsearch

```
void * bsearch (void * chave, void * inicio, long num_de_elementos, long tamanho_elem, int(* fcmp)(void *, void *));
```

- Faz uma busca binária com o algoritmo definido pelo retorno da função fcmp() que deverá retornar 0 ao encontrar uma coincidência. Os parâmetros para fcmp() são fornecidos automaticamente por bsearch

cabs

```
double cabs (estrutura);
```

- Retorna o valor absoluto de um número complexo
- Estrutura previamente definida → "complex"
- Retorno → HUGE_VAL indica overflow

calloc

```
void * calloc (long num_de_elementos , long tamanho_elem);
```

- Aloca memória principal

ceil

double ceil (double x);

- Arredonda x para o menor inteiro não menor que x

cgets

*char * cgets (char * string);*

- Lê uma string do teclado, string[0] deve conter o máximo número de elementos a serem lidos, e string[1] retornará o número de elementos lidos
- Retorno → pointer para string[2] (sempre)

chdir

*int chdir (char * path);*

- Altera o diretório corrente
- Retorno → 0 indica sucesso, -1 indica erro

_chmod

*int _chmod (char * path, int mode[, int atributo]);*

- Retorna atributo do arquivo se mode = 0
- Altera atributo do arquivo se mode = 1

chmod

*int chmod (char * path, int mode);*

- Altera o modo de acesso de um arquivo
- Retorno → 0 indica sucesso, -1 indica erro

chsize

int chsize (int handle, long novo_tamanho);

- Altera o tamanho do arquivo
- Retorno → 0 indica sucesso, -1 indica erro

circle

void far circle (int x, int y, int raio);

- Desenha um círculo com centro em x,y

_clear87

unsigned int _clear87 (void);

- Limpa o status de ponto flutuante

cleardevice

void far cleardevice (void);

- Limpa a tela gráfica

clearerr

*void clearerr (FILE * stream);*

- Limpa a indicação de erro relativo a stream

clearviewport

void far clearviewport (void);

- Limpa a janela gráfica corrente

clock

long clock (void);

- Retorna o número de clocks desde o início do programa
pode ser transformado em segundos, dividindo o valor
de retorno por 18.2

_close

int _close (int handle);

- Fecha o arquivo
- Retorno → 0 indica sucesso, -1 indica erro

close

int close (int handle);

- Fecha o arquivo
- Retorno → 0 indica sucesso, -1 indica erro

closegraph

void far closegraph (void);

- Encerra as operações com o sistema gráfico

clreol

void clreol (void);

- Limpa do cursor ao final da linha na atual janela de textos

clrscr

void clrscr (void);

- Limpa a atual janela de textos

_control87

*unsigned int _control87 (unsigned int novo,
 unsigned int mascara);*

- Altera a palavra de controle de ponto flutuante
- Retorno → nova palavra de controle

coreleft

unsigned [long] coreleft (void);

- Retorna o montante de RAM não usada (a opção long é utilizada nos modos compact, large e huge)

cos

double cos (double x);

- Retorna o cosseno do ângulo x (em radianos)

cosh

double cosh (double x);

- Retorna o cosseno hiperbólico do ângulo x (em radianos)

country

estrutura country (int xcode, estrutura);

- Especifica formato de informações dependentes do país
- Estrutura previamente definida → "country"

cprintf

*int cprintf (char * format [, argumentos, ...]);*

- Escreve na atual janela de textos (diretamente na RAM de vídeo se a variável directvideo = 1 (default = 1))
- Retorno → número de caracteres escritos

cputs

*int cputs (char * string);*

- Escreve a string na atual janela de textos
- Retorno → último caracter escrito

_creat

*int _creat (char * path, int atributo);*

- Cria um novo arquivo ou escreve sobre ele caso já exista

creat

*int creat (char * path, int mode);*

- Cria um novo arquivo ou escreve sobre ele caso já exista

creatnew

*int creatnew (char *path, int atribuito);*

- Cria um novo arquivo ou o mantém intacto caso já exista

creattemp

*int creattemp (char * path, int atributo);*

- Cria um arquivo com um nome inexistente no diretório path
(path deve terminar com \)

cscanf

*int cscanf (char *format[, argumentos, ...]);*

- Lê entrada formatada do teclado
- Retorno → número de argumentos lidos

ctime

*char *ctime (long *time);*

- Converte data e hora para uma string

ctrlbrk

*void ctrlbrk (int (*função) (void));*

- Determina função a ser executada em um control break

delay

void delay (unsigned milisegundos);

- Espera

delline

void delline (void); → Elimina uma linha na janela de textos

- Elimina linha na atual janela de textos

detectgraph

*void far detectgraph (int far * graphdriver, int far * graphmode);*

- Determina o tipo placa gráfica que está sendo usada no momento

difftime

double difftime (long time2, long time1);

- Calcula a diferença entre dois tempos

disable

void disable (void);

- Desabilita as interrupções (Só NMI permanece habilitada)

div

estrutura div (int numerador, int denominador);

- Divide dois inteiros
- Estrutura previamente definida → "div_t"

dosexterr

int dosexterr (estrutura);

- Captura informação expandida de erro do DOS (função DOS 0x59)
- Retorno → 0 indica que não houve falha na última chamada uma função do DOS
- Estrutura previamente definida → "DOSERROR"

dostounix

long dostounix (estrutura date, estrutura time);

- Converte data e hora para formato UNIX
- Estrutura previamente definida → "date" e "time"

drawpoly

*void far drawpoly (int num_pontos, int far * polipontos);*

- Desenha um polígono

dup

```
int dup (int handle);
```

- Duplica o handle para que se possa ter um duplo acesso para o mesmo file

dup2

```
int dup2 (int oldhandle, int newhandle);
```

- Idem ao Dup, Exceto que o arquivo original será encerado

ecvt

```
char * ecvt (double valor, int num_de_digitos,  
             int * decimal_point_register, int * sinal);
```

- Converte número de ponto flutuante para um string

ellipse

```
void far ellipse (int x, int y, int ang_inicial, int ang_final,  
                  int raioX, int raioY);
```

- Desenha uma elipse com centro em x,y

emit

```
void __emit__ (argumento,...);
```

- Permite inserir valores literais diretamente no código objeto
que é gerado.

enable

```
void enable (void);
```

- Habilita todas interrupções

```
int eof (int handle);
```

- Retorno → 0 indica mais dados a serem lidos
-1 indica erro

```
int execl (char * path, char *arg0, ..., NULL);
```

- Executa programa externo
- Não retorna ao programa principal caso o programa externo seja executado

execle

```
int execle (char * path, char * arg0, ..., NULL,  
           char ** ambiente);
```

- Executa programa externo utilizando novo ambiente
- Não retorna ao programa principal caso o programa externo seja executado

execlp

```
int execlp (char * path, char *arg0, ..., NULL);
```

- Executa programa externo utilizando o DOS PATH para localizá-lo
- Não retorna ao programa principal caso o programa externo seja executado

execle

```
int execle (char * path, char * arg0, ..., NULL,  
           char ** ambiente);
```

- Executa programa externo utilizando DOS PATH e novo ambiente
- Não retorna ao programa principal caso o programa externo seja executado

execv

*int execv (char * path, char * argv[]);*

- Executa programa externo com argumentos passados por matriz de pointers
- Não retorna ao programa principal caso o programa externo seja executado

execve

*int execve (char * path, char * argv[], char ** ambiente);*

- Executa programa externo com argumentos passados por matriz de pointers utilizando o novo ambiente
- Não retorna ao programa principal caso o programa externo seja executado

execvp

*int execvp (char * path, char * argv[]);*

- Executa programa externo com argumentos passados por matriz de pointers utilizando o DOS PATH para localizar o programa
- Não retorna ao programa principal caso o programa externo seja executado

execvpe

```
int execvpe (char * path, char * argv[], char ** ambiente);
```

- Executa programa externo com argumentos passados por matriz de pointers utilizando o DOS PATH e novo ambiente
- Não retorna ao programa principal caso o programa externo seja executado

_exit

```
void _exit (int returncode);
```

- Abandona o programa sem fechar arquivos, nem chamar funções definidas em atexit

exit

```
void exit (int returncode);
```

- Descarrega todos os buffers de saída, fecha todos arquivos, executa arquivos definidos em atexit e termina o programa

exp

```
double exp (double x);
```

- Retorna "e" elevado a x

fabs

função

double fabs (double x);

- Retorna o valor absoluto de um número de ponto flutuante

farcalloc

função

*void far * farcalloc (unsigned long nun_elem,
 unsigned long tamanho_elem);*

- Aloca memória do sistema
- Retorno → endereço inicial da memória alocada, 0 indica erro

farcoreleft

função

unsigned long farcoreleft (void);

- Retorna a quantidade de memória disponível em kbytes

farfree

função

*void farfree (void far * bloco);*

- Libera a RAM previamente alocada

farmalloc

*void far * farmalloc (unsigned long num_de_bytes);*

- Retorno → endereço inicial da memória alocada,
0 indica erro

farrealloc

*void far * farrealloc (void far * end_velho, unsigned long num_de_bytes);*

- Reajusta o tamanho de bloco alocado

fclose

*int fclose (FILE * stream);*

- Fecha arquivo
- Retorno → 0 indica sucesso, EOF indica erro

fcloseall

int fcloseall (void);

- Fecha todos arquivos
- Retorno → número de arquivos fechados, EOF indica erro

fcvt

```
char * fcvt (double valor, int num_de_dig,  
             int *numero_de_casas_dec, int * sinal);
```

- Converte um número de ponto flutuante para uma string

fdopen

```
FILE * fdopen (int handle, char * type);
```

- Transforma o handle em um stream, o handle permanece inalterado

feof

```
int feof (FILE * stream);
```

- Retorna não zero se o fim do arquivo foi alcançado

ferror

```
int ferror (FILE * stream);
```

- Detecta erro no arquivo
- Retorno → 0 indica sucesso, não zero indica erro

fflush

*int fflush (FILE * stream);*

- Descarrega o buffer do arquivo, sem no entretanto fechá-lo
- Retorno → 0 indica sucesso, EOF indica erro

fgetc

*int fgetc (FILE * stream);*

- Lê o próximo caractere do arquivo
- Retorno → EOF indica erro ou fim de arquivo

fgetchar

int fgetchar (void);

- Lê o próximo caractere do stdin
- Retorno → EOF indica erro ou fim de arquivo

fgetpos

*int fgetpos (FILE * stream, long * pos);*

- Retorna em pos a posição atual de leitura ou escrita no arquivo
- Retorno → 0 indica sucesso, não zero indica erro

fgets

*char * fgets (char * dest, int n, FILE * stream);*

- Lê string do arquivo (limitado em n-1 caracteres)
- Retorno → NULL indica erro ou fim de arquivo

filelength

long filelength (int handle);

- Retorna tamanho do arquivo

fileno

*int fileno (FILE * stream);*

- Transforma o stream em um handle, o stream permanece inalterado

fillellipse

void far fillellipse (int x, int y, int raioX, int raioY);

- Desenha e pinta elipse com centro em x,y

fillpoly

```
void far fillpoly (int numpoints, int far * polypoints);
```

- Desenha e pinta um polígono

findfirst

```
int findfirst (char * path, estrutura,int attrib);
```

- Inicia a procura num diretório
- Estrutura previamente definida → "ffblk"
- Retorno → 0 indica sucesso, -1 indica erro ou arquivo não encontrado

findnext

```
int findnext (estrutura);
```

- Continua a procura iniciada por findfirst
- Estrutura previamente definida → "ffblk"
- Retorno → 0 indica sucesso, -1 indica erro ou fim da lista de arquivos

floodfill

```
void far floodfill (int x, int y, int border);
```

- Preenche área fechada contendo x,y com atual cor e padrão

floor

double floor (double x);

- Retorna maior inteiro não maior que x

flushall

int flushall (void);

- Descarrega todos os buffers relacionados com streams
- Retorno → número de streams abertos

fmod

double fmod (double x, double y);

- Retorna o resto da divisão de x por y

fnmerge

*void fnmerge (char * path, char * drive, char * dir, char * nome,
char * ext);*

- Compõe um path a partir dos argumentos

fnsplit

```
int fnsplit (char * path, char * drive, char * dir, char * nome,  
            char * ext);
```

- Decompõe o path
- Retorno → flags de existência dos argumentos

fopen

```
FILE * fopen (char * path, char * mode);
```

- Abre arquivo path e o associa a um stream que é retornado

FP_OFF

```
unsigned FP_OFF (farpointer);
```

- Retorna offset do far pointer

_fpreset

```
void _fpreset (void);
```

- Reinicializa status de ponto flutuante

fprintf

*int fprintf (FILE * stream, char * format [, argumentos, ...]);*

- Copia formatadamente para arquivo
- Retorno → EOF indica erro

FP_SEG

unsigned FP_SEG (farpointer);

- Retorna segmento do far pointer

fputc

*int fputc (int c, FILE * stream);*

- Escreve o caracter c no arquivo
- Retorno → EOF indica erro

fputchar

int fputchar (int c);

- Escreve o caracter c em stdout
- Retorno → EOF indica erro

fputs

funcionalidade

```
int fputs (char * s, FILE * stream);
```

- Copia a string para o arquivo
- Retorno → EOF indica erro

fread

funcionalidade

```
long fread (void * buffer, long tamanho_item, long num_de_itens,
FILE * stream);
```

- Lê bloco do arquivo para buffer
- Retorno → num_de_itens indica sucesso

free

funcionalidade

```
void free (void * bloco);
```

- Libera bloco de memória principal previamente alocada

freemem

funcionalidade

```
int freemem (unsigned segx);
```

- Libera bloco de memória do DOS previamente alocada
- Retorno → 0 indica sucesso, -1 indica erro

freopen

*FILE * freopen (char * path, char * mode, FILE * stream);*

- Substitui o arquivo relacionado a stream por path. O arquivo previamente relacionado é incondicionalmente fechado

frexp

*double frexp (double x, int * expoente);*

- Decompõe x em mantissa e expoente (na base 2)
- Retorno → mantissa

fscanf

*int fscanf (FILE * stream, char * format [, argumentos, ...]);*

- Copia formatadamente do arquivo
- Retorno → número de argumentos lidos,
EOF indica fim de arquivo

fseek

*int fseek (FILE * stream, long num_de_bytes, int fromwhere);*

- Altera o pointer de leitura ou escrita do arquivo
- Retorno → 0 indica sucesso

fsetpos

*int fsetpos (FILE * stream, long * pos);*

- Altera o pointer de leitura ou escrita do arquivo
- Retorno → 0 indica sucesso

fstat

int fstat (int handle, estrutura);

- Captura informações do arquivo
- Estrutura previamente definida → "stat"

ftell

*long ftell (FILE * stream);*

- Retorna a posição atual de leitura ou escrita do arquivo
- Retorno → -1 indica erro

ftime

void ftime (estrutura);

- Armazena hora atual
- Estrutura previamente definida → "timeb"

fwrite

```
long fwrite (void * buffer, long tamanho_item, long num_de_itens,  
FILE * stream);
```

- Copia bloco a partir de buffer para o arquivo
- Retorno → num_de_itens indica sucesso

gcvt

```
char * gcvt (double x, int ndec, char * string);
```

- Converte x de ponto flutuante em string com ndec algarismos significativos

geninterrupt

```
void geninterrupt (int interrupt_num);
```

- Gera interrupção de software

getarccoords

```
void far getarccoords (estrutura);
```

- Captura as coordenadas da última chamada a arcMA -
- Estrutura previamente definida → "arccoordstype"

getaspectratio

```
void far getaspectratio (int far * xasp, int far * yasp);
```

- Captura a atual relação de aspecto da tela gráfica

getbkcolor

```
int far getbkcolor (void);
```

- Retorna atual cor de fundo no modo gráfico

getc

```
int getc (FILE * stream);
```

- Retorna caractere lido do arquivo

getcbrk

```
int getcbrk (void);
```

- Retorna atual estado de ctrl-break
- Retorno → 0 indica control break desativado

getch

int getch (void);

- Retorna caractere capturado do teclado sem apresentá-lo na tela
-

getchar

int getchar (void);

- Retorna caractere capturado de stdin
 - Retorno → EOF indica erro ou fim de arquivo
-

getche

int getche (void);

- Retorna caractere capturado do teclado mostrando-o na atual janela de textos
-

getcolor

int far getcolor (void);

- Retorna atual cor da tela gráfica

getcurdir

```
int getcurdir (int drive, char * diretório);
```

- Captura atual diretório de trabalho para drive especificado
- Retorno → 0 indica sucesso, -1 indica erro

getcwd

```
char * getcwd (char * diretório, int buflen);
```

- Captura atual diretório de trabalho limitado em buflen

getdate

```
void getdate (estrutura);
```

- Captura a data do sistema
- Estrutura previamente definida → "date"

getdefaultpalette

```
estrutura far getdefaultpalette ( void );
```

- Retorna tabela de cores inicializada em initgraph
- Estrutura previamente definida → "palettetype"

getdfree

void getdfree (unsigned char drive, estrutura);

- Captura espaço disponível no disco especificado por drive (0 para default, 1 para A:, ...)
- Estrutura previamente definida → "dfree"

getdisk

int getdisk (void);

- Retorna atual drive (0 para A:, 1 para B: ...)

getdrivername

*char * far getdrivername (void);*

- Retorna um pointer para o nome do atual driver gráfico

getdta

*char far * getdta (void);*

- Retorna far pointer para DTA atual

getenv

```
char * getenv (char * string);
```

- Captura o valor associado a variável de ambiente string

getfat

```
void getfat (unsigned char drive, estrutura);
```

- Captura informações do FAT para o drive especificado
(0 para default, 1 para A:, ...)
- Estrutura previamente definida → "fatinfo"

getfadt

```
void getfadt (estrutura);
```

- Captura informações do FAT para o drive atual
- Estrutura previamente definida → "fatinfo"

getfillpattern

```
void far getfillpattern (char far * padrão);
```

- Copia padrão de cores definidas pelo usuário para memória

getfillsettings

```
void far getfillsettings (estrutura);
```

- Captura informações sobre o atual padrão de preenchimento de figuras e suas cores
- Estrutura previamente definida → "fillsettingstype"

getftime

```
int getftime (int handle, estrutura);
```

- Captura data e hora do arquivo
- Estrutura previamente definida → "ftime"
- Retorno → 0 indica sucesso, -1 indica erro

getgraphmode

```
int far getgraphmode (void);
```

- Retorna o atual modo gráfico

getimage

```
void far getimage (int esquerdo, int topo, int direito, int base,  
                  void far * bitmap);
```

- Salva imagem formada de bits da tela gráfica em bitmap

getlinesettings

void far getlinesettings (estrutura);

- Captura estilo, espessura e padrão de linha
- Estrutura previamente definida → "linesettingstype"

getmaxcolor

int far getmaxcolor (void);

- Retorna o maior valor de cor permitido para setcolor

getmaxmode

int far getmaxmode (void);

- Retorna o maior valor de modo para o atual driver gráfico

getmaxx

int far getmaxx (void);

- Retorna maior valor de coordenada x para driver gráfico e modo atuais

getmaxy

int far getmaxy (void);

- Retorna maior valor de coordenada y para driver gráfico e modo atuais

getmodename

*char * far getmodename (int modo);*

- Retorna pointer para string que contém nome do modo gráfico especificado

getmoderange

*void far getmoderange (int driver, int far * min, int far * max);*

- Captura valores mínimo e máximo para driver gráfico

getpalette

void far getpalette (estrutura);

- Captura informação sobre tabela atual de cores
- Estrutura previamente definida → "palettetype"

getpalettesize

int far getpalettesize (void);

- Retorna o número de cores da tabela atual

getpass

*char * getpass (char * mensagem);*

- Lê senha do console após apresentar mensagem

getpid

unsigned getpid (void);

- Retorna identificador de programa (code-segment de psp)

getpixel

unsigned far getpixel (int x, int y);

- Retorna cor do ponto de coordenada x,y da tela gráfica

getpsp

unsigned getpsp (void);

- Retorna o endereço do code-segment de psp
(função DOS 62H)

gets

*char * gets (char * dest);*

- Captura string de stdin (converte CR em \0)

gettext

*int gettext (int esquerdo, int topo, int direito, int base,
void * dest);*

- Copia texto da tela de textos para memória
- Retorno → 1 indica sucesso, 0 indica erro

gettextinfo

void gettextinfo (estrutura);

- Captura informações da atual janela de texto
- Estrutura previamente definida → "text_info"

gettextsettings

```
void far gettextsettings (estrutura);
```

- Captura informações sobre atual fonte de texto no modo gráfico
- Estrutura previamente definida → "textsettingstype"

gettime

```
void gettime (estrutura);
```

- Captura a hora do sistema
- Estrutura previamente definida → "time"

getvect

```
void interrupt (* getvect (int interrupt_num))();
```

- Retorna um interrupt pointer para o endereço de tratamento da interrupção interrupt_num

getverify

```
int getverify (void);
```

- Retorna o atual estado de verify (0- desativado, 1- ativado)

getviewsettings

void far getviewsettings (estrutura);

- Captura informações sobre atual port gráfico
- Estrutura previamente definida → "viewporttype"

getw

*int getw (FILE * stream);*

- Retorna inteiro lido do arquivo

getx

int far getx (void);

- Retorna coordenada x na atual janela gráfica

gety

int far gety (void);

- Retorna coordenada y na atual janela gráfica

gmtime

*void * far gmtime (long * timer);*

- Converte data e hora para GMT (Greenwich Mean Time)
- Estrutura previamente definida → "tm"

gotoxy

void gotoxy (int x, int y);

- Posiciona o cursor na atual janela de texto

graphdefaults

void far graphdefaults (void);

- Inicializa os parâmetros gráficos

grapherrmsg

*char * far grapherrmsg (int errorcode);*

- Retorna pointer para mensagem relativa a errorcode gerado por graphresult

_graphfreemem

*void far _graphfreemem (void far * ptr, unsigned tamanho);*

- Libera memória gráfica previamente alocada por graphgetmem

_graphgetmem

*void far * far _graphgetmem (unsigned tamanho);*

- Aloca memória gráfica

graphresult

int far graphresult (void);

- Retorna o código do último erro gráfico ocorrido

harderr

*void harderr (int(* função)());*

- Define função para tratamento de erro de hardware

hardresume

void hardresume (int axret);

- Função de retorno após tratamento de erro de hardware

hardretn

void hardretn (int retn);

- Função de retorno após tratamento de erro de hardware

highvideo

void highvideo (void);

- Seleciona alta-intensidade da atual cor dos caracteres

hypot

double hypot (double x, double y);

- Calcula hipotenusa do triângulo retângulo de catetos x e y
- Retorno → **HUGE_VAL** indica erro

imagesize

comparamos

unsigned far imagesize (int esquerdo, int topo, int direito, int base);

- Retorna o número de bytes necessários para armazenar imagem formada de bits limitada em 64k

initgraph

disponível

*void far initgraph (int far * graphdriver, int far * graphmode, char far * pathtodriver);*

- Inicializa sistema gráfico

import

comparamos

int import (int port_de_entrada);

- Retorna valor da palavra lida (16 bits)

inportb

logon

unsigned char inportb (int port_de_entrada);

- Retorna valor do byte lido

insline

void insline (void);

- Insere linha vazia na atual janela de texto

installuserdriver

*int far installuserdriver (char far * nome, int huge(* detect)(void));*

- Instala driver de vídeo do usuário
- Retorno → número do driver a ser utilizado no initgraph

installuserfont

*int far installuserfont (char far * nome);*

- Instala fonte de caracteres de vídeo do usuário
- Retorno → número da fonte a ser utilizado no settextstyle

int86

*int int86 (int intno, union REGS * inregs, union REGS * outregs);*

- Executa interrupção do 8086 especificada por intno
- Retorno → dosAX

int86x

```
int int86x (int intno, union REGS * inregs, union REGS * outregs  
estrutura);
```

- Executa interrupção do 8086 especificada por intno
- Estrutura previamente definida → "SREGS"
- Retorno → dosAX

intdos

```
int intdos (union REGS * inregs, union REGS * outregs);
```

- Executa interrupção DOS 21H
- inregs → h.al especifica a função invocada
- Retorno → dosAX

intdosx

```
int intdosx (union REGS * inregs, union REGS * outregs,  
estrutura);
```

- Executa interrupção DOS 21H
- inregs → h.al especifica a função invocada
- Estrutura previamente definida → "SREGS"
- Retorno → dosAX

intr

void intr (int num_de_interrupção, estrutura);

- Alternativa para execução de interrupção do 8086
- Estrutura previamente definida → "REGPACK"

ioctl

*int ioctl (int handle, int modo [, void * argdx, void * argcx]);*

- Controla dispositivo de entrada e saída (função DOS 44H)

isalnum

int isalnum (int c)

- Retorna não zero para caracteres alfanuméricos
(A-Z, a-z, 0-9)

isalpha

int isalpha (int c)

- Retorna não zero para caracteres alfabéticos (A-Z, a-z)

isascii

int isascii (int c);

- Retorna não zero para caracteres ASCII (0x00 - 0x7F)

isatty

int isatty (int handle);

- Retorna não zero se handle estiver associado com um dos dispositivos : terminal, console, printer, serial port

iscntrl

int iscntrl (int c);

- Retorna não zero para caracteres : 0x00 - 0x1F, ou 0x7F

isdigit

int isdigit (int c);

- Retorna não zero para caracteres numéricos decimais (0 - 9)

isgraph

int isgraph (int c);

- Retorna não zero para caracteres entre 0x21 e 0x7E

islower

int islower (int c);

- Retorna não zero para caracteres alfabéticos minúsculos (a-z)

isprint

int isprint (int c);

- Retorna não zero para caracteres entre 0x20 e 0x7E

ispunct

int ispunct (int c);

- Retorna não zero para caracteres : 0x00 - 0x20 ou 0x7F

isspace

int isspace (int c);

- Retorna não zero para os caracteres : espaço, tab, CR, newline, vertical tab ou form feed (0x09 - 0x0D, ou 0x20)

isupper

int isupper (int c);

- Retorna não zero para caracteres alfabéticos maiúsculos (A-Z)

isxdigit

int isxdigit (int c);

- Retorna não zero para caracteres numéricos hexadecimais (0 - 9, A - F, a - f)

itoa

*char * itoa (int valor, char * dest, int base);*

- Converte valor em uma string utilizando base

kbhit

int kbhit (void);

- Retorno → 0 indica que não há tecla disponível

keep

void keep (unsigned char returncode, unsigned size);

- Termina programa mas o mantém residente
(função DOS 31H)

labs

long labs (long x);

- Retorna o valor absoluto de long x

Idexp

double Idexp (double x, int expoente);

- Retorna x vezes 2 elevado a expoente

Idiv

estrutura Idiv (long numerador, long denominador);

- Executa divisão
- Utiliza estrutura definida anteriormente
- Estrutura previamente definida → "Idiv_t"

Ifind

void * Ifind (void * chave, void * inicio, long * num_de_elem,
long tamanho_elem, int(*fcmp)(void *, void *));

- Faz uma busca linear com o algoritmo definido pelo retorno da função fcmp() que deverá retornar 0 ao encontrar uma coincidência. Os parametros para fcmp() são fornecidos automaticamente por Ifind

line

void far line (int x1, int y1, int x2, int y2);

- Desenha linha entre os pontos de coordenadas x1,y1 e x2,y2

linerel

void far linerel (int dx, int dy);

- Desenha linha entre o cursor e o ponto de distância relativa

lineto

Diagrama

```
void far lineto (int x, int y);
```

- Desenha linha entre o cursor e o ponto de coordenada x,y

localtime

Diagrama

```
estrutura Iccalctime (long * timer);
```

- Converte data e hora para a estrutura
- Estrutura previamente definida → "tm"

lock

Diagrama

```
int lock (int handle, long offset, long length);
```

- Especifica proteção contra abertura simultânea do arquivo na região especificada
- Retorno → 0 indica sucesso, -1 indica erro

log

Diagrama

```
double log (double x);
```

- Calcula o logaritmo neperiano de x (ln x)

log10

double log10 (double x);

- Calcula o logaritmo na base 10 de x (log x)

longjmp

void longjmp (estrutura, int valor_retorno);

- Executa goto não local restaurando status salvo na última chamada de setjmp com o mesmo argumento
- Estrutura previamente definida → "jmp_buf"

lowvideo

void lowvideo (void);

- Seleciona baixa-intensidade na atual cor dos caracteres

_lrotl

unsigned long _lrotl (unsigned long valor, int num);

- Retorna valor deslocado num bits para a esquerda

_lrotr

```
unsigned long _lrotr (unsigned long valor, int num);
```

- Retorna valor deslocado num bits para a direita

lsearch

```
void * lsearch (void * chave, void * inicio, long * num_de_elem,  
                long tamanho_elem, int( * fcmp)(void *, void *));
```

- Faz uma busca linear com o algoritmo definido pelo retorno da função fcmp() que deverá retornar 0 ao encontrar uma coincidência. Os parâmetros para fcmp() são fornecidos automaticamente por lsearch
- Em caso de não achado, a chave é adicionada na tabela

lseek

```
long lseek (int handle, long num_de_bytes, int fromwhere);
```

- Altera o pointer de leitura ou escrita do arquivo
- Retorno → -1L indica erro

ltoa

```
char * ltoa (long valor, char * dest, int base);
```

- Converte long valor para string dest utilizando base

malloc

```
void * malloc (long size);
```

- Aloca bloco de tamanho size na memória principal

_matherr

```
double _matherr (_mexcep why, char * fun, double * arg1p,  
                 double * arg2p, double ret);
```

- Manuseia erros de ponto flutuante

matherr

```
int matherr (estrutura);
```

- Manuseia erros gerados pela biblioteca math
- Estrutura previamente definida → "exception"

max

```
type max (type a, type b);
```

- Compara dois valores do mesmo tipo, retornando o maior

memccpy

```
void * memccpy (void * dest, void * origem, int c,  
                long num_de_bytes);
```

- Copia bloco até que c seja copiado limitado em num_de_bytes
-

memchr

```
void * memchr (void * string, int c, long num_de_bytes);
```

- Procura c nos primeiros num_de_bytes da string
-

memcmp

```
int memcmp (void * string1, void * string2, long num_de_bytes);
```

- Compara primeiros num_de_bytes de string1 com string2
-

memcpy

```
void * memcpy (void * dest, void * origem, long num_de_bytes);
```

- Copia bloco de origem para dest. Em caso de sobreposição de origem com dest o resultado é indefinido

memicmp

*int memicmp (void * string1, void * string2, long num_de_bytes);*

- Compara bloco de string1 com string2 ignorando maiúsculo/minúsculo

memmove

*void * memmove (void * dest, void * origem, long num_de_bytes);*

- Copia bloco de origem para dest mesmo em caso de sobreposição de origem com dest a cópia é correta.

memset

*void * memset (void * dest, int c, long num_de_bytes);*

- Iguala o bloco dest ao caracter c

min

type min (type a, type b);

- Compara dois valores do mesmo tipo, retornando o menor

MK_FP

```
void far * MK_FP (unsigned seg, unsigned offset);
```

- Cria um far pointer com segmento e offset

mkdir

```
int mkdir (char * path);
```

- Cria novo diretório
- Retorno → 0 indica sucesso, -1 indica erro

mktemp

```
char * mktemp (char * máscara);
```

- Cria um nome de arquivo único utilizando máscara

modf

```
double modf (double x, double * parte_inteira);
```

- Salva inteiro de x e retorna a parte fracionária

MK_FP

```
void far * MK_FP (unsigned seg, unsigned offset);
```

- Cria um far pointer com segmento e offset

mkdir

```
int mkdir (char * path);
```

- Cria novo diretório
- Retorno → 0 indica sucesso, -1 indica erro

mktemp

```
char * mktemp (char * máscara);
```

- Cria um nome de arquivo único utilizando máscara

modf

```
double modf (double x, double * parte_inteira);
```

- Salva inteiro de x e retorna a parte fracionária

movedata

```
void movedata (unsigned srcseg, unsigned srcoff,  
               unsigned dstseg, unsigned dstoff, long num_de_bytes)
```

- Copia bloco do endereço srcseg:srcoff para dstseg:dstoff

moverel

```
void far moverel (int dx, int dy);
```

- Move o cursor dx pixels na direção x e dy pixels na direção y

movetext

```
int movetext (int esquerdo, int topo, int direito, int base,  
              int dest_esquerdo, int dest_topo);
```

- Copia retângulo de texto da tela para outro de mesmo tamanho
- Retorno → 0 indica erro

moveto

```
void far moveto (int x, int y);
```

- Move a posição do cursor para coordenada x,y na janela gráfica

movmem

início

```
void movmem (void * origem, void * dest,  
             unsigned num_de_bytes);
```

- Copia bloco de origem para dest, mesmo em caso de sobreposição de origem com dest a cópia é feita corretamente

normvideo

início

```
void normvideo (void);
```

- Restaura as cores selecionadas antes do início do programa

nosound

início

```
void nosound (void);
```

- Desativa som gerado pela função sound

_open

início

```
int _open (char * path, modo_de_compartilhamento);
```

- Abre arquivo para leitura ou escrita utilizando a variável global _fmode

open

*int open (char * path, int flags [, unsigned mode]);*

- Abre arquivo para leitura ou escrita

outport

void outport (int port_de_saída, int valor);

- Carrega o port_de_saída com a palavra valor (16 bits)

outportb

void outportb (int port_de_saída, unsigned char valor);

- Carrega o port_de_saída com o byte valor

outtext

*void far outtext (char far * texto);*

- Escreve texto na atual janela gráfica

outtextxy

*void far outtextxy (int x, int y, char far * texto);*

- Escreve texto na posição x,y da janela gráfica

parsfnm

*char * parsfnm (char * cmdline, estrutura int opt);*

- Carrega cmdline em FCB utilizando função DOS 29H
- Estrutura previamente definida → "fcb"

peek

int peek (unsigned segment, unsigned offset);

- Retorna a palavra (16 bits) armazenada no endereço segment:offset

peekb

char peekb (unsigned segment, unsigned offset);

- Retorna o byte armazenado no endereço segment:offset

perror

```
void perror (char * string);
```

- Escreve string seguida de mensagem de erro em stderr
- A mensagem depende da variável global errno
- sys_errlist definida em stdio.h possui as mensagens de erro

pieslice

```
void far pieslice (int x, int y, int ang_inicial, int ang_final, int raio);
```

- Desenha gráfico percentual circular (pizza) com centro em x,y

poke

```
void poke (unsigned segment, unsigned offset, int palavra);
```

- Armazena palavra (16 bits) na memória de endereço segment:offset

pokeb

```
void pokeb (unsigned segment, unsigned offset, char byte);
```

- Armazena byte na memória de endereço segment:offset

poly

```
double poly (double x, int grau, double coeficientes[]);
```

- Retorna o valor de um polinômio especificado pelos argumentos avaliado para o valor x

pow

```
double pow (double x, double y);
```

- Retorna x elevado a y
- Retorna → HUGE_VAL indica erro

pow10

```
double pow10 (int x);
```

- Retorna 10 elevado a x

printf

```
int printf (char * format [, argumentos, ...]);
```

- Copia formatadamente para stdout
- Retorno → número de bytes copiados, EOF indica erro

putc

*int putc (int c, FILE * stream);*

- Escreve o caracter c no arquivo especificado
- Retorno → c, EOF indica erro

putch

int putch (int c);

- Escreve o caracter c na atual janela de texto
- Retorno → c, EOF indica erro

putchar

int putchar (int c);

- Escreve o caracter c em stdout
- Retorno → c, EOF indica erro

putenv

*int putenv (char * string);*

- Adiciona string ao ambiente atual
- Retorno → 0 indica sucesso, -1 indica erro

putimage

```
void far putimage (int esquerdo, int topo, void far * bitmap,  
                   int opção);
```

- Copia imagem formada de bits armazenada em bitmap na tela gráfica utilizando opção

putpixel

```
void far putpixel (int x, int y, int cor);
```

- Desenha um ponto na tela gráfica

puts

```
int puts (char * string);
```

- Escreve string em stdout
- Retorno → EOF indica erro

puttext

```
int puttext (int esquerdo, int topo, int direito, int base,  
            void * origem);
```

- Copia texto da memória para tela de texto
- Retorno → 0 indica erro

putw

*int putw (int palavra, FILE * stream);*

- Escreve a palavra (16 bits) no arquivo especificado
- Retorno → EOF indica erro

qsort

*void qsort (void * base, long num_de_elem, long tamanho_elem,
int(* fcmp)(void *, void *));*

- Executa um sort utilizando algoritmo "quicksort"

raise

int raise (int sinal);

- Envia sinal ao programa principal. Se sinal não foi especificado pela função signal a ação default será executada
- Retorno → 0 indica sucesso

rand

int rand (void);

- Retorna número aleatório

randbrd

int randbrd (estrutura, int recnum);

- Executa leitura aleatória de blocos utilizando FCB
- Estrutura previamente definida → "fcb"

randbwr

int randbwr (estrutura, int recnum);

- Executa escrita aleatória de blocos utilizando FCB
- Estrutura previamente definida → "fcb"

random

int random (int num_máx);

- Retorna número aleatório entre 0 e num_máx - 1

randomize

void randomize (void);

- Inicializa gerador de número aleatório

read

```
int _read (int handle, void * dest, unsigned num_de_bytes);
```

- Lê num_de_bytes do arquivo

read

```
int read (int handle, void * dest, unsigned num_de_bytes);
```

- Lê num_de_bytes do arquivo
- Para arquivos abertos no modo texto remove CR.

realloc

```
void * realloc (void * bloco, long novo_tamanho);
```

- Reajusta tamanho da memória principal

rectangle

```
void far rectangle (int esquerdo, int topo, int direito,int base);
```

- Desenha retângulo na tela gráfica

registerbgidriver

*int registerbgidriver (void(* driver)(void));*

- Associa driver ao sistema gráfico
- Retorno → número negativo indica driver ou fonte inválido

registerbgifont

*int registerbgifont (void(* fonte)(void));*

- Informa ao sistema gráfico que fonte foi incluida durante o processo de link
- Retorno → número negativo indica driver ou fonte inválido

remove

*int remove (char * path);*

- Apaga arquivo
- Retorno → 0 indica sucesso, -1 indica erro

rename

*int rename (char * nomeantigo, char * nomenovo);*

- Altera nome de arquivo nome antigo para nome novo
- Retorno → 0 indica sucesso, -1 indica erro

restorecrtmode

void far restorecrtmode (void);

- Restaura a tela para o modo texto

rewind

*void rewind (FILE * stream);*

- Altera pointer de leitura ou escrita para o início do arquivo

rmdir

*int rmdir (char * path);*

- Apaga diretório se estiver vazio, não for o atual nem o raiz
- Retorno → 0 indica sucesso, -1 indica erro

_rotl

unsigned _rotl (unsigned valor, int num);

- Retorna valor deslocado num bits para a esquerda

_rotr

```
unsigned _rotr (unsigned valor, int num);
```

- Retorna valor deslocado num bits para a direita

sbrk

```
void * sbrk (int incremento);
```

- Altera a alocação do espaço de data-segment

scanf

```
int scanf (char * format [, argumentos, ...]);
```

- Copia formatadamente de stdin
- Retorno → número de argumentos copiados, EOF indica erro

searchpath

```
char * searchpath (char * arquivo);
```

- Procura arquivo no path especificado no DOS

sector

```
void far sector (int X, int Y, int ang_inicial, int ang_final, int raioX, int raioY);
```

- Desenha gráfico percentual elíptico (pizza em perspectiva) com centro em x,y

segread

```
void segread (estrutura);
```

- Armazena atuais valores dos registros de segmento
- Estrutura previamente definida → "SREGS"

setactivepage

```
void far setactivepage (int num_da_pagina);
```

- Seleciona página gráfica

setallpalette

```
void far setallpalette (estrutura);
```

- Altera tabela de cores gráficas
- Estrutura previamente definida → "palettetype"

setaspectratio

void far setaspectratio (int xasp, int yasp);

- Altera a relação de aspecto da tela gráfica

setbkcolor

void far setbkcolor (int cor);

- Seleciona cor de fundo no modo gráfico

setblock

int setblock (unsigned segment, unsigned novotamanho);

- Modifica tamanho do segmento de memória previamente alocada

setbuf

*void setbuf (FILE * stream, char * buffer);*

- Determina armazenamento temporário de entrada e saída em buffer

setcbrk

alterações

int setcbrk (int cmd);

- Com cmd = 0 desativa ctrl-break, cmd = 1 ativa ctrl-break
- Retorno → cmd

setcolor

alterações

void far setcolor (int cor);

- Seleciona cor no modo gráfico

setdate

alterações

void setdate (estrutura);

- Ajusta a data do sistema
- Estrutura previamente definida → "date"

setdisk

alterações

int setdisk (int drive);

- Seleciona drive para default (0 - A:, 1 - B:, ...)
- Retorno → número total de drives

setdta

void setdta (char far * dta); sets the data pointer to point to the memory block.

- Carrega o valor de Data Transfer Área

setfillpattern

```
void far setfillpattern (char far * padrão_definido, int cor);
```

- Seleciona padrão_definido para preenchimento de figuras e suas cores

setfillstyle

void far setfillstyle (int padrão, int cor);

- Seleciona padrão de preenchimento de figuras e suas cores

setftime

int setftime (int handle, estrutura);

- Altera data e hora do arquivo
 - Estrutura previamente definida → "ftime"
 - Retorno → 0 indica sucesso, -1 indica erro

setgraphbufsize

unsigned far setgraphbufsize (unsigned novo_tamanho);

- Altera tamanho do buffer gráfico interno (deve ser chamado antes de initgraph)
- Retorna tamanho anterior

setgraphmode

void far setgraphmode (int modo);

- Seleciona modo gráfico e apaga a tela

setjmp

int setjmp (estrutura);

- Salva status para próxima chamada a longjmp com mesmo argumento
- Estrutura previamente definida → "jmp_buf"
- Retorno → 0 indica primeira chamada

setlinestyle

void far setlinestyle (int estilo, unsigned padrão, int espessura,

- Seleciona estilo, espessura e padrão de linha no modo gráfico

setmem

```
void setmem (void * dest, unsigned comprimento, int modo  
char caracter);
```

- Carrega região de memória com caracter

setmode

```
int setmode (int handle, int modo);
```

- Altera o modo do arquivo para texto ou binário
- Retorno → 0 indica sucesso, -1 indica erro

setpalette

```
void far setpalette (int num_cor, int nova_cor);
```

- Altera num_cor da atual tabela de cores

setrgbpalette

```
void far setrgbpalette (int num_de_cor, int vermelho, int verde,  
int azul);
```

- Permite usuário definir cores para VGA ou IBM8514

settextjustify

void far settextjustify (int horiz, int vert);

- Ativa justificação de texto no modo gráfico

settextstyle

void far settextstyle (int fonte, int direção, int tamanho);

- Seleciona características de texto

settime

void settim (estrutura);

- Ajusta a hora do sistema
- Estrutura previamente definida → "time"

setusercharsize

void far setusercharsize (int multx, int divx, int multy, int divy);

- Ajusta fator de ampliação dos textos no modo gráfico

setvbuf

```
int setvbuf (FILE * stream, char * buffer, int tipo, long tamano_do_buffer);
```

- Determina entrada e saída bufferizada para stream
- Se char * buffer = NULL o sistema aloca tamano_do_buffer automaticamente
- Retorno → 0 indica sucesso

setvect

```
void setvect (int num_interrupção, void interrupt (* isr)());
```

- Altera valor do vetor de interrupção

setverify

```
void setverify (int cmd);
```

- Ativa (cmd = 1) ou desativa (cmd = 0) a opção verify do DOS

setviewport

```
void far setviewport (int esquerda, int topo, int direita, int base, int clip);
```

- Define nova janela gráfica

setvisualpage

void far setvisualpage (int página);

- Seleciona número de página gráfica

setwritemode

void far setwritemode (int modo);

- Seleciona modo de escrita de linha gráfica (0 = sobrescreve)

signal

void (signal (int sinal, void (* função)(int sinal
[,int subcode])))(int);*

- Especifica ação de manipulação de sinal

sin

double sin (double x);

- Retorna o seno de x (x em radianos)

sinh

double sinh (double x);

- Retorna o seno hiperbólico de x (x em radianos)

sleep

void sleep (unsigned segundos);

- Suspende execução durante o tempo especificado

sopen

*int sopen (char * path, int acesso,
 int modo_de_compartilhamento, int modo);*

- Abre arquivo para leitura ou escrita

sound

void sound (unsigned frequência);

- Emite som com a frequência especificada

spawnl

```
int spawnl (int modo, char * path, char * arg0, ..., NULL);
```

- Executa programa externo
- Retorna ao programa principal caso modo = P_WAIT

spawnle

```
int spawnle (int modo, char * path, char * arg0, ..., NULL,  
            char ** ambiente);
```

- Executa programa externo utilizando o novo ambiente
- Retorna ao programa principal caso modo = P_WAIT

spawnlp

```
int spawnlp (int modo, char * path, char * arg0, ..., NULL);
```

- Executa programa externo utilizando o DOS PATH
- Retorna ao programa principal caso modo = P_WAIT

spawnlpe

```
int spawnlpe (int modo, char * path, char * arg0, ..., NULL,  
             char ** ambiente);
```

- Executa programa externo utilizando DOS PATH e novo ambiente
- Retorna ao programa principal caso modo = P_WAIT

spawnv

*int spawnv (int modo, char * path, char * argv[]);*

- Executa programa externo com argumentos passados por matriz de pointers
- Retorna ao programa principal caso modo = P_WAIT

spawnve

*int spawnve (int modo, char * path, char * argv[], char ** env);*

- Executa programa externo com argumentos passados por matriz de pointers utilizando o novo ambiente
- Retorna ao programa principal caso modo = P_WAIT

spawnvp

*int spawnvp (int modo, char * path, char * argv[]);*

- Executa programa externo com argumentos passados por matriz de pointers utilizando o DOS PATH
- Retorna ao programa principal caso modo = P_WAIT

spawnvpe

*int spawnvpe (int modo, char * path, char * argv[], char ** env);*

- Executa programa externo com argumentos passados por matriz de pointers utilizando o DOS PATH e o novo ambiente
- Retorna ao programa principal caso modo = P_WAIT

sprintf

*int sprintf (char * dest, char * format [, argumentos, ...]);*

- Copia formatadamente para dest
- Retorno → número de bytes copiados, EOF indica erro

sqrt

double sqrt (double x);

- Retorna a raiz quadrada de x

strand

void strand (unsigned semente);

- Inicializa gerador de número aleatório

sscanf

```
int sscanf (char * dest, char * format [, argumentos, ...]);
```

- Copia formatadamente de dest
- Retorno → número de argumentos, EOF indica erro

stat

```
int stat (char * path, estrutura);
```

- Captura informações do arquivo
- Estrutura previamente definida → "stat"
- Retorno → 0 indica sucesso, -1 indica erro

_status87

```
unsigned int _status87 (void);
```

- Retorna o status de ponto flutuante

stime

```
int stime (long * num_de_segundos);
```

- Ajusta data e hora do sistema com num_de_segundos desde 00:00:00 de 1 de janeiro de 1970 (GMT)

stpcpy

*char * stpcpy (char * dest, char * origem);*

- Copia string

strcat

*char * strcat (char * dest, char * origem);*

- Adiciona string origem à dest

strchr

*char * strchr (char * string, int caracter);*

- Retorna pointer para primeira ocorrência de caracter em string

strcmp

*int strcmp (char * string1, char * string2);*

- Retorna < 0 se string1 < string2, 0 se string1 = string2 e > 0 se string1 > string2

strcmpi

*int strcmpi (char * string1, char * string2);*

- Compara string1 com string2 ignorando maiúsculo/minúsculo
- Retorna < 0 se string1 < string2, = 0 se string1 = string2 e 0 > se string1 > string2

strcpy

*char * strcpy (char * dest, char * origem);*

- Copia string origem para destino

strcspn

*long strcspn (char * string1, char * string2);*

- Retorna o comprimento inicial de string1 que consiste inteiramente de string2

strdup

*char * strdup (char * string);*

- Retorna pointer para cópia duplicada de string

_strerror

*char * _strerror (char * mensagem_de_erro);*

- Retorna pointer para mensagem_de_erro. Se mensagem_de_erro = null retorna pointer para última mensagem

strerror

*char * strerror (int num_erro);*

- Retorna pointer para mensagem de erro associada com num_erro

strcmp

*int strcmp (char * string1, char * string2);*

- Compara string1 com string2 ignorando maiúsculo/minúsculo
- Retorna < 0 se string1 < string2, = 0 se string1 = string2 e >0 se string1 > string2

strlen

*long strlen (char * string);*

- Retorna o comprimento de string

strlwr

*char * strlwr (char * string);*

- Converte string para minúsculos

strncat

*char * strncat (char * dest, char * origem, long num_de_bytes);*

- Adiciona no máximo num_de_bytes da origem para dest

strcmp

*int strcmp (char * string1, char * string2, long num_de_bytes);*

- Compara no máximo num_de_bytes de string1 com string2
- Retorna < 0 se string1 < string2, = 0 se string1 = string2 e >0 se string1 > string2

strncMPI

*int strncMPI (char * string1, char * string2, long num_de_bytes);*

- Compara no máximo num_de_bytes de string1 com string2 ignorando maiúsculo/minúsculo
- Retorna < 0 se string1 < string2, = 0 se string1 = string2 e >0 se string1 > string2

strncpy

*char * strncpy (char *dest, char * origem, long num_de_bytes)*

- Copia no máximo num_de_bytes

strnicmp

*int strnicmp (char * string1, char * string2, long num_de_bytes);*

- Compara no máximo num_de_bytes de string1 com string2 ignorando maiúsculo/minúsculo
- Retorna < 0 se string1 < string2, = 0 se string1 = string2 e >0 se string1 > string2

strnset

*char * strnset (char * string, int caracter, long num_de_bytes);*

- Iguala os primeiros num_de_bytes de string a caracter

strupr

*char *strupr (char * string1, char * string2);*

- Retorna pointer para primeira ocorrência de qualquer caracter de string2 em string1

strrchr

*char * strrchr (char * string, int caracter);*

- Retorna pointer para última ocorrência de caracter em string

strrev

*char * strrev (char * string);*

- Inverte os caracteres de string

strset

*char * strset (char * string, int caracter);*

- Iguala todos os caracteres de string a caracter

strspn

*long strspn (char * string1, char * string2);*

- Retorna comprimento inicial de string1 que contém inteiramente caracteres de string2

Funções de manipulação de strings

strstr

`char * strstr (char * string1, char * string2);`

- Retorna pointer para string2 em string1

strtod

`double strtod (char * string, char ** endptr);`

- Converte string para double
- Retorna → HUGE_VAL indica overflow

strtok

`char * strtok (char * string1, char * string2);`

- Procura em string1 por uma marca de string2

strtol

`long strtol (char * string, char ** endptr, int base);`

- Converte string para long utilizando base

strtoul

*unsigned long strtoul (char * string, char ** endptr, int base);*

- Converte string para unsigned long utilizando "base"

strupr

*char *strupr (char * string);*

- Converte letras minúsculas de string para maiúsculas

swab

*void swab (char * origem, char * dest, int num_de_bytes);*

- Copia num_de_bytes da string origem para dest, trocando as posições pares com as ímpares

system

*int system (char * comando);*

- Executa o comando do DOS (Ex.: dir, copy, type, command,...)
- Retorno → 0 indica sucesso, -1 indica erro

tan

double tan (double x);

- Retorna a tangente do ângulo x (em radianos)

tanh

double tanh (double x);

- Retorna a tangente hiperólica do ângulo x (em radianos)

tell

long tell (int handle);

- Retorna o valor atual do pointer de leitura ou escrita do arquivo

textattr

void textattr (int novo_attributo);

- Seleciona as cores dos caracteres e de fundo

textbackground

void textbackground (int nova_cor);

- Seleciona a cor de fundo

textcolor

void textcolor (int nova_cor);

- Seleciona a cor dos caracteres

textheight

*int far textheight (char far * texto);*

- Retorna altura (em pixels) do texto no modo gráfico utilizando tamanho e fonte

textmode

void textmode (int novo_modo);

- Seleciona o modo de texto específico novo_modo

textwidth

*int far textwidth (char far * texto);*

- Retorna largura (em pixels) do texto no modo gráfico utilizando tamanho e fonte

time

*long time (long * dest);*

- Captura hora atual em segundos desde 00:00:00 de 1 janeiro de 1970
- Retorno → novo valor armazenado em dest

tmpfile

*FILE * tmpfile (void);*

- Cria e abre um arquivo binário temporário que é removido assim que for fechado ou o programa finalizado

tmpnam

*char * tmpnam (char * string);*

- Cria um nome de arquivo único

:toascii

int toascii (int caracter);

- Retorna caractere convertido para ASCII

tolower

int _tolower (int caracter);

- Macro que retorna caractere convertido para minúsculo
(só deve ser usado para caracteres maiúsculos)

tolower

int tolower (int caracter);

- Retorna caractere convertido para minúsculo

_toupper

int _toupper (int caracter);

- Macro que retorna caractere convertido para maiúsculo
(só deve ser usado para caracteres minúsculos)

toupper

int toupper (int caracter);

- Retorna caracter convertido para maiúsculo

tzset

void tzset (void);

- Seleciona as variáveis globais : daylight, timezone e tzname baseadas na variável de ambiente TZ

ultoa

*char * ultoa (unsigned long valor, char * string, int base);*

- Converte valor em uma string utilizando base

ungetc

*int ungetc (int caracter, FILE * stream);*

- Devolve caracter para stream
- Retorno → EOF indica falha

ungetch

int ungetch (int caracter);

- Devolve caracter para o teclado
- Retorno → EOF indica falha

unixtodos

void unixtodos (long hora_unix, estrutura date, estrutura time);

- Converte data e hora no formato UNIX em hora_unix
- Estrutura previamente definida → "date" e "time"

unlink

*int unlink (char * path);*

- Apaga arquivo
- Retorno → 0 indica sucesso, -1 indica erro

unlock

int unlock (int handle, long offset, long tamanho);

- Elimina proteção de abertura simultânea do arquivo (ver lock)

va_start

```
void va_start (va_list ap, lastfix);
```

- Inicializa passagem de lista argumentos com quantidade desconhecida

va_arg

```
type va_arg (va_list ap, type);
```

- Executa passagem de lista argumentos com quantidade desconhecida

va_end

```
void va_end (va_list ap);
```

- Finaliza passagem de lista argumentos com quantidade desconhecida

vfprintf

```
int vfprintf (FILE * stream, char * format, va_list arglist);
```

- Copia formatadamente para arquivo com argumentos arglist
- Retorno → número de bytes copiados, EOF indica erro

vfscanf

*int vfscanf (FILE * stream, char * format, va_list arglist);*

- Copia formatadamente de arquivo com argumentos arglist
- Retorno → número de argumentos copiados, EOF indica fim de arquivo

vprintf

*int vprintf (char * format, va_list arglist);*

- Copia formatadamente para "stdout" com argumentos arglist
- Retorno → numero de bytes copiados, EOF indica erro

vscanf

*int vscanf (char * format, va_list arglist);*

- Copia formatadamente de "stdin" com argumentos arglist
- Retorno → número de argumentos copiados, EOF indica fim de arquivo

vsprintf

*int vsprintf (char * buffer, char * format, va_list arglist);*

- Copia formatadamente para buffer com os argumentos arglist
- Retorno → número de bytes copiados, EOF indica erro

vsscanf

*int vsscanf (char * buffer, char * format, va_list arglist);*

- Copia formatadamente de buffer com os argumentos arglist
- Retorno → número de argumentos copiados, EOF indica fim de string

wherex

int wherex (void);

- Retorna posição horizontal do cursor na atual janela de texto

wherey

int wherey (void);

- Retorna posição vertical do cursor na atual janela de texto

window

void window (int esquerdo, int topo, int direito, int base);

- Define janela de textos

_write

*int _write (int handle, void * buffer, unsigned num_de_bytes);*

- Escreve num_de_bytes (< 65535) de buffer para o arquivo
- Não converte LF em CR/LF pois só usa arquivos binários
- Retorno → número de bytes escritos, -1 indica erro

write

*int write (int handle, void * buffer, unsigned num_de_bytes);*

- Escreve num_de_bytes de buffer para o arquivo
- Converte LF em CR/LF em arquivos texto
- Retorno → número de bytes escritos, -1 indica erro

VARIÁVEIS GLOBAIS

_8087

int _8087

- Flag do chip de coprocessador (não zero se existe)

_argc

int _argc

- Contém o número de argumentos da linha de comando

argv

*char * argv[]*

- Matriz de pointers para argumentos da linha de comando

_ctype

char ctype []

- Matriz de caracteres com informação de atributo

directvideo

int directvideo

- Flag de controle de saída de vídeo

environ

*char * environ[]*

- Matriz de pointers para strings utilizadas para verificar ou alterar variáveis de ambiente do DOS

errno

int errno

- Indica o tipo de erro quando ocorre falha
(utilizado por perror)

_doserrno

int _doserrno

- Indica o tipo de erro quando ocorre falha numa chamada ao DOS
(não utilizado por perror)

sys_errlist

*char * sys_errlist[]*

- Matriz com mensagens erro (utilizado por perror)

sys_nerr

int sys_nerr

- Variável com o número de mensagens de erro contidas em sys_errlist

_fmode

int _fmode

- Determina modo de abertura de arquivos (texto ou binário)

_heaplen

unsigned _heaplen

- Indica o tamanho de "near heap" (apenas no modos tiny small e medium)

_openfd

unsigned int _openfd[]

- Matriz de modos de acesso a arquivos e dispositivos

_osmajor

unsigned char _osmajor

- Contém inteiros relativos a versão do DOS

_osminor

unsigned char _osminor

- Contém fracionários relativos a versão do DOS

_psp

unsigned int _psp

- Contém o endereço de code-segment do psp

_stklen

unsigned _stklen

- Especifica o tamanho do stack (no mínimo 128 palavras - default = 4K bytes)

timezone

long timezone

- Contém diferença em segundos entre hora local e GMT

tzname

*char * tzname [2]*

- Matriz de pointers para nomes de "time zone"

_version

unsigned int _version

- Contém a versão do DOS (completa)

**TABELA DE CORES DEFINIDAS EM CONIO.H E
GRAPHICS.H**

CGA	NOME	EGA/VGA	NOME
0	BLACK	0	EGA_BLACK
1	BLUE	1	EGA_BLUE
2	GREEN	2	EGA_GREEN
3	CYAN	3	EGA_CYAN
4	RED	4	EGA_RED
5	MAGENTA	5	EGA_MAGENTA
6	BROWN	7	EGA_LIGHTGRAY
7	LIGHTGRAY	20	EGA_BROWN
8	DARKGRAY	56	EGA_DARKGRAY
9	LIGHTBLUE	57	EGA_LIGHTBLUE
10	LIGHTGREEN	58	EGA_LIGHTGREEN
11	LIGHTCYAN	59	EGA_LIGHTCYAN
12	LIGHTRED	60	EGA_LIGHTRED
13	LIGHTMAGENTA	61	EGA_LIGHTMAGENTA
14	YELLOW	62	EGA_YELLOW
15	WHITE	63	EGA_WHITE

BLINK - deve ser adicionado a cor dos caracteres para ativar modo piscante

Os streams definidos em STDIO.H abertos automaticamente no início do programa são :

stdin	dispositivo de entrada padrão
stdout	dispositivo de saída padrão
stderr	dispositivo de saída de erro padrão
stdaux	dispositivo auxiliar padrão
stdprn	impressora padrão

Os especificadores de formato são :

% [flags] [tam] [.prec] [modific] type

[flag]	Função
nenhum	alinhamento à direita , colocando 0 ou espaços à esquerda
-	alinhamento à esquerda, colocando espaços à direita
+	sempre inicia com + ou -
espaço	cópia sinal somente para valores negativos
#	converte utilizando forma alternativa : c,s,d,i,u sem efeito
0	0 precedido ao arg não zero
x or X	0x or 0X precedido ao arg
e, E, f	sempre usa ponto decimal
g or G	mesmo que L ou E sem zeros iniciais

[tam]	Efeito na saída
n	em printf = número mínimo de caracteres a copiar em scanf = número máximo de caracteres a ler

0n	no mínimo n caracteres com preenchimento de 0 a esquerda
*	próximo argumento é um tamanho

[.prec] **Efeito na saída (apenas com printf)**

nenhum	precisão default
.0	d,i,o,u,x precisão default e, E, f sem ponto decimal
.n	limita em n caracteres
*	próximo argumento é um tamanho

[Modific] **Como arg é interpretado**

F	arg é far pointer
N	arg é near pointer
h	d,i,o,u,x,X arg é short int
l	d,i,o,u,x,X arg é long int
l	e, E, f, g, G arg é double (scanf only)
L	e,E,f,g,G arg é long double

[type] **Formato de Saída**

d	signed decimal int
i	signed decimal int
o	unsigned octal int
u	unsigned decimal int
x	em printf = unsigned hexdecimal int minúsculo

	em scanf = hexadecimal int
X	em printf = unsigned hexadecimal int maiúsculo
	em scanf = hexadecimal long
f	ponto flutuante [-]ddddddd
e	ponto flutuante [-]d.ddd e [+/-]ddd
g	format e ou f dependendo da precisão
E	mesmo que e com E para expoente
G	mesmo que g com E para expoente
c	um caractere
s	copia caracteres até '\0' ou [.prec]
%	caractere %
p	pointer: near - YYYY; far - XXXX:YYYY
n	número de caracteres copiados, na posição indicada pelo argumento

AGRUPAMENTO DAS FUNÇÕES POR HEADER

ALLOC.H

brk	calloc	coreleft	
free	malloc	realloc	H.BRKT.O
sbrk	farcalloc	farcoreleft	
farfree	farmalloc	farrealloc	

ASSERT.H

assert

BIOS.H

bioscom	biosdisk	bioequip
bioskey	biosmemory	biosprint
biostime		

CONIO.H

clreol	clrscr	delline	
gettext	gettextinfo	gotoxy	
highvideo	insline	lowvideo	
movetext	normvideo	puttext	
textattr	textbackground	textcolor	

textmode	wherex	wherey	getchar
window	cgets	cprintf	getchar
cputs	cscanf	getch	kbhit
getche	getpass	kbhit	kbhit
putch	ungetch	kbhit	kbhit
CTYPE.H			
isalnum	isalpha	isascii	isgraph
iscntrl	isdigit	isgraph	isprint
islower	isprint	ispunct	ispunct
isspace	isupper	isxdigit	isxdigit
tolower	toupper	toascii	tolower
_tolower	_toupper	_tolower	_tolower
DIR.H			
chdir	findfirst	findnext	getcurdir
fnmerge	fnsplit	getcurdir	getcurdir
getcwd	getdisk	mkdir	mktemp
mktemp	rmdir	searchpath	setdisk
setdisk	setpath	setpath	setpath
	setpath	setpath	setpath

DOS.H

absread	abswrite	allocmem
bdos	bdosptr	country
ctrlbrk	delay	disable
dosexterr	dostounix	_emit_
enable	freemem	getcbrk
getdate	getdfree	getfat
getfadt	getpsp	getswitchchar
gettime	interrupt	getverify
harderr	hardresume	hardretn
inport	inportb	int86
int86x	intdos	intdosx
intr	keep	nosound
outport	outportb	parsfnm
peek	peekb	poke
pokeb	randbrd	randbwr
segread	setblock	setcbrk
setdate	settime	setvect
setverify	sleep	sound
unixtodos	unlink	getdta
setdta	MK_FP	FP_OFF
FP_SEG	geninterrupt	getcwd
getftime	getvect	inp
outp	outp	outp

FLOAT.H		H.800	
_clear87	_control87	_freset	_freload
_status87	_set87	_vsyst	_righto
GRAPHICS.H		_graphinit	_graphexit
arc	bar	bar3d	_fillimg
circle	cleardevice	clearviewport	_rotimg
closegraph	detectgraph	drawpoly	_animap
ellipse	filleepse	fillpoly	_transl
floodfill	getarccoords	getaspectratio	_transr
getbkcolor	getcolor	getdefautpalette	_color
getdrivername	getfillpattern	getfillsettings	_img
getgraphmode	getimage	getlinesettings	_stroke
getmaxcolor	getmaxmode	getmaxx	_size3d
getmaxy	getmodename	getmoderange	_device
getpixel	getpalette	getpalettesize	_imgsize
gettextsettings	getviewsettings	getx	_planar
gety	graphdefaults	grapherrmsg	_device2
_graphfreemem	_graphgetmem	graphresult	_imgsize2
imagesize	initgraph	installuserdriver	_device3
installuserfont	line	linerel	_imgsize3
lineto	moverel	moveto	_imgsize4
outtext	outtextxy	pieslice	_color2

putimage	putpixel	rectangle
restorecrtmode	sector	setactivepage
setallpalette	setaspectratio	setbkcolor
setcolor	setfillpattern	setfillstyle
setgraphbufsize	setgraphmode	setlinestyle
setpalette	setrgbpalette	settextjustify
settextstyle	setusercharsize	setviewport
setvisualpage	setwritemode	textheight
textwidth	registerbgidriver	registerfarbgidriver
registerbgifont		

IO.H

access	_chmod	chmod
chsize	_close	close
_creat	creat	creatnew
creattemp	dup	dup2
eof	filelength	getftime
ioctl	isatty	lock
lseek	_open	open
_read	read	setftime
setmode	tell	umask
unlink	unlock	_write
write	sopen	

	Standard	Extended	Extended
MATH.H			
abs	abs	abs	abs
atan	atan2	atan2	atan2
ceil	cos	cosh	cosh
exp	fabs	floor	exp
fmod	frexp	labs	expm1
ldexp	log	log10	expm1
modf	pow	sin	expm1
sinh	sqrt	tan	expm1
tanh	matherr	hypot	expm1
pow10	_matherr	cabs	expm1
poly	besselj0	besselj0	expm1
	besselj0	besselj0	expm1
MEM.H			
memccpy	memchr	memcmp	memchr
memcpy	memicmp	memmove	memchr
memset	movedata	movmem	memchr
setmem	memmove	memset	memchr
	memmove	memset	memchr
PROCESS.H			
abort	execl	execle	execle
execlp	execlepe	execv	execle

execve	execvp	execvpe	execvpe
exit	_exit	spawnl	spawnl
spawnle	spawnlp	spawnlpe	spawnlpe
spawnv	spawnve	spawnvp	spawnvp
spawnvpe	system	spawnvpe	system
SETJMP.H			
longjmp	setjmp	longjmp	longjmp
SIGNAL.H			
raise	signal	sigdel	sigdel
STDIO.H			
clearerr	fclose	fflush	fflush
fgetc	fgetpos	fgets	fgets
fopen	fprintf	fputc	fputc
fputs	fread	freopen	freopen
fscanf	fseek	fsetpos	fsetpos
ftell	fwrite	gets	gets
perror	printf	puts	puts
rename	rewind	scanf	scanf
setbuf	setvbuf	sprintf	sprintf
sscanf	strerror	tmpfile	tmpfile
tmpnam	ungetc	vfprintf	vfprintf
vfscanf	vprintf	vscanf	vscanf

vsprintf	vsscanf	fcloseall	_ioblock
fdopen	fgetchar	flushall	_ioblock
fputchar	getw	putw	_ioblock
_strerror	unlink	getc	_ioblock
putc	feof	ferror	_ioblock
fileno	_strerror	getcchar	
putchar	remove		MMFILE.C

STDLIB.H

abort	abs	atexit	_ioblock
atof	atoi	atol	
bsearch	calloc	div	MMFILE.C
exit	free	getenv	_ioblock
labs	ldiv	malloc	
qsort	rand	realloc	
srand	strtod	strtol	
strtoul	system	ecvt	
_exit	fcvt	gcvt	
itoa	lfind	_lrotl	
_lrotr	lsearch	ltoa	
putenv	_rotl	_rotr	
swab	ultoa	max	
min	random	randomize	

STRING.H

memccpy	memchr	memcmp
memcpy	memicmp	memmove
memset	movedata	stpcpy
strcat	strchr	strcmp
strcpy	strcspn	strupr
strerror	strcmp	strlen
strlwr	strncat	strncmp
strncpy	strnicmp	strnset
strpbrk	strrchr	strrev
strset	strspn	strstr
strtok	strupr	movemem
setmem	stremp	_strerror
strnampi		

SYS \ STAT.H

fstat	stat
-------	------

SYS \ TIMEB.H

ftime

TIME.H

asctime	ctime	difftime
gmtime	localtime	time
clock	stime	tzset

AGRUPAMENTO DAS FUNÇÕES POR CATEGORIA

FUNÇÕES DE ALOCAÇÃO DE MEMÓRIA

	allocmem	groupmem	gpmem
allocmem	brk	calloc	gpmem
coreleft	farcalloc	farcoreleft	gpmem
farfree	farmalloc	farrealloc	gpmem
free	malloc	realloc	gpmem
sbrk	setblock	talloc	gpmem

FUNÇÕES RELATIVAS A JANELA DE TEXTOS

	video	text	text
clreol	clrscr	delline	text
gettext	gettextinfo	gotoxy	text
highvideo	insline	lwvideo	text
movetext	normvideo	puttext	text
textattr	textbackground	textcolor	text
textmode	wherex	wherey	text
window	window	window	text

FUNÇÕES DE CLASSIFICAÇÃO

	rot	rot	rot
isalnum	isalpha	isascii	HEMIT
iscntrl	isdigit	isgraph	ambas
islower	ispunct	ispunct	ambas
isspace	isupper	isxdigit	ambas

FUNÇÕES DIVERSAS

delay	longjmp	nosound
setjmp	sound	

FUNÇÕES GRÁFICAS

arc	bar	bar3d
circle	cleardevice	clearviewport
closegraph	detectgraph	drawpoly
ellipse	fillellipse	fillpoly
loodfill	getarccoords	getaspectratio
getbkcolor	getcolor	getdefaultpalette
getdrivername	getfillpattern	getfillsettings
getgraphmode	getimage	getlinesettings
getmaxcolor	getmaxmode	getmaxx
getmaxy	getmodename	getmoderange
getpalette	getpalettesize	getpixel
gettextsettings	getviewsettings	getx
gty	graphdefaults	grapherrmsg
_graphfreemem	_graphgetmem	graphresult
magesize	initgraph	installuserdriver
nstalluserfont	line	linerel
ineto	moverel	moveto
puttext	outtextxy	pieslice

putimage	putpixel	rectangle
registerbgidriver	registerbgifont	restorecrtmode
sector	setactivepage	setallpalette
setaspectratio	setbkcolor	setcolor
setfillpattern	setfillstyle	setgraphbufsize
setgraphmode	setlinestyle	setpalette
setrgbpalette	settextjustify	settextstyle
setusercharsize	setviewport	setvisualpage
setwritemode	textheight	textwidth

FUNÇÕES MATEMÁTICAS

abs	acos	asin
atan	atan2	atof
atoi	atol	cabs
ceil	_clear87	_control87
cos	cosh	div
ecvt	exp	fabs
fcvt	floor	fmod
_fpreset	frexp	gcvt
hypot	itoa	labs
ldexp	ldiv	log
log10	_lrotl	_lrottr
ltoa	_matherr	matherr
modf	poly	pow

pow10	rand	random
randomize	_rotl	_rotr
sin	sinh	sqrt
srand	_status87	strtod
strtol	strtoul	tan
tanh	ultoa	

FUNÇÕES DE CONTROLE DE PROCESSOS

abort	execl	execle
execlp	execlepe	execv
execve	execvp	execvpe
_exit	exit	getpid
raise	signal	spawnl
spawnle	spawnlp	spawnlpe
spawnnv	spawnve	spawnvp
spawnvpe	system	

FUNÇÕES PADRÃO

abort	abs	atexit
atof	atoi	atol
bsearch	calloc	ecvt
_exit	exit	fcvt
free	gcvt	getenv

itoa	labs	lfind	lflowr
lsearch	ltoa	malloc	lscanfbs
putenv	qsort	rand	lta
realloc	srand	strtod	lstrch
strtol	swab	system	lstrts
ultoa	swab	time	ltime

FUNÇÕES DE MANIPULAÇÃO

memccpy	memchr	memcmp	memrchr
memcpy	memicmp	memmove	memrvers
memset	movedata	movmem	memwchr
setmem	stpcpy	strcat	memwvers
strchr	strcmp	strcmpi	memwvers
strcpy	strcspn	strdup	memwvers
strerror	stricmp	strlen	memwvers
strlwr	strncat	strncpy	memwvers
strncmpi	strncpy	strnicmp	memwvers
strnset	strpbrk	strrchr	memwvers
strrev	strset	strspn	memwvers
strstr	strtok	strupr	memwvers

FUNÇÕES DE DATA E HORA

asctime	ctime	difftime	localtime
dostounix	ftime	getdate	gmtime

gettime	gmtime	localtime	tzset
setdate	settime	stime	tzset
time	tzset	unixtodos	tzset

FUNÇÕES DE ENTREDA E SAÍDA

access	cgets	_chmod	utime
chmod	chsize	clearerr	utime
_close	close	cprintf	utime
cputs	_creat	creat	utime
creatnew	createmp	cscanf	utime
dup	dup2	eof	utime
fclose	fcloseall	fdopen	utime
feof	ferror	fflush	utime
fgetc	fgetchar	fgetpos	utime
fgets	filelength	fileno	utime
flushall	fopen	fprintf	utime
fputc	fputchar	fputs	utime
fread	freopen	fscanf	utime
fseek	fsetpos	fstat	utime
ftell	fwrite	getchar	utime
getchar	getche	getftime	utime
getpass	gets	getw	utime
gsignal	ioctl	isatty	utime
kbhit	lock	lseek	utime

_open	open	perror	writing
printf	putc	putch	writing
puchar	puts	putw	writing
_read	read	remove	writing
rename	rewind	scanf	writing
setbuf	settime	setmode	writing
setvbuf	sopen	sprintf	writing
sscanf	stat	_strerror	writing
strerror	tell	tmpfile	writing
tmpnam	ungetc	ungetch	writing
unlock	vfprintf	vfscanf	writing
vprintf	vscanf	vsprintf	writing
vsscanf	_write	write	writing

FUNÇÕES DE INTERFACE

absread	abswrite	bdos	nodevice
bdosptr	bioscom	biosdisk	nodevice
biosequip	bioskey	biosmemory	nodevice
biosprint	biostime	country	nodevice
ctrlbrk	disable	dosexterr	nodevice
enable	FP_OFF	FP_SEG	nodevice
freemem	geninterrupt	getcbrk	nodevice
getdfree	getdta	getfat	nodevice
getfadt	getpsp	getvect	nodevice

getverify	harderr	hardresume
hardretn	import	importb
int86	int86x	intdos
intdosx	intr	keep
MK_FP	outport	outportb
parsfnm	peek	peekb
poke	pokeb	randbrd
randbwr	segread	setcbrk
setdta	setvect	setverify
sleep	unlink	

FUNÇÕES DE DIAGNÓSTICO

assert	matherr	perror
--------	---------	--------

FUNÇÕES DE CONTROLE DE DIRETÓRIO

chdir	findfirst	findnext
fnmerge	fnsplit	getcurdir
getcwd	getdisk	mkdir
mktemp	rmdir	searchpath
setdisk		

BIBLIOGRAFIA

- TURBO C REFERENCE GUIDE - Version 2.0
Borland International
- TURBO C USERS GUIDE - Version 2.0
Borland International

ОГЛАСОВАНО В ЗАРОДЫХ

ПОД РЕДАКЦИЕЙ А. СИДОРЕНКО

ИЗДАТОМ А. СИДОРЕНКО

ФОТОГРАФИИ А. СИДОРЕНКО

СТИЛЯГИ А. СИДОРЕНКО

Impressão e acabamento

(com filmes fornecidos):

EDITORIA SANTUÁRIO

Fone (0125) 36-2140

APARECIDA - SP