THƯ VIỆN TIME.H

Các đối tượng kiểu struct tm được sử dụng để lưu trữ ngày tháng và thời gian.

```
struct tm
{
int tm_sec;
int tm_min;
int tm_hour;
int tm_mday;
int tm_mon;
int tm_year; //year since 1900
int tm_wday; //days since Sunday
int tm_yday; //days since 1 January: [0,365]
int tm_isdst; //daylight saving time flag
}
```

Cờ tm_isdst là số dương (+) nếu daylight saving time có tác dụng, bằng 0 nếu không có, là số âm (-) nếu không có thông tin.

Mã quy cách	**Ý nghĩa**
%a	Tên ngày trong tuần viết tắt
%A	Tên ngày trong tuần đầy đủ
%b	Tên tháng viết tắt
%B	Tên tháng đầy đủ
%c	Date và time
%d	Ngày trong tháng
%H	Giờ trong ngày, 24 giờ
%h	Giờ trong ngày, 12 giờ
%j	Ngày trong năm
%m	Tháng
%M	Phút sau giờ
%p	AM hay PM
%s	Giây trong giờ
%U	Tuần trong năm
%w	Ngày trong tuần (0-6)
%x	Date
%X	Time
%y	Năm trong thế kỷ
%Y	Năm
%Z	Múi giờ
%%	Ký tự %

Truy nhập vào đồng hồ

```
clock_t clock (void);
```

Đây là số xung đồng hồ của máy. Muốn tính ra giây, ta đem chia với CLOCK_PER_SEC. Nếu tạo xung đồng hồ của CPU không có, hàm trả lại giá trị 1.

Truy nhập vào thời gian
double difftime (time_t t0, time_t t1);
char *asctime (const struct tm *tp);
size_t strftime (char *s, size_t n, const char *cntrl_str, const struct tm*tp);
structtm *gmtime (const time_t *t_ptr);
struct tm *localtime (const time_t *t_ptr);
time_t mktime (struct tm *tp);
time_t time (time_t *timer); nhận thời gian hệ thống quy ra giây
void getdate (struct date *datep); nhận ngày hệ thống
void setdate (struct date *datep); thiết lập ngày hệ thống
void gettime (struct time *timep); nhận giờ hệ thống
void settime (struct time *timep); thiết lập giờ hệ thống

THƯ VIỆN STRING.H

Các hàm quản lý bộ nhớ
int memcmp (const void *s1, const void *s2, size_t n); so sánh n byte trong 2 chuỗi s1, s2 (phân biệt chữ hoa, chữ thường)
Nếu giá trị trả về >0 thì chuỗi s1 > chuỗi s2
Nếu giá trị trả về <0 thì chuỗi s2 < chuỗi s1
Nếu giá trị trả về =0 thì chuỗi s2 giống chuỗi s1
int memicmp (const void *s1, const void *s2, size_t n); so sánh n byte trong 2 chuỗi s1, s2 (không phân biệt chữ hoa, chữ thường)
Nếu giá trị trả về >0 thì chuỗi s1 > chuỗi s2
Nếu giá trị trả về <0 thì chuỗi s2 < chuỗi s1
Nếu giá trị trả về =0 thì chuỗi s2 giống chuỗi s1

void *memchr (const void *s, int c, size_t n); tìm ký tự c trong n byte đầu của vùng s, nếu tìm thấy, hàm trả về địa chỉ của byte chứa ký tự c đầu tiên trong s, trái lại, trả về NULL
void *memccpy (void *dest, const void *src, int c, size_t n); sao chép các ký tự từ vùng src sang vùng dest, việc sao chép kết thúc khi gặp ký tự c hoặc đã sao chép đủ n ký tự
void *memcpy (void *dest, const void *src, size_t n); sao chép n ký tự từ vùng src sang vùng dest, hàm cho lại địa chỉ vùng dest
void *memmove (void *dest, const void *src, size_t n);
void *memset (void *s, int c, size_t n); gửi ký tự c vào n byte đầu của chuỗi s, hàm trả lại địa chỉ chuỗi s
void movedata (unsigned srcseg, unsigned srcoff, unsigned destseg, unsigned destoff, size_t n); sao chép n byte từ phân đoạn srcseg:srcoff đến địa chỉ destseg:destoff
Các hàm quản lý xâu ký tự
char *gets (char *s); nhập chuỗi
char *puts (char *s); xuất chuỗi
char *strcat (char *s1, const char *s2); ghép chuỗi s2 vào đuôi chuỗi s1
char *strchr (const char *s, int c); tìm ký tự c trong chuỗi s (bắt đầu từ bên trái), không có trả về NULL
char *strcpy (char *s1, const char *s2); sao chép nội dung trong s2 vào trong s1
char *strdup (const char *s); gấp đôi chuỗi s
char *strerror (int error_number);
char *strlwr (char *s); đổi chuỗi s thành chữ thường

char *strncat (char *s1, const char *s2, size_t n);
char *strncpy (char *s1, const char *s2, size_t n); sao chép tối đa n ký tự đầu của chuỗi s2 vào trong s1
char *strnset (char *s, int c, int n); gán n lần ký tự c vào trong chuỗi s
char *strpbrk (const char *s1, const char *s2); tìm lần xuất hiện đầu tiên của một ký tự thuộc s2 trong s1, nếu có, hàm cho địa chỉ của ký tự tìm thấy trong s1, trái lại, hàm cho NULL
char *strrchr (const char *s, int c); tìm ký tự c trong chuỗi s (bắt đầu từ bên phải), không có trả về NULL
char *strrev (char *s); đảo ngược các ký tự trong chuỗi s
char *strset (char *s, int c); đặt ký tự c vào mọi vị trí trong chuỗi s (thay các ký tự trong s bằng c)
char *strstr (const char *s1, const char *s2); tìm chuỗi s2 trong chuỗi s1, trả về vị trí chuỗi s2 trong chuỗi s1
char *strtok (char *s1, const char *s2);
char *strupr (char *s); đổi chuỗi s thành chữ hoa

int strcmp (const char *s1, const char *s2); trả về kết quả so sánh 2 chuỗi s1 và s2, không phân biệt chữ hoa, chữ thường của cùng một ký tự
Nếu giá trị trả về >0 thì chuỗi s1 chứa chuỗi s2
Nếu giá trị trả về <0 thì chuỗi s2 chứa chuỗi s1
Nếu giá trị trả về =0 thì chuỗi s2 giống chuỗi s1
int strcoll (const char *s1, const char *s2);
int stricmp (const char *s1, const char *s2); trả về kết quả so sánh 2 chuỗi s1 và s2, phân biệt chữ hoa, chữ thường của cùng một ký tự
Nếu giá trị trả về >0 thì chuỗi s1 chứa chuỗi s2
Nếu giá trị trả về <0 thì chuỗi s2 chứa chuỗi s1
Nếu giá trị trả về =0 thì chuỗi s2 giống chuỗi s1
int strncmp (const char *s1, const char *s2, size_t n); so sánh n ký tự đầu tiên của chuỗi s1 và s2
int strnicmp (const char *s1, const char *s2, size_t n); so sánh n ký tự đầu tiên của chuỗi s1 và s2, không phân biệt chữ hoa, chữ thường của cùng một ký tự

size_t strcspn (const char *s1, const char *s2); tìm độ dài đoạn đầu của chuỗi
size_t strlen (const char *s); xác định chiều dài chuỗi s
size_t strspn (const char *s1, const *s2); tìm độ dài đoạn đầu của chuỗi
size_t strxfrm (char *s1, const char *s2, size_t n);

unsigned *strlen (const char *s); xác định chiều dài chuỗi s

THƯ VIỆN STDLIB.H
File tiêu đề này chứa các nguyên mẫu của các hàm được sử dụng vào các mục đích chung, hoặc với các macro và các định nghĩa kiểu có liên quan.

Cấp phát bộ nhớ động
unsigned coreleft (void); cho biết bộ nhớ khả dụng trong vùng cấp phát động đối với mô hình tiny, small và medium
unsigned long coreleft (void); cho biết bộ nhớ khả dụng trong vùng cấp phát động đối với mô hình compact large và huge
void *calloc (size_t n, size_t size); cấp phát vùng nhớ cho n đối tượng kích cỡ size byte
void *malloc (size_t size); cấp phát vùng nhớ cho size byte

void *realloc (void *block, size_t size); cấp phát lại bộ nhớ
void free (void *block); giải phóng vùng nhớ đã cấp phát

Tìm kiếm và sắp xếp
void *bsearch (const void *key_ptr; const void *a_ptr, size_t n_els, size_t el_size, int compare (const void *, const void *));
void qsort (vois *a_ptr, size_t n_els, size_t el_size, int compare (const void *, const void *));

Tạo số nhẫu nhiên
int random (int n); tạo các số ngẫu nhiên trong khoảng từ 0 đến (num-1)
int rand (void); tạo số ngẫu nhiên trong khỏang từ 0 đến 32767
void randomize (void); khởi động cơ chế tạo số ngẫu nhiên bằng giá trị ngẫu nhiên
void srand (unsigned seed); khởi tạo bộ tạo số ngẫu nhiên bằng giá trị seed, không có giá trị trả lại

Truyền thông với môi trường
char *getenv (const char *name);
int system (const char *s);

Số học
int abs (int x);
long labs (long x);
div_t div (int numer, int denom);
ldiv_t ldiv (long numer, long deniom);

Chuyển đổi xâu ký tự
char *itoa (int x, char *s, int cs); chuyển số nguyên x trong hệ đếm cơ số cs sang chuỗi và lưu vào vùng nhớ s, hàm trả về địa chỉ của vùng s
char *ltoa (long x, char *s, int cs); chuyển số kiểu long x trong hệ đếm cơ số cs sang chuỗi và lưu vào vùng nhớ s, hàm trả về địa chỉ của vùng s
char *ultoa (unsigned long x, char *s, int cs); chuyển số kiểu unsigned long x trong hệ đếm cơ số cs sang chuỗi và lưu vào vùng nhớ s, hàm trả về địa chỉ của vùng s
double atof (const char *s); chuyển đổi xâu các chữ số str thành một số float
int atoi (const char*s); chuyển đổi xâu các chữ số str thành một số int
long atol (cont char *s); chuyển đổi xâu các chữ số str thành một số long
double strtod (const char *s, char **end_ptr);
long strtol (const char *s, char **end_ptr, int base);
unsigned long strtoul (const char *s, char **end_ptr, int base);
char *ecvt (double value, int ndig, int *dec, int *sign); chuyển giá trị kiểu double sang chuỗi chỉ gồm các chữ số
char *fcvt (double value, int ndig, int *dec, int *sign); chuyển giá trị kiểu double sang chuỗi chỉ gồm các chữ số
char *gcvt (double value, int ndec, char *buf); chuyển giá trị kiểu double sang chuỗi có cả dấu chấm thập phân và dấu (-) cho số âm

Các hàm ký tự nhiều byte
int mblen (const char *s, size_t n);
int mbrowc (wchar_t *p, const char *s, size_t n);
int wctomb (char *s, wchar_t wc);

Các hàm xâu ký tự nhiều byte
size_t mbstowcs (wchar_t *wcs, const char *mbs, size_t n);
int wcstombs (char *mbs, const wchar_t *wcs, size_t n);

Rời khỏi chương trình
void abort (void); kết thúc chương trình một cách không bình thường
int atexit (atexit_t func);
void exit (int status); kết thúc chương trình một cách bình thường

THƯ VIỆN CONIO.H

char *cgets (char *str);
char *getpass (const char *prompt); đọc password

Ví dụ:
#include
#include
main()
{
char *password;
password=getpass
cout<<"Enter password:"<
return 0;
}

extern int _wscroll;
int cputs (const char *str);
int fgetc (FILE *stream);
int fputc (int c, FILE *stream);
int getch (void); đọc một ký tự từ bàn phím, không hiện lên ký tự gõ vào
int getche (void); đọc một ký tự từ bàn phím, có hiện lại ký tự gõ vào. Ký tự e cuối có nghĩa là hiện lại (echo)
int gettext (int x1, int y1, int x2, int y2, void *destin);
int inp (unsigned portid);
int kbhit (void); kiểm tra xem có ký tự gõ vào hay không

Ví dụ:
#include
#include
main()
{
clrscr();
cout<<"Press any key";
while (!kbhit());
cout<<"\r\Continue\r\n";
return 0;

```
getch();
}

int movetext (int x1, int y1, int x2, int y2, int destleft, int desttop);
int outp (unsigned portid, int value);
int putch (int c); đưa một ký tự lên cửa sổ văn bản trên màn hình
int puttext (int x1, int y1, int x2, int y2, void *source);
int textmode (int mode);
int ungetch (int c);
int wherex (void); cho biết hoành độ hiện tại của con trỏ
int wherey (void); cho biết tung độ hiện tại của con trỏ
void _setcursortype (int cur_t);
void clrscr (void); xoá trắng màn hình (Clear Screen)
void clreol (void); xoá các ký tự nằm bên phải điểm nhắc (Clear End Of Line)
void delline (void); xoá một dòng trong cửa sổ
void gettextinfo (struct text_info *r); cho thông tin về kiểu hiển thị văn bản
void gotoxy (int x, int y); di chuyển con trỏ tới tọa độ (x,y) trên màn hình
void highvideo (void); làm độ sáng của ký tự tăng lên
void insline (void); xen một dòng trong cửa sổ
void lowvideo(void); làm độ sáng của ký tự yếu đi
void normvideo(void); làm độ sáng của ký tự bình thường
void textbackground (int color); chọn màu nền
void textcolor(int color); lựa chọn màu ký tự mới
void textattr (int attr); xác lập thuộc tính của ký tự trên màn hình
void window (int x1, int y1, int x2, int y2); tạo cửa sổ văn bản có tọa độ 2 góc: góc trên-bên trái (x1,y1) và góc
dưới-bên phải (x2,y2). Sau đó, mọi văn bản trên màn hình sẽ nằm trong cửa sổ này.
```

THƯ VIỆN STDIO.H

File tiêu đề này chứa các macro, các định nghĩa kiểu và các nguyên mẫu prototype của các hàm được người lập trình sử dụng để truy nhập vào file. Sau đây là một số macro và các định nghĩa kiểu:

```
#define BUFSIZ      1024 //kích thước cho tất cả cá bộ nhớ đệm
#define EOF      (-1) //giá trị trả lại của End Of File
#define FILENAME_MAX      255 //độ dài lớn nhất tên file
#define FOPEN_MAX      20 //số file lớn nhất có thể mở
#define L_tmpnam      16 //kích thước mảng cho tmp tên file
#define NULL      0 //giá trị con trỏ NULL
#define PATH_MAX      1024 //độ dài cực đại của đường dẫn
#define TMP_MAX      65535 //số lớn nhất của các tên file duy nhất

typedef long pos_t; //được sử dụng với fsetpos()
typedef unsigned size_t //kiểu từ toán tử sizeof
typedef char *va_list; //được sử ụng với họ vfprintf()
```

Cấu trúc file với từ khoá FILE có các thành phần mô tả trạng thái hiện tại của một file. Tên và số phần tử của nó phụ

thuộc vào từng hệ thống

Một đối tượng kiểu FILE có thể ghi tất cả các thông tin cần thiết để điều khiển một luồn (stream), kể cả một hiển thị hay cờ thông báo (indicator) vị trí của file, một con trỏ tới buffer của nó, một hiển thị hay cờ báo lỗi chứa các lỗi ghi/đọc có thể xảy ra, và một hiển thị hay cờ báo end of file để ghi nhận đã gặp dấu hiệu kết thúc file chưa.

Các macro được dùng để định nghĩa stdin, stdout và stderr. Mặc dù chúng ta nghĩ chúng là các file, song thực chất chúng là cá con trỏ.

#define stdin ($_iob[0])
#define stdout ($_iob[1])
#define stderr ($_iob[2])

Không giống các file khác,stdin, stdout và stderr không cần phải mở ra một cách tường minh.
Một số macro khác được sử dụng với các hàm như sau:

#define _IOFBF 0 //setvbuf(): full buffering
#define _IOFBF 0x80 //setvbuf(): full buffering
#define _IOFBF 0x04 //setvbuf(): full buffering
#define SEEK_SET 0 //fseek(): beginning of file
#define SEEK_CUR 1 //fseek(): current position in file
#define SEEK_END 2 //fseek(): end of file

Khi một file được mở, hệ điều hành kết nối nó với một stream và giữ thông tin về stream trong một đối tượng kiểu FILE. Một con trỏ trỏ tới FILE có thể xem như đang được kết nối với file đó hoặc với luồng stream, hoặc cả hai.

Các hàm cấp 2
Các hàm cấp 2 sử dụng cấu trúc FILE và mã kết thúc EOF, tất cả đều được khai báo và định nghĩa trong . Mã EOF bằng -1 còn cấu trúc FILE gồm các thành phần dung để quản lý tập tin như:
+ level cho biết có còn vùng đệm trong dữ liệu hay không
+ bsize độ lớn vùng đệm (mặc định là 512 bytes)
+ flags các cờ trạng thái
FILE *fdopen (int handle, char *type);
FILE *fopen (const char *filename, const char *mode); mở một file
Các đối (mode)
"r", "rt" mở một file để đọc theo kiểu văn bản, file cầ tồn tại, nếu không sẽ có lỗi
"w", "wt" mở một file để ghi theo kiểu văn bản, nếu file đã tồn tại, nó sẽ bị xoá
"a", "at" mở một file để ghi bổ sung theo kiểu văn bản, nếu file chưa tồn tại thì tạo file mới
"rb" mở một file để đọc theo kiểu nhị phân, file cần tồn tại, nếu không sẽ có lỗi
"wb" mở một file để ghi theo kiểu nhị phân, nếu file đã tồn tại, nó sẽ bị xoá
"ab" mở một file để ghi bổ sung theo kiểu nhị phân, nếu file chưa tồn tại thì tạo file mới
"r+", "r+t" mở một file để đọc/ghi theo kiểu văn bản, file cần tồn tại, nếu không sẽ có lỗi
"w+", "w+t" mở một file mới để đọc/ghi theo kiểu văn bản, nếu file đã tồn tại, nó sẽ bị xoá
"r+b" mở một file để đọc/ghi theo kiểu nhị phân, file cần tồn tại, nếu không sẽ có lỗi
"w+b" mở một file mới để đọc/ghi theo kiểu nhị phân, nếu file đã tồn tại, nó sẽ bị xoá
"a+b" mở một file để đọc/ghi bổ sung theo kiểu nhị phân, nếu file chưa tồn tại thì tạo file mới

int fclose (FILE *stream); đóng một file
int fcloseall (void); đóng tất cả các file đang mở
int fflush (FILE *stream); xoá vùng đệm bàn phím
int flushall (void); xoá vùng đệm bàn phím, thường sử dụng trước các hàm như gets hoặc scanf
FILE *freopen (const char *filename, const char *mode, FILE *fp);
FILE *_fsopen (const char *filename, const char *mode, int shflg);
void setbuf (FILE *stream, char *buf);
int setvbuf (FILE *stream, int mode, size_t n);
FILE *tmpfile (void);
char *tmpnam (char *s);

Truy nhập vào cờ vị trí file
int fseek (FILE *stream, long offset, int whence); chuyển con trỏ đến vị trí bất ký trên file (nên dùng theo kiểu nhị phân)
long ftell (FILE *stream); cho biết vị trí hiện tại của con trỏ chỉ vị
void rewind (FILE *stream); chuyển con trỏ về vị trí đầu file
int fgetpos (FILE *stream, fops_t *pos);
int fsetpos (FILE *stream, const fops_t *pos);

Quản lý lỗi
void clearer (FILE *stream);
int feof (FILE *stream); cho biết đã đến cuối file hay chưa
int ferror (FILE *stream); cho biết có lỗi (khác 0) hay không có lỗi (bằng 0)
void perror (const char *s); thông báo lỗi trên màn hình (khi biết có lỗi)

Các hàm xuất/nhập ký tự
int getc (FILE *stream); đọc ký tự từ file
int getchar (void); nhận một ký tự từ stdin, hàm trả về ký tự nhận được
char *gets (char *s); nhập một chuỗi ký tự từ stdin
int fgetc (FILE *stream); đọc ký tự từ file
char *fgets (char *c, int n, FILE *stream); đọc một chuỗi ký tự từ file
int fputc (int c, FILE *stream); ghi ký tự lên file
int fputs (const char *s, FILE *stream); ghi một chuỗi ký tự lên file
int putc (int c, FILE *stream); ghi ký tự lên file
int putchar (int c); đưa một ký tự ra stdout
int puts (const char *s); đưa một chuỗi ký tự ra stdout
int ungetc (int c, FILE *stream);

Các hàm nhập xuất theo kiểu văn bản
int cprintf (const char *format [,argument,...]);
int cscanf (const char *format [,address,...]);
int fprintf (FILE *stream, const char *format [,argument,...]); ghi dữ liệu theo khuôn dạng lên file
int fscanf (FILE *stream, const char *format [,address,...]); đọc dữ liệu theo khuôn dạng từ file
int printf (const char *format [,argument,...]); xuất dữ liệu theo định dạng
int scanf (const char *format [,address,...]); nhập dữ liệu theo định dạng

int sprintf (char *buffer, const char *format [,argument,...]);
int sscanf (const char *buffer, const char *format [,address,...]);
int vfprintf (FILE *stream, const char *format, va_list arglist);
int vfscanf (FILE *stream, const char *format, va_list arglist);
int vprintf (const char *format, va_list arglist);
int vscanf (const char *format, va_list arglist);
int vsprintf (char *buffer, const char *format, va_list arglist);
int vsscanf (char *buffer, const char *format, va_list arglist);

Các hàm nhập xuất theo kiểu nhị phân
int getw (FILE *stream); đọc một số nguyên từ file
int putw (int w, FILE *stream); ghi một số nguyên lên file
size_t fread (void *ptr, size_t size, size_t n, FILE *stream); đọc một số mẫu tin từ file
size_t fwrite (void *ptr, size_t size, size_t n, FILE *stream); ghi một số mẫu tin lên file

Xoá hoặc đổi tên file
int remove (const char *filename); xoá một file
int rename (const char *filename1, const char *filename2); đổi tên một file
int unlink (const char *filename); xoá một file

Các hàm nhập xuất cấp 1
Các file tiêu đề và biến chuẩn
Để sử dụng các hàm cấp 1, ta cần tới các file tiêu đề sau:
io.h chứa các nguyên mẫu của các hàm cấp 1
fcntl.h chứa các định nghĩa quyền truy nhập (access)
sys/stat.h chứa các định nghĩa thuộc tính (mode)
dos.h chứa các định nghĩa thuộc tính (attribute) theo DOS

Ngoài ra, còn cần đến biến chuẩn của C _fmode (định nghĩa trong fcntl.h và stdlib.h) để xác định kiểu nhập xuất (nhị phân hay văn bản)
int creat (const char *path, int mode); tạo một file mới có thuộc tính cho bởi mode. Trong trường hợp file đã tồn tại:
-Nếu file để ghi, nó sẽ bị xoá
-Nếu file để đọc thì bị lỗi
-Khi có lỗi, hàm trả về -1
-Khi thành công, hàm trả về số hiệu file (handle)
Thuộc tính
S_IREAD file để đọc, không thể xoá, sửa chữa, bổ sung
S_IWRITE file để ghi, có thể xoá, sửa chữa, bổ sung
int _creat (const char *path, int attrib); tạo một file mới theo kiểu nhị phân
Thuộc tính
FA_RDONLY file chỉ đọc, không thể xoá, sử chữa, bổ sung
FA_ARCH file để ghi
FA_HIDDEN file ẩn, không hiện trong lệnh DIR của DOS
int open (const char *path, int access [, unsigned mode]); mở một file đã có hoặc xây dựng file mới để đọc, ghi
Đối access
Giá trị Ý nghĩa

O_APPEND ghi bổ sung
O_BINARY kiểu nhập xuất nhị phân
O_CREAT tạo file (nếu chưa có)
O_RDONLY chỉ đọc
O_RDWR đọc và ghi
O_TEXT kiểu nhập xuất văn bản
O_TRUNC xoá file nếu có tồn tại
O_WRONLY nbsp;chỉ ghi

int _open (const char *filename, int oflag); mở một file đã tồn tại để đọc, ghi
int close (int handle); đóng một file
int _close (int handle); đóng một file
int chmod (const char *path, int amode); thay đổi thuộc tính file
int _chmod (const char *path, int func [, int attrib]); thay đổi thuộc tính file theo kiểu DOS
int write (int handle, void *buf, unsigned len); ghi một dãy các byte lên file, nếu thành công, hàm trả về môt số bằng số byte ghi được, có lỗi trả về -1
int read (int handle, void *buf, unsigned len); đọc một dãy các byte từ file
long lseek (int handle, long offset, int fromwhere); di chuyển con trỏ chỉ vị

DANH MỤC CÁC HÀM TRONG THƯ VIỆN ĐỒ HỌA
arc vẽ cung tròn có góc bắt đầu, góc kết thúc, tọa độ tâm
bar vẽ hình chữ nhật có tô bên trong bar3d vẽ hình chữ nhật theo không gian 3 chiều có tô bên trong
circle vẽ hình tròn
cleardevice xoá màn hình, đưa con trỏ về góc trên-bên trái
clearviewport xoá khung hình
closegraph đóng chế độ đồ họa
detectgraph kiểm tra phần cứng và xác định trình điều khiển và chế độ
drawpoly vẽ đa giác với kiểu nết vẽ và màu hiện tại
ellipse vẽ cung elip
fillellipse vẽ hình elip có tô màu
fillpoly tô đa giác có sử dụng bộ chuyển đổi quét
floodfill tô một miền bị chặn, dung mẫu tô và màu hiện tại
getarccoords nhận lại tọa độ để vẽ cung
getaspectratio trả lại hệ số tương quan tỷ lệ trên màn hình
getbkcolor nhận lại màu nền hiện tại
getcolor nhận lại màu vẽ hiện tại
getdefaultpalette nhận lại bảng màu ngầm định
getdrivername nhận lại tên vỉ mạch đồ họa
getfillpattern nhận lại mẫu tô
getfillsettings nhận lại mẫu tô được thiết lập mới nhất
getgraphmode nhận lại chế độ đồ họa hiện tại
getimage cất ảnh bit của một vùng hình vào trong bộ nhớ đệm
getlinesettings nhận lại kiểu vẽ, nét vẽ và độ dày nét vẽ
getmaxcolor nhận lại giá trị màu lớn nhất có thể có của chế độ đồ họa
getmaxmode nhận lại giá trị chế độ cao nhất có thể có

getmaxx nhận lại giá trị độ phân giải ngang
getmaxy nhận lại giá trị độ phân giải dọc
getmodename nhận lại tên chế độ đồ họa
getmoderamge nhận lại chế độ lớn nhất và thấp nhất của vỉ đồ họa
getpalettesize nhận lại giá trị bảng màu
getpixel nhận lại màu của điểm vẽ
getpalette nhận lại giá trị bảng màu
gettextsettings nhận lại giá trị về kiểu chữ, hướng viết, kích thước
getviewsettings nhận lại thông tin về khung hình và các tham số
getx nhận lại tọa độ x của vị trí đồ họa hiện tại
gety nhận lại tọa độ y của vị trí đồ họa hiện tại
graphdefaults đưa vị trí con trỏ hiện tại về góc trên bên trái, khởi động lại chế độ đồ họa
grapherrormsg nhận lại các xâu ký tự thông báo lỗi cho errorcode
graphresult nhận lại giá trị báo lỗi của thao tác đồ họa cuối cùng
imagesize trả lại giá trị số byte cần thiết để cất một vùng chữ nhật trên màn hình
intalluserdrive cài đặt các trình điều khiển đồ họa mới vào bảng BGI
intalluserfont cài đặt một font chữ mới chưa ó trong hệ thống BGI
initgraph khởi tạo đê vào chế độ đồ họa
line vẽ một đoạn thẳng giữa 2 điểm chỉ rõ
linerel vẽ một đoạn thẳng với khoảng cách tương đối
lineto vẽ một đoạn thẳng từ điểm hiện tại tới...
moverel dịch chuyển vị trí hiện tại tới điểm mới theo tọa độ tương đối
moveto dịch chuyển vị trí hiện tại tới điểm mới
outtext viết ra dòng văn bản tại vị trí hiện tại
outteaxtxy viết ra dòng văn bản tại vị trí (x,y)
pieslice vẽ một miếng bánh tròn
putimage nạp hình ảnh bit vào màn hình
putpixel vẽ một điểm ảnh tại tọa độ (x,y)
rectangle vẽ hình chữ nhật không tô bên trong với màu và nét vẽ hiện tại
registerbgidriver đăng ký trình điều khiển BGI với hệ thống đồ họa
registerbgifont đăng ký font BGI với hệ thống đồ họa
restorecrtmode khôi phục lại chế độ màn hình gốc trước khi chế độ đồ họa được khởi tạo để dùng
sector vẽ và tô một miếng khung hình elip
setactivepage thay đổi trang tích cực để cho ra đồ họa
setallpalette thay đổi toàn bộ bảng màu
setaspectratio thay đổi tỷ lệ tương quan ngang dọc
setbkcolor đặt màu nền
setcolor đặt màu vẽ hiện tại
setfillpattern đặt mẫu tô do người dung định nghĩa
setfillstyle đặt mẫu và màu tô
setgraphbufsize thay đổi kích thước bộ nhớ đệm để quét và tô
setgraphmode đặt hệ thống tới chế độ đồ họa và xoá màn hình
setlinestyle đặt kiểu nét vẽ
setpalette thay đổi giá trị bảng màu
setrgbppalette thay đổi giá trị bảng màu cho vỉ mạch IBM8514 và VGA
settextjustify đặt chế độ căn lề cho outtext và outtextxy

settexttyle thiết lập font chữ, hướng, kích thước viết chũ đồ họa
setusercharsize thay đổi độ rộng và chiều cao font vector
setviewport thiết lập khung nhình đồ họa
setvisualpage thiết lập số trang nhìn
setwritemode thiết lập cách thức ghi lên màn hình vẽ là COPY đè lên hay XNOR
textheight trả lại độ cao của xâu chữ, tính theo pixel
textwidth trả lại độ rộng củ xâu chữ, tính theo pixel

THƯ VIỆN PROCESS.H

Các hàm kiểm soát quá trình
int system (const char *command); thực hiệm một câu lệnh DOS, thành công trả về 0, có lỗi trả về -1
void abort (void); kết thúc chương trình một cách không bình thường
void exit (int status); kết thúc chương trình một cách bình thường

THƯ VIỆN MATH.H
Các hàm toán học dấu chấm động
double acos (double x); trả về arc cosine của x, được biểu diễn từ 0 đến Л
double asin (double x); trả về arc sine của x, được biểu diễn từ -Л/2 đến Л/2
double atan (double x); trả về arc tangent của x, được biểu diễn từ -Л/2 đến Л/2
double atan2 (double x, double y); trả về arc tangent của x/y, được biểu diễn từ -Л đến Л
double cabs (struct complex x); trả về giá trị tuyệt đối của số phức x
double ceil (double x); trả về phần nguyên của số chấm động không nhỏ hơn x (làm tròn số lên)
double cos (double x); trả về cos của x, được biểu diễn theo radian
double cosh (double x); trả về giá trị cosine hyperbolic của x
double exp (double x); trả về ex
double fabs (double x); trả về giá trị tuyệt đối của số thực x
double floor (double x); trả về phần nguyên của số chấm động không lớn hơn x (làm tròn số xuống)
double fmod (double x, double y); lấy phần dư của phép chia (x/y), y phải khác 0
double frexp (double x, int *exponent); lấy phần giá trị của x khi tách nhỏ m trong khoảng 0,5 đến 1 hay m=0
double ldexp (double x, int exponent);
double log (double x); trả về logarit tự nhiên của x
double log10 (double x); trả về logarit cơ số 10 của x
double modf (double x, double *ipart); tách số x thành phần số nguyên và phần số lẻ sau dấu chấm thập phân, cất phần nguyên trong *ipart và trả về phần lẻ
double pow (double x, double y); trả về xy
double pow10 (int x); trả về x10
double sin (double x); trả về sin của x, được biểu diễn theo radian
double sinh (double x); trả về giá trị sine hyperbolic của x
double sqrt (double x); trả về căn bậc 2 của x
double tan (double x); trả về tang của x, được biểu diễn theo radian
double tanh (double x); trả về giá trị tangent hyperbolic của x
int abs (int x); trả về giá trị tuyệt đối của số nguyên x
long double cabsl (struct complexl (x));
long double frexp (long double (x));

long double fabsl (long double @E (x));
long double frexp (long double (x), int *(exponent));
long double ldexpl (long double (x), int (exponent));
long double modfl (long double (x), long double *(ipart));
long double pow101 (int (x)); trả về x101
long int labs (long int x);

THƯ VIỆN DOS.H
Các hàm truy nhập trực tiếp vào bộ nhớ
char peekb (unsigned segment, unsigned offset); nhận một byte tại địa chỉ phân đoạn segment:offset
int peek (unsigned segment, unsigned offset); nhận lại một từ tại địa chỉ phân đoạn segment:offset
void poke (unsigned segment, unsigned offset, int value); gửi giá trị nguyên value vào bộ nhớ tại địa chỉ phân đoạn segment:offset
void pokeb (unsigned segment, unsigned offset, char value); gửi giá trị ký tự value vào bộ nhớ tại địa chỉ phân đoạn segment:offset

THƯ VIỆN DIR.H
Các hàm kiểm soát thư mục
char *getcwd (char *buf, int buflen); lấy tên thư mục chủ, hàm trả về buf
int chdir (const char *path); đổi thư mục chủ (có đường dẫn), nếu thành công trả về 0, có lỗi trả về -1
int findfirst (const char *pathname, struct ffblk *ffblk, int attrib); tìm file trên thư mục (có đường dẫn, thuộc tính), nếu tìm thấy trả về 0, có lỗi trả về -1
int findnext (struct ffblk *ffblk); tiếp tục tìm file trên thư mục theo các chỉ dẫn cho trong ffblk, nếu tìm thấy trả về 0, có lỗi trả về -1
int getcurdir (int driver, char *directory); chuyển thư mục hiện hành), nếu thành công trả về 0, có lỗi trả về -1
int mkdir (const char *path); tạo thư mục mới (có đường dẫn), nếu thành công trả về 0, có lỗi trả về -1
int rmdir (const char *path); xoá thư mục (có đường dẫn), nếu thành công trả về 0, có lỗi trả về -1

THƯ VIỆN CTYPE.H
Các hàm quản lý, kiểm tra ký tự
int isalnum (int c); trả về khác 0 nếu c biểu diễn một ký tự alphanumeric (chữ cái hay chữ số)
int isalpha (int c); trả về khác 0 nếu c biểu diễn một ký tự chữ alphabetic (A-Z hay a-z)
int isascii (int c); trả về khác 0 nếu c biểu diễn một ký tự có mã ASCII từ 0-127
int iscntrl (int c); trả về khác 0 nếu c biểu diễn một ký tự điều khiển, có mã ASCII từ 0 đến 0x1F hoặc mã bằng 0x7F (DEL)
int isdigit (int c); trả về khác 0 nếu c biểu diễn một ký tự số (0-9)
int isgraph (int c); trả về khác 0 nếu c biểu diễn một ký tự in được, có mã ASCII từ 0x21 đến 0x7E (không kể ký tự khoảng trống)
int islower (int c); trả về khác 0 nếu c biểu diễn một ký tự chữ thường (a-z)
int isprint (int c); trả về khác 0 nếu c biểu diễn một ký tự in được, có mã ASCII từ 0x20 đến 0x7E (kể cả ký tự khoảng trống)
int ispunct (int c); trả về khác 0 nếu c biểu diễn một ký tự dấu (khác ký tự alphanumeric và ký tự khoảng trống)
int isspace (int c); trả về khác 0 nếu c biểu diễn một ký tự khoảng trống
int isupper (int c); trả về khác 0 nếu c biểu diễn một ký tự chữ hoa (A-Z)
int isxdigit (int c); trả về khác 0 nếu c biểu diễn một ký tự số thập lục phân (0-9, A-F hay a-f)
int toascii (int c); chuyển c về mã ASCII tương ứng nếu c biểu diễn một ký tự trong bảng mã ASCII

int tolower (int c); chuyển c thành ký tự thường tương ứng nếu c biểu diễn một ký tự hoa
int toupper (int c); chuyển c thành ký tự hoa tương ứng nếu c biểu diễn một ký tự thường