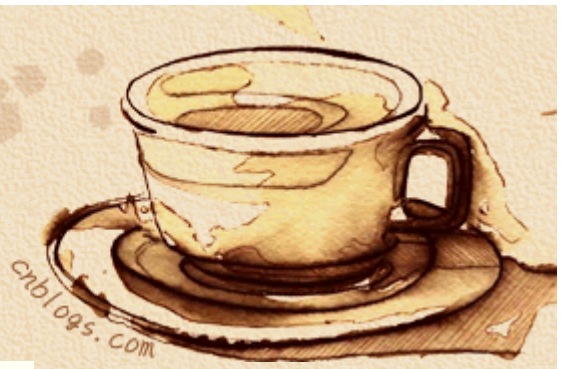
Arther's Blog



准备是成功的首要前提

c语言库函数大全--资料收集

Turbo C 2.0 函数中文说明大全

分类函数,所在函数库为ctype.h
int isalpha(int ch) 若ch是字母('A'-'Z','a'-'z')返回非0值,否则返回0
int isalnum(int ch) 若ch是字母('A'-'Z','a'-'z')或数字('0'-'9'),返回非0值,否则返回0
int isascii(int ch) 若ch是字符(ASCII码中的0-127)返回非0值,否则返回0
int iscntrl(int ch) 若ch是作废字符(0x7F)或普通控制字符(0x00-0x1F),返回非0值,否则返回0
int isdigit(int ch) 若ch是数字('0'-'9')返回非0值,否则返回0
int isgraph(int ch) 若ch是可打印字符(不含空格)(0x21-0x7E)返回非0值,否则返回0
int islower(int ch) 若ch是小写字母('a'-'z')返回非0值,否则返回0
int isprint(int ch) 若ch是可打印字符(含空格)(0x20-0x7E)返回非0值,否则返回0
int ispunct(int ch) 若ch是标点字符(0x00-0x1F)返回非0值,否则返回0
int isspace(int ch) 若ch是空格(' '),水平制表符('\t'),回车符('\r'), 走纸换行('\f'),垂直制表符('\v'),换行符('\n'), 返回非0值,否则返回0
int isupper(int ch) 若ch是大写字母('A'-'Z')返回非0值,否则返回0
int isxdigit(int ch) 若ch是16进制数('0'-'9','A'-'F','a'-'f')返回非0值, 否则返回0
int tolower(int ch) 若ch是大写字母('A'-'Z')返回相应的小写字母('a'-'z')
int toupper(int ch) 若ch是小写字母('a'-'z')返回相应的大写字母('A'-'Z')

数学函数,所在函数库为math.h、stdlib.h、string.h、float.h
int abs(int i) 返回整型参数i的绝对值
double cabs(struct complex znum) 返回复数znum的绝对值
double fabs(double x) 返回双精度参数x的绝对值
long labs(long n) 返回长整型参数n的绝对值
double exp(double x) 返回指数函数ex的值
double frexp(double value,int *eptr) 返回value=x*2n中x的值,n存贮在eptr中
double ldexp(double value,int exp); 返回value*2exp的值
double log(double x) 返回logex的值
double log10(double x) 返回log10x的值
double pow(double x,double y) 返回xy的值
double pow10(int p) 返回10p的值
double sqrt(double x) 返回x的开方
double acos(double x) 返回x的反余弦cos-1(x)值,x为弧度

与我联系

发短消息

搜索

找找看

常用链接

我的随笔
我的空间
我的短信
我的评论
更多链接

留言簿

给我留言
查看留言

随笔分类(174)

.Net (rss)
Algorithm(15) (rss)
Basic(10) (rss)
C&C++(33) (rss)
DBase(2) (rss)
Essay(112) (rss)
Java(2) (rss)

随笔档案(208)

2007年8月 (1)
2007年5月 (5)
2007年4月 (4)
2007年3月 (1)
2007年1月 (21)
2006年12月 (48)
2006年11月 (121)
2006年10月 (7)

double asin(double x) 返回x的反正弦sin-1(x)值,x为弧度
double atan(double x) 返回x的反正切tan-1(x)值,x为弧度
double atan2(double y,double x) 返回y/x的反正切tan-1(x)值,y的x为弧度
double cos(double x) 返回x的余弦cos(x)值,x为弧度
double sin(double x) 返回x的正弦sin(x)值,x为弧度
double tan(double x) 返回x的正切tan(x)值,x为弧度
double cosh(double x) 返回x的双曲余弦cosh(x)值,x为弧度
double sinh(double x) 返回x的双曲正弦sinh(x)值,x为弧度
double tanh(double x) 返回x的双曲正切tanh(x)值,x为弧度
double hypot(double x,double y) 返回直角三角形斜边的长度(z), x和y为直角边的长度,z2=x2+y2
double ceil(double x) 返回不小于x的最小整数
double floor(double x) 返回不大于x的最大整数
void srand(unsigned seed) 初始化随机数发生器
int rand() 产生一个随机数并返回这个数
double poly(double x,int n,double c[]) 从参数产生一个多项式
double modf(double value,double *iptr) 将双精度数value分解成尾数和阶
double fmod(double x,double y) 返回x/y的余数
double frexp(double value,int *eptr) 将双精度数value分成尾数和阶
double atof(char *nptr) 将字符串nptr转换成浮点数并返回这个浮点数
double atoi(char *nptr) 将字符串nptr转换成整数并返回这个整数
double atol(char *nptr) 将字符串nptr转换成长整数并返回这个整数
char *ecvt(double value,int ndigit,int *decpt,int *sign)
将浮点数value转换成字符串并返回该字符串
char *fcvt(double value,int ndigit,int *decpt,int *sign)
将浮点数value转换成字符串并返回该字符串
char *gcvt(double value,int ndigit,char *buf)
将数value转换成字符串并存于buf中,并返回buf的指针
char *ultoa(unsigned long value,char *string,int radix)
将无符号整型数value转换成字符串并返回该字符串,radix为转换时所用基数
char *ltoa(long value,char *string,int radix)
将长整型数value转换成字符串并返回该字符串,radix为转换时所用基数
char *itoa(int value,char *string,int radix)
将整数value转换成字符串存入string,radix为转换时所用基数
double atof(char *nptr) 将字符串nptr转换成双精度数,并返回这个数,错误返回0
int atoi(char *nptr) 将字符串nptr转换成整型数, 并返回这个数,错误返回0

long atol(char *nptr) 将字符串nptr转换成长整型数,并返回这个数,错误返回0
double strtod(char *str,char **endptr)将字符串str转换成双精度数,并返回这个数,
long strtol(char *str,char **endptr,int base)将字符串str转换成长整型数, 并返回这个数,
int matherr(struct exception *e) 用户修改数学错误返回信息函数(没有必要使用)
double _matherr(_mexcep why,char *fun,double *arg1p, double *arg2p,double retval)
用户修改数学错误返回信息函数(没有必要使用)
unsigned int _clear87() 清除浮点状态字并返回原来的浮点状态
void _fpreset() 重新初使化浮点数学程序包
unsigned int _status87() 返回浮点状态字

相册

BOOK
photo
自己做的

Expert

anjou
jeffreyzhao
Ricktroy
shenfx
吕震宇
随心所欲
小寒

Friends

Hack Site

Hack a day
Hack And Slash
Hack Canada
It don't mean jack if it ain't got that hack.
Hack This site
Hack-the-planet
MIT IHTFP Gallery
T-hack
Wilhelm Hack Museum

Links

2600.com
80x86汇编小站
BlogJava
C++Builder研究
C++博客
CMS系统中文网站
C语言学习
Flier's Sky
HTML教程
IBM-java-中国
IT博客网
J道:Java和J2EE解决之道
linux宝库
Linux操作指南
linux技术中坚站
Linux网站大全
Linux伊甸园
PHP文章PHP教程基础
PHP中文站
Rootkit.com
SourceForge
VB-问专家

目录函数,所在函数库为dir.h、dos.h
int chdir(char *path) 使指定的目录path（如:"C:\\WPS"）变成当前的工作目录,成功返回0
int findfirst(char *pathname,struct ffblk *ffblk,int attrib)
查找指定的文件,成功返回0
pathname为指定的目录名和文件名,如"C:\\WPS\\TXT"
ffblk为指定的保存文件信息的一个结构,定义如下:

```
struct ffblk
{
 char ff_reserved[21]; /*DOS保留字*/
 char ff_attrib; /*文件属性*/
 int ff_ftime; /*文件时间*/
 int ff_fdate; /*文件日期*/
 long ff_fsize; /*文件长度*/
 char ff_name[13]; /*文件名*/
}
```

attrib为文件属性,由以下字符代表

FA_RDONLY 只读文件	FA_LABEL 卷标号
FA_HIDDEN 隐藏文件	FA_DIREC 目录
FA_SYSTEM 系统文件	FA_ARCH 档案

例:

```
struct ffblk ff;
findfirst("*.wps",&ff,FA_RDONLY);
```

int findnext(struct ffblk *ffblk) 取匹配finddirst的文件,成功返回0
void fumerge(char *path,char *drive,char *dir,char *name,char *ext)
此函数通过盘符drive(C:、A:等), 路径dir(\TC、\BC\LIB等), 文件名name(TC、WPS等),扩展名ext(.EXE、.COM等)组成一个文件名存与path中.
int fnsplit(char *path,char *drive,char *dir,char *name,char *ext)
此函数将文件名path分解成盘符drive(C:、A:等), 路径dir(\TC、\BC\LIB等), 文件名name(TC、WPS等),扩展名ext(.EXE、.COM等),并分别存入相应的变量中.
int getcurdir(int drive,char *direc)
此函数返回指定驱动器的当前工作目录名称。成功返回0
drive 指定的驱动器(0=当前,1=A,2=B,3=C等)
direc 保存指定驱动器当前工作路径的变量
char *getcwd(char *buf,iint n) 此函数取当前工作目录并存入buf中,直到n个字节长为为止.错误返回NULL
int getdisk() 取当前正在使用的驱动器,返回一个整数(0=A,1=B,2=C等)
int setdisk(int drive) 设置要使用的驱动器drive(0=A,1=B,2=C等), 返回可使用驱动器总数
int mkdir(char *pathname) 建立一个新的目录pathname,成功返回0
int rmdir(char *pathname) 删除一个目录pathname,成功返回0
char *mktemp(char *template) 构造一个当前目录上没有的文件名并存于template中
char *searchpath(char *pathname) 利用MSDOS找出文件filename所在路径, 此函数使用DOS的PATH变量,未找到文件返回NULL

进程函数,所在函数库为stdlib.h、process.h
void abort() 此函数通过调用具有出口代码3的_exit写一个终止信息于stder

r，并异常终止程序。无返回值

int exec…装入和运行其它程序
int execl(char *pathname,char *arg0,char *arg1,…,char *argn,NULL)
int execle(char *pathname,char *arg0,char *arg1,…, char *argn,NULL,char *envp[])
int execlp(char *pathname,char *arg0,char *arg1,…,NULL)
int execlpe(char *pathname,char *arg0,char *arg1,…,NULL,char *envp[])
int execv(char *pathname,char *argv[])
int execve(char *pathname,char *argv[],char *envp[])
int execvp(char *pathname,char *argv[])
int execvpe(char *pathname,char *argv[],char *envp[])

exec函数族装入并运行程序pathname，并将参数arg0(arg1,arg2,argv[],envp[])传递给子程序,出错返回-1。
在exec函数族中,后缀l、v、p、e添加到exec后，所指定的函数将具有某种操作能力。
有后缀 p时，函数可以利用DOS的PATH变量查找子程序文件。
l时，函数中被传递的参数个数固定。
v时，函数中被传递的参数个数不固定。
e时，函数传递指定参数envp，允许改变子进程的环境，
无后缀 e时，子进程使用当前程序的环境。

void _exit(int status) 终止当前程序,但不清理现场
void exit(int status) 终止当前程序,关闭所有文件,写缓冲区的输出(等待输出), 并调用任何寄存器的"出口函数",无返回值

int spawn…运行子程序
int spawnl(int mode,char *pathname,char *arg0,char *arg1,…, char *argn,NULL)
int spawnle(int mode,char *pathname,char *arg0,char *arg1,…, char *argn,NULL,char *envp[])
int spawnlp(int mode,char *pathname,char *arg0,char *arg1,…, char *argn,NULL)
int spawnlpe(int mode,char *pathname,char *arg0,char *arg1,…, char *argn,NULL,char *envp[])
int spawnv(int mode,char *pathname,char *argv[])
int spawnve(int mode,char *pathname,char *argv[],char *envp[])
int spawnvp(int mode,char *pathname,char *argv[])
int spawnvpe(int mode,char *pathname,char *argv[],char *envp[])

spawn函数族在mode模式下运行子程序pathname,并将参数arg0(arg1,arg2,argv[],envp[])传递给子程序.出错返回-1
mode为运行模式：
mode为 P_WAIT 表示在子程序运行完后返回本程序
P_NOWAIT 表示在子程序运行时同时运行本程序(不可用)
P_OVERLAY 表示在本程序退出后运行子程序

在spawn函数族中,后缀l、v、p、e添加到spawn后，所指定的函数将具有某种操作能力
有后缀 p时，函数利用DOS的PATH查找子程序文件
l时，函数传递的参数个数固定.

积分与排名

积分 - 16840
排名 - 2913

最新评论

阅读排行榜

评论排行榜

v时, 函数传递的参数个数不固定.
e时, 指定参数envp可以传递给子程序,允许改变子程序运行环境.
无后缀 e时,子程序使用本程序的环境.

int system(char *command)
将MSDOS命令command传递给DOS执行转换子程序,函数库为math.h、st
dlib.h、ctype.h、float.h
char *ecvt(double value,int ndigit,int *decpt,int *sign)
将浮点数value转换成字符串并返回该字符串
char *fcvt(double value,int ndigit,int *decpt,int *sign)
将浮点数value转换成字符串并返回该字符串
char *gcvt(double value,int ndigit,char *buf)
将数value转换成字符串并存于buf中,并返回buf的指针
char *ultoa(unsigned long value,char *string,int radix)
将无符号整型数value转换成字符串并返回该字符串,radix为转换时所用基数
char *ltoa(long value,char *string,int radix)
将长整型数value转换成字符串并返回该字符串,radix为转换时所用基数
char *itoa(int value,char *string,int radix)
将整数value转换成字符串存入string,radix为转换时所用基数
double atof(char *nptr) 将字符串nptr转换成双精度数,并返回这个数,错误
返回0
int atoi(char *nptr) 将字符串nptr转换成整型数, 并返回这个数,错误返回0

long atol(char *nptr) 将字符串nptr转换成长整型数,并返回这个数,错误返
回0
double strtod(char *str,char **endptr)
将字符串str转换成双精度数,并返回这个数,
long strtol(char *str,char **endptr,int base)
将字符串str转换成长整型数, 并返回这个数,
int toascii(int c) 返回c相应的ASCII
int tolower(int ch) 若ch是大写字母('A'-'Z')返回相应的小写字母('a'- 'z')
int _tolower(int ch) 返回ch相应的小写字母('a'-'z')
int toupper(int ch) 若ch是小写字母('a'-'z')返回相应的大写字母('A'- 'Z')
int _toupper(int ch) 返回ch相应的大写字母('A'-'Z')

诊断函数,所在函数库为assert.h、math.h
void assert(int test) 一个扩展成if语句那样的宏，如果test测试失败，就显
示一个信息并异常终止程序,无返回值
void perror(char *string) 本函数将显示最近一次的错误信息，格式如：字
符串string:错误信息
char *strerror(char *str) 本函数返回最近一次的错误信息,格式如: 字符串
str:错误信息
int matherr(struct exception *e)
用户修改数学错误返回信息函数(没有必要使用)
double _matherr(_mexcep why,char *fun,double *arg1p, double *
arg2p,double retval)
用户修改数学错误返回信息函数(没有必要使用)

输入输出子程序, 函数库为io.h、conio.h、stat.h、dos.h、stdio.h、signa
l.h

int kbhit() 本函数返回最近所敲的按键
int fgetchar() 从控制台(键盘)读一个字符，显示在屏幕上
int getch() 从控制台(键盘)读一个字符，不显示在屏幕上
int putch() 向控制台(键盘)写一个字符

int getchar() 从控制台(键盘)读一个字符，显示在屏幕上
int putchar() 向控制台(键盘)写一个字符
int getche() 从控制台(键盘)读一个字符，显示在屏幕上
int ungetch(int c) 把字符c退回给控制台(键盘)
char *cgets(char *string) 从控制台(键盘)读入字符串存于string中
int scanf(char *format[,argument…])
从控制台读入一个字符串,分别对各个参数进行赋值,使用BIOS进行输出
int vscanf(char *format,Valist param)
从控制台读入一个字符串,分别对各个参数进行赋值,使用BIOS进行输出,参数从Valist param中取得
int cscanf(char *format[,argument…])
从控制台读入一个字符串,分别对各个参数进行赋值,直接对控制台作操作,比如显示器在显示时字符时即为直接写频方式显示
int sscanf(char *string,char *format[,argument,…])
通过字符串string, 分别对各个参数进行赋值
int vsscanf(char *string,char *format,Vlist param)
通过字符串string,分别对各个参数进行赋值,参数从Vlist param中取得
int puts(char *string) 发关一个字符串string给控制台(显示器), 使用BIOS进行输出
void cputs(char *string) 发送一个字符串string给控制台(显示器), 直接对控制台作操作,比如显示器即为直接写频方式显示
int printf(char *format[,argument,…])
发送格式化字符串输出给控制台(显示器)，使用BIOS进行输出
int vprintf(char *format,Valist param)
发送格式化字符串输出给控制台(显示器)，使用BIOS进行输出,参数从Valist param中取得
int cprintf(char *format[,argument,…])
发送格式化字符串输出给控制台(显示器), 直接对控制台作操作,比如显示器即为直接写频方式显示
int vcprintf(char *format,Valist param)
发送格式化字符串输出给控制台(显示器), 直接对控制台作操作,比如显示器即为直接写频方式显示, 参数从Valist param中取得
int sprintf(char *string,char *format[,argument,…])
将字符串string的内容重新写为格式化后的字符串
int vsprintf(char *string,char *format,Valist param)
将字符串string的内容重新写为格式化后的字符串,参数从Valist param中取得
int rename(char *oldname,char *newname)将文件oldname的名称改为newname
int ioctl(int handle,int cmd[,int *argdx,int argcx])
本函数是用来控制输入/输出设备的，请见下表：

cmd值	功能
0	取出设备信息
1	设置设备信息
2	把argcx字节读入由argdx所指的地址
3	在argdx所指的地址写argcx字节
4	除把handle当作设备号(0=当前,1=A,等)之外,均和cmd=2时一样
5	除把handle当作设备号(0=当前,1=A,等)之外,均和cmd=3时一样
6	取输入状态

7	取输出状态
8	测试可换性;只对于DOS 3.x
11	置分享冲突的重算计数;只对DOS 3.x

int (*ssignal(int sig,int(*action)())() 执行软件信号(没必要使用)
int gsignal(int sig) 执行软件信号(没必要使用)

int _open(char *pathname,int access)为读或写打开一个文件, 按后按access来确定是读文件还是写文件,access值见下表

access值	意义
O_RDONLY	读文件
O_WRONLY	写文件
O_RDWR	即读也写
O_NOINHERIT	若文件没有传递给子程序,则被包含
O_DENYALL	只允许当前处理必须存取的文件
O_DENYWRITE	只允许从任何其它打开的文件读
O_DENYREAD	只允许从任何其它打开的文件写
O_DENYNONE	允许其它共享打开的文件

int open(char *pathname,int access[,int permiss])为读或写打开一个文件, 按后按access来确定是读文件还是写文件,access值见下表

access值	意义
O_RDONLY	读文件
O_WRONLY	写文件
O_RDWR	即读也写
O_NDELAY	没有使用;对UNIX系统兼容
O_APPEND	即读也写,但每次写总是在文件尾添加
O_CREAT	若文件存在,此标志无用;若不存在,建新文件
O_TRUNC	若文件存在,则长度被截为0,属性不变
O_EXCL	未用;对UNIX系统兼容
O_BINARY	此标志可显示地给出以二进制方式打开文件
O_TEXT	此标志可用于显示地给出以文本方式打开文件

permiss为文件属性,可为以下值:
S_IWRITE允许写 S_IREAD允许读 S_IREAD|S_IWRITE允许读、写
int creat(char *filename,int permiss) 建立一个新文件filename，并设定读写性。
permiss为文件读写性，可以为以下值
S_IWRITE允许写 S_IREAD允许读 S_IREAD|S_IWRITE允许读、写
int _creat(char *filename,int attrib) 建立一个新文件filename，并设定文件属性。
attrib为文件属性，可以为以下值
FA_RDONLY只读 FA_HIDDEN隐藏 FA_SYSTEM系统
int creatnew(char *filenamt,int attrib) 建立一个新文件filename，并设定文件属性。
attrib为文件属性，可以为以下值
FA_RDONLY只读 FA_HIDDEN隐藏 FA_SYSTEM系统
int creattemp(char *filenamt,int attrib) 建立一个新文件filename，并设定文件属性。
attrib为文件属性，可以为以下值

FA_RDONLY只读 FA_HIDDEN隐藏 FA_SYSTEM系统
int read(int handle,void *buf,int nbyte) 从文件号为handle的文件中读nbyte个字符存入buf中
int _read(int handle,void *buf,int nbyte) 从文件号为handle的文件中读nbyte个字符存入buf中,直接调用MSDOS进行操作.
int write(int handle,void *buf,int nbyte) 将buf中的nbyte个字符写入文件号为handle的文件中
int _write(int handle,void *buf,int nbyte) 将buf中的nbyte个字符写入文件号为handle的文件中
int dup(int handle) 复制一个文件处理指针handle,返回这个指针
int dup2(int handle,int newhandle) 复制一个文件处理指针handle到newhandle
int eof(int *handle) 检查文件是否结束,结束返回1,否则返回0
long filelength(int handle) 返回文件长度，handle为文件号
int setmode(int handle,unsigned mode)本函数用来设定文件号为handle的文件的打开方式
int getftime(int handle,struct ftime *ftime)
读取文件号为handle的文件的时间，并将文件时间存于ftime结构中，成功返回0, ftime结构如下：

```
struct ftime
{
 unsigned ft_tsec:5; /*秒*/
 unsigned ft_min:6; /*分*/
 unsigned ft_hour:5; /*时*/
 unsigned ft_day:5; /*日*/
 unsigned ft_month:4;/*月*/
 unsigned ft_year:1; /*年-1980*/
}
```

int setftime(int handle,struct ftime *ftime) 重写文件号为handle的文件时间,
新时间在结构ftime中.成功返回0.结构ftime如下:

```
struct ftime
{
 unsigned ft_tsec:5; /*秒*/
 unsigned ft_min:6; /*分*/
 unsigned ft_hour:5; /*时*/
 unsigned ft_day:5; /*日*/
 unsigned ft_month:4;/*月*/
 unsigned ft_year:1; /*年-1980*/
}
```

long lseek(int handle,long offset,int fromwhere)
本函数将文件号为handle的文件的指针移到fromwhere后的第offset个字节处.
SEEK_SET文件开关 SEEK_CUR当前位置 SEEK_END文件尾
long tell(int handle) 本函数返回文件号为handle的文件指针,以字节表示
int isatty(int handle)本函数用来取设备handle的类型
int lock(int handle,long offset,long length) 对文件共享作封锁
int unlock(int handle,long offset,long length) 打开对文件共享的封锁

int close(int handle) 关闭handle所表示的文件处理,handle是从_creat、creat、

creatnew、creattemp、dup、dup2、_open、open中的一个处调用获得的文件处理
成功返回0否则返回-1,可用于UNIX系统
int _close(int handle) 关闭handle所表示的文件处理,handle是从_creat、creat、
creatnew、creattemp、dup、dup2、_open、open中的一个处调用获得的文件处理
成功返回0否则返回-1,只能用于MSDOS系统

FILE *fopen(char *filename,char *type) 打开一个文件filename,打开方式为type，
并返回这个文件指针，type可为以下字符串加上后缀

type	读写性	文本/2进制文件	建新/打开旧文件
r	读	文本	打开旧的文件
w	写	文本	建新文件
a	添加	文本	有就打开无则建新
r+	读/写	不限制	打开
w+	读/写	不限制	建新文件
a+	读/添加	不限制	有就打开无则建新

可加的后缀为t、b。加b表示文件以二进制形式进行操作，t没必要使用
例:

```
#include<stdio.h>
main()
{
 FILE *fp;
 fp=fopen("C:\\WPS\\WPS.EXE","r+b");
```

FILE *fdopen(int ahndle,char *type)
FILE *freopen(char *filename,char *type,FILE *stream)
int getc(FILE *stream) 从流stream中读一个字符，并返回这个字符
int putc(int ch,FILE *stream) 向流stream写入一个字符ch
int getw(FILE *stream) 从流stream读入一个整数，错误返回EOF
int putw(int w,FILE *stream) 向流stream写入一个整数
int ungetc(char c,FILE *stream) 把字符c退回给流stream，下一次读进的字符将是c
int fgetc(FILE *stream) 从流stream处读一个字符，并返回这个字符
int fputc(int ch,FILE *stream) 将字符ch写入流stream中
char *fgets(char *string,int n,FILE *stream)
从流stream中读n个字符存入string中
int fputs(char *string,FILE *stream)将字符串string写入流stream中
int fread(void *ptr,int size,int nitems,FILE *stream)
从流stream中读入nitems个长度为size的字符串存入ptr中
int fwrite(void *ptr,int size,int nitems,FILE *stream)
向流stream中写入nitems个长度为size的字符串,字符串在ptr中
int fscanf(FILE *stream,char *format[,argument,…])
以格式化形式从流stream中读入一个字符串
int vfscanf(FILE *stream,char *format,Valist param)
以格式化形式从流stream中读入一个字符串,参数从Valist param中取得
int fprintf(FILE *stream,char *format[,argument,…])
以格式化形式将一个字符串写给指定的流stream
int vfprintf(FILE *stream,char *format,Valist param)
以格式化形式将一个字符串写给指定的流stream,参数从Valist param中取

得
int fseek(FILE *stream,long offset,int fromwhere)
函数把文件指针移到fromwhere所指位置的向后offset个字节处,fromwhere可以为以下值:
SEEK_SET 文件开关 SEEK_CUR 当前位置 SEEK_END 文件尾
long ftell(FILE *stream) 函数返回定位在stream中的当前文件指针位置,以字节表示
int rewind(FILE *stream) 将当前文件指针stream移到文件开头
int feof(FILE *stream) 检测流stream上的文件指针是否在结束位置
int fileno(FILE *stream) 取流stream上的文件处理，并返回文件处理
int ferror(FILE *stream) 检测流stream上是否有读写错误，如有错误就返回1
void clearerr(FILE *stream) 清除流stream上的读写错误
void setbuf(FILE *stream,char *buf) 给流stream指定一个缓冲区buf
void setvbuf(FILE *stream,char *buf,int type,unsigned size)
给流stream指定一个缓冲区buf,大小为size,类型为type,type的值见下表

type值	意义
_IOFBF	文件是完全缓冲区,当缓冲区是空时,下一个输入操作将企图填满整个缓冲区.在输出时,在把任何数据写到文件之前,将完全填充缓冲区.
_IOLBF	文件是行缓冲区.当缓冲区为空时,下一个输入操作将仍然企图填整个缓冲区.然而在输出时,每当新行符写到文件,缓冲区就被清洗掉.
_IONBF	文件是无缓冲的.buf和size参数是被忽略的.每个输入操作将直接从文件读,每个输出操作将立即把数据写到文件中.

int fclose(FILE *stream) 关闭一个流，可以是文件或设备(例如LPT1)
int fcloseall() 关闭所有除stdin或stdout外的流
int fflush(FILE *stream)
关闭一个流，并对缓冲区作处理处理即对读的流，将流内内容读入缓冲区；对写的流，将缓冲区内内容写入流。成功返回0
int fflushall()
关闭所有流，并对流各自的缓冲区作处理处理即对读的流，将流内内容读入缓冲区；对写的流，将缓冲区内内容写入流。成功返回0
int access(char *filename,int amode)
本函数检查文件filename并返回文件的属性, 函数将属性存于amode中，amode由以下位的组合构成
06可以读、写 04可以读 02可以写 01执行(忽略的) 00文件存在
如果filename是一个目录,函数将只确定目录是否存在函数执行成功返回0,否则返回-1
int chmod(char *filename,int permiss) 本函数用于设定文件filename的属性
permiss可以为以下值
S_IWRITE允许写 S_IREAD允许读 S_IREAD|S_IWRITE允许读、写
int _chmod(char *filename,int func[,int attrib]);
本函数用于读取或设定文件filename的属性，
当func=0时，函数返回文件的属性；当func=1时，函数设定文件的属性
若为设定文件属性，attrib可以为下列常数之一

FA_RDONLY只读 FA_HIDDEN隐藏 FA_SYSTEM系统

接口子程序,所在函数库为:dos.h、bios.h
unsigned sleep(unsigned seconds) 暂停seconds微秒(百分之一秒)
int unlink(char *filename) 删除文件filename
unsigned FP_OFF(void far *farptr) 本函数用来取远指针farptr的偏移量
unsigned FP_SEG(void far *farptr) 本函数用来没置远指针farptr的段值
void far *MK_FP(unsigned seg,unsigned off)根据段seg和偏移量off构造一个far指针
unsigned getpsp() 取程序段前缀的段地址,并返回这个地址
char *parsfnm(char *cmdline,struct fcb *fcbptr,int option)
函数分析一个字符串,通常,对一个文件名来说,是由cmdline所指的一个命令行.
文件名是放入一个FCB中作为一个驱动器,文件名和扩展名.FCB是由fcbptr所指定的.
option参数是DOS分析系统调用时,AL文本的值.
int absread(int drive,int nsects,int sectno,void *buffer)
本函数功能为读特定的磁盘扇区,
drive为驱动器号(0=A,1=B等),nsects为要读的扇区数,sectno为开始的逻辑扇区号,buffer为保存所读数据的保存空间
int abswrite(int drive,int nsects,int sectno,void *buffer)
本函数功能为写特定的磁盘扇区,
drive为驱动器号(0=A,1=B等),nsects为要写的扇区数,sectno为开始的逻辑扇区号,buffer为保存所写数据的所在空间
void getdfree(int drive,struct dfree *dfreep)
本函数用来取磁盘的自由空间,
drive为磁盘号(0=当前,1=A等).函数将磁盘特性的由dfreep指向的dfree结构中. dfree结构如下:

```
┌──────────────────────────────────┐
| struct dfree  |
| {  |
|  unsigned df_avail; /*有用簇个数*/  |
|  unsigned df_total; /*总共簇个数*/  |
|  unsigned df_bsec; /*每个扇区字节数*/ |
|  unsigned df_sclus; /*每个簇扇区数*/  |
| }  |
└──────────────────────────────────┘
```

char far *getdta() 取磁盘转换地址DTA
void setdta(char far *dta) 设置磁盘转换地址DTA
void getfat(int drive,fatinfo *fatblkp)
本函数返回指定驱动器drive(0=当前,1=A,2=B等)的文件分配表信息并存入结构fatblkp中,结构如下:

```
┌──────────────────────────────────┐
| struct fatinfo  |
| {  |
|  char fi_sclus; /*每个簇扇区数*/  |
|  char fi_fatid; /*文件分配表字节数*/ |
|  int fi_nclus; /*簇的数目*/  |
|  int fi_bysec; /*每个扇区字节数*/  |
| }  |
└──────────────────────────────────┘
```

void getfatd(struct fatinfo *fatblkp) 本函数返回当前驱动器的文件分配表信息, 并存入结构fatblkp中,结构如下:

```
┌──────────────────────────────────┐
| struct fatinfo  |
```

```
|{ |
| char fi_sclus; /*每个簇扇区数*/ |
| char fi_fatid; /*文件分配表字节数*/|
| int fi_nclus; /*簇的数目*/ |
| int fi_bysec; /*每个扇区字节数*/ |
|} |
└──────────────────────────────┘
```

int bdos(int dosfun,unsigned dosdx,unsigned dosal)
本函数对MSDOS系统进行调用, dosdx为寄存器dx的值,dosal为寄存器al的值,dosfun为功能号
int bdosptr(int dosfun,void *argument,unsiigned dosal)
本函数对MSDOS系统进行调用,
argument为寄存器dx的值,dosal为寄存器al的值,dosfun为功能号
int int86(int intr_num,union REGS *inregs,union REGS *outregs)
执行intr_num号中断,用户定义的寄存器值存于结构inregs中, 执行完后将返回的寄存器值存于结构outregs中.
int int86x(int intr_num,union REGS *inregs,union REGS *outregs, struct SREGS *segregs)
执行intr_num号中断,用户定义的寄存器值存于结构inregs中和结构segregs中,执行完后将返回的寄存器值存于结构outregs中.
int intdos(union REGS *inregs,union REGS *outregs)
本函数执行DOS中断0x21来调用一个指定的DOS函数,用户定义的寄存器值存于结构inregs中,执行完后函数将返回的寄存器值存于结构outregs中
int intdosx(union REGS *inregs,union REGS *outregs,struct SREGS *segregs)
本函数执行DOS中断0x21来调用一个指定的DOS函数,用户定义的寄存器值存于结构inregs和segregs中,执行完后函数将返回的寄存器值存于结构outregs中
void intr(int intr_num,struct REGPACK *preg)
本函数中一个备用的8086软件中断接口它能产生一个由参数intr_num指定的8086软件中断.
函数在执行软件中断前, 从结构preg复制用户定义的各寄存器值到各个寄存器.软件中断完成后,
函数将当前各个寄存器的值复制到结构preg中.参数如下:
intr_num 被执行的中断号，preg为保存用户定义的寄存器值的结构,结构如下

```
┌──────────────────────────────────────┐
| struct REGPACK |
|{ |
| unsigned r_ax,r_bx,r_cx,r_dx; |
| unsigned r_bp,r_si,r_di,r_ds,r_es,r_flags; |
|} |
└──────────────────────────────────────┘
```

函数执行完后,将新的寄存器值存于结构preg中
void keep(int status,int size)
以status状态返回MSDOS,但程序仍保留于内存中,所占用空间由size决定.
void ctrlbrk(int (*fptr)()) 设置中断后的对中断的处理程序.
void disable() 禁止发生中断
void enable() 允许发生中断
void geninterrupt(int intr_num) 执行由intr_num所指定的软件中断
void interrupt(* getvect(int intr_num))()
返回中断号为intr_num的中断处理程序, 例如: old_int_10h=getvect(0x10);
void setvect(int intr_num,void interrupt(* isr)())
设置中断号为intr_num的中断处理程序为isr,例如: setvect(0x10,new_int

_10h);
void harderr(int (*fptr)())
定义一个硬件错误处理程序, 每当出现错误时就调用fptr所指的程序
void hardresume(int rescode) 硬件错误处理函数
void hardretn(int errcode) 硬件错误处理函数
int inport(int prot) 从指定的输入端口读入一个字,并返回这个字
int inportb(int port) 从指定的输入端口读入一个字节,并返回这个字节
void outport(int port,int word) 将字word写入指定的输出端口port
void outportb(int port,char byte) 将字节byte写入指定的输出端口port
int peek(int segment,unsigned offset)
函数返回segment:offset处的一个字
char peekb(int segment,unsigned offset)
函数返回segment:offset处的一个字节
void poke(int segment,int offset,char value)
将字value写到segment:offset处
void pokeb(int segment,int offset,int value)
将字节value写到segment:offset处
int randbrd(struct fcb *fcbptr,int reccnt)
函数利用打开fcbptr所指的FCB读reccnt个记录.
int randbwr(struct fcb *fcbptr,int reccnt)
函数将fcbptr所指的FCB中的reccnt个记录写到磁盘上
void segread(struct SREGS *segtbl)函数把段寄存器的当前值放进结构s
egtbl中
int getverify() 取检验标志的当前状态(0=检验关闭,1=检验打开)
void setverify(int value)
设置当前检验状态, value为0表示关闭检验,为1表示打开检验
int getcbrk()本函数返回控制中断检测的当前设置
int setcbrk(int value)本函数用来设置控制中断检测为接通或断开当value=
0时,为断开检测.当value=1时,为接开检测
int dosexterr(struct DOSERR *eblkp)
取扩展错误.在DOS出现错误后,此函数将扩充的错误信息填入eblkp所指的D
OSERR结构中.该结构定义如下:

```
struct DOSERR
{
 int exterror;/*扩展错误*/
 char class; /*错误类型*/
 char action; /*方式*/
 char locus; /*错误场所*/
}
```

int bioscom(int cmd,char type,int port) 本函数负责对数据的通讯工作,
cmd可以为以下值:
0 置通讯参数为字节byte值 1 发送字符通过通讯线输出
2 从通讯线接受字符 3 返回通讯的当前状态
port为通讯端口,port=0时通讯端口为COM1,port=1时通讯端口为COM2,以
此类推
byte为传送或接收数据时的参数,为以下位的组合:

byte值	意义	byte值	意义	byte值	意义
0x02	7数据位	0x03	8数据位	0x00	1停止位
0x04	2停止位	0x00	无奇偶性	0x08	奇数奇偶性
0x18	偶数奇偶性	0x00	110波特	0x20	150波特
0x40	300波特	0x60	600波特	0x80	1200波特

0xA0	2400波特	0xC0	4800波特	0xE0	9600波特

例如:0xE0|0x08|0x00|0x03即表示置通讯口为9600波特,奇数奇偶性,1停止位,
8数据位. 函数返回值为一个16位整数,定义如下:
第15位 超时
第14位 传送移位寄存器空
第13位 传送固定寄存器空
第12位 中断检测
第11位 帧错误
第10位 奇偶错误
第 9位 过载运行错误
第 8位 数据就绪
第 7位 接收线信号检测
第 6位 环形指示器
第 5位 数据设置就绪
第 4位 清除发送
第 3位 δ接收线信号检测器
第 2位 下降边环形检测器
第 1位 δ数据设置就绪
第 0位 δ清除发送

int biosdisk(int cmd,int drive,int head,int track, int sector,int nsects,void *buffer)
本函数用来对驱动器作一定的操作,cmd为功能号, drive为驱动器号(0=A,1=B,0x80=C,0x81=D,0x82=E等).cmd可为以下值:

0 重置软磁盘系统.这强迫驱动器控制器来执行硬复位.忽略所有其它参数.
1 返回最后的硬盘操作状态.忽略所有其它参数
2 读一个或多个磁盘扇区到内存.读开始的扇区由head、track、sector给出。扇区号由nsects给出。把每个扇区512个字节的数据读入buffer
3 从内存读数据写到一个或多个扇区。写开始的扇区由head、track、sector给出。扇区号由nsects给出。所写数据在buffer中，每扇区512个字节。
4 检验一个或多个扇区。开始扇区由head、track、sector给出。扇区号由nsects给出。
5 格式化一个磁道，该磁道由head和track给出。buffer指向写在指定track上的扇区磁头器的一个表。以下cmd值只允许用于XT或AT微机：
6 格式化一个磁道，并置坏扇区标志。
7 格式化指定磁道上的驱动器开头。
8 返回当前驱动器参数，驱动器信息返回写在buffer中(以四个字节表示)。
9 初始化一对驱动器特性。
10 执行一个长的读，每个扇区读512加4个额外字节
11 执行一个长的写，每个扇区写512加4个额外字节
12 执行一个磁盘查找
13 交替磁盘复位
14 读扇区缓冲区
15 写扇区缓冲区
16 检查指定的驱动器是否就绪
17 复核驱动器
18 控制器RAM诊断
19 驱动器诊断
20 控制器内部诊

函数返回由下列位组合成的状态字节：

0x00 操作成功
0x01 坏的命令
0x02 地址标记找不到
0x04 记录找不到
0x05 重置失败
0x07 驱动参数活动失败
0x09 企图DMA经过64K界限
0x0B 检查坏的磁盘标记
0x10 坏的ECC在磁盘上读
0x11 ECC校正的数据错误（注意它不是错误）
0x20 控制器失效
0x40 查找失败
0x80 响应的连接失败
0xBB 出现无定义错误
0xFF 读出操作失败

int biodquip()
检查设备，函数返回一字节，该字节每一位表示一个信息，如下：
第15位 打印机号
第14位 打印机号
第13位 未使用
第12位 连接游戏I/O
第11位 RS232端口号
第 8位 未使用
第 7位 软磁盘号
第 6位 软磁盘号,
00为1号驱动器,01为2号驱动器,10为3号驱动器,11为4号驱动器

第 5位 初始化
第 4位 显示器模式
00为未使用，01为40x25BW彩色显示卡
10为80x25BW彩色显示卡，11为80x25BW单色显示卡
第 3位 母扦件
第 2位 随机存贮器容量,00为16K,01为32K,10为48K,11为64K
第 1位 浮点共用处理器
第 0位 从软磁盘引导

int bioskey(int cmd)本函数用来执行各种键盘操作，由cmd确定操作。
cmd可为以下值：
0 返回敲键盘上的下一个键。若低8位为非0,即为ASCII字符；若低8位为0,
则返回扩充了的键盘代码。
1 测试键盘是否可用于读。返回0表示没有键可用；否则返回下一次敲键之
值。
敲键本身一直保持由下次调用具的cmd值为0的bioskey所返回的值。
2 返回当前的键盘状态，由返回整数的每一个位表示，见下表：

位	为0时意义	为1时意义
7	插入状态	改写状态
6	大写状态	小写状态
5	数字状态，NumLock灯亮	光标状态，NumLock灯熄
4	ScrollLock灯亮	ScrollLock灯熄
3	Alt按下	Alt未按下
2	Ctrl按下	Ctrl未按下
1	左Shift按下	左Shift未按下

0	右Shift按下	右Shift未按下

int biosmemory() 返回内存大小,以K为单位.
int biosprint(int cmd,int byte,int port) 控制打印机的输入/输出.
port为打印机号,0为LPT1,1为LPT2,2为LPT3等
cmd可以为以下值:
0 打印字符,将字符byte送到打印机
1 打印机端口初始化
2 读打印机状态
函数返回值由以下位值组成表示当前打印机状态
0x01 设备时间超时
0x08 输入/输出错误
0x10 选择的
0x20 走纸
0x40 认可
0x80 不忙碌

int biostime(int cmd,long newtime)计时器控制,cmd为功能号,可为以下值

0 函数返回计时器的当前值
1 将计时器设为新值newtime

struct country *country(int countrycmode,struct country *country p)
本函数用来控制某一国家的相关信息,如日期,时间,货币等.
若countryp=-1时,当前的国家置为countrycode值(必须为非0).否则,由countryp所指向的country结构用下列的国家相关信息填充:
(1)当前的国家(若countrycode为0或2)由countrycode所给定的国家.
结构country定义如下:

```
struct country
{
 int co_date; /*日期格式*/
 char co_curr[5]; /*货币符号*/
 char co_thsep[2]; /*数字分隔符*/
 char co_desep[2]; /*小数点*/
 char co_dtsep[2]; /*日期分隔符*/
 char co_tmsep[2]; /*时间分隔符*/
 char co_currstyle; /*货币形式*/
 char co_digits; /*有效数字*/
 int (far *co_case)(); /*事件处理函数*/
 char co_dasep; /*数据分隔符*/
 char co_fill[10]; /*补充字符*/
} 
```

co_date的值所代表的日期格式是:
0 月日年 1 日月年 2 年月日
co_currstrle的值所代表的货币显示方式是
0 货币符号在数值前,中间无空格
1 货币符号在数值后,中间无空格
2 货币符号在数值前,中间有空格
3 货币符号在数值后,中间有空格

操作函数,所在函数库为string.h、mem.h

mem…操作存贮数组
void *memccpy(void *destin,void *source,unsigned char ch,unsign
ed n)
void *memchr(void *s,char ch,unsigned n)
void *memcmp(void *s1,void *s2,unsigned n)
int memicmp(void *s1,void *s2,unsigned n)
void *memmove(void *destin,void *source,unsigned n)
void *memcpy(void *destin,void *source,unsigned n)
void *memset(void *s,char ch,unsigned n)

这些函数,mem…系列的所有成员均操作存贮数组.在所有这些函数中,数组是
n字节长.
memcpy从source复制一个n字节的块到destin.如果源块和目标块重迭,则选
择复制方向, 以例正确地复制覆盖的字节.
memmove与memcpy相同. memset将s的所有字节置于字节ch中.s数组的
长度由n给出.
memcmp比较正好是n字节长的两个字符串s1和s2.些函数按无符号字符比较
字节,因此,
memcmp("0xFF","\x7F",1)返回值大于0. memicmp比较s1和s2的前n个
字节,不管字符大写或小写.
memccpy从source复制字节到destin.复制一结束就发生下列任一情况:
(1)字符ch首选复制到destin.
(2)n个字节已复制到destin.
memchr对字符ch检索s数组的前n个字节.
返回值:memmove和memcpy返回destin
memset返回s的值
memcmp和memicmp—┬—若s1<s2返回值小于0
├—若s1=s2返回值等于0
└—若s1>s2返回值大于0
memccpy若复制了ch,则返回直接跟随ch的在destin中的字节的一个指针;

否则返回NULL
memchr返回在s中首先出现ch的一个指针;如果在s数组中不出现ch,就返回N
ULL.

void movedata(int segsrc,int offsrc, int segdest,int offdest, unsigne
d numbytes)
本函数将源地址(segsrc:offsrc)处的numbytes个字节复制到目标地址(seg
dest:offdest)
void movemem(void *source,void *destin,unsigned len)
本函数从source处复制一块长len字节的数据到destin.若源地址和目标地址
字符串重迭,则选择复制方向,以便正确的复制数据.
void setmem(void *addr,int len,char value)
本函数把addr所指的块的第一个字节置于字节value中.

str…字符串操作函数
char stpcpy(char *dest,const char *src) 将字符串src复制到dest
char strcat(char *dest,const char *src) 将字符串src添加到dest末尾
char strchr(const char *s,int c) 检索并返回字符c在字符串s中第一次出现
的位置
int strcmp(const char *s1,const char *s2) 比较字符串s1与s2的大小,
并返回s1-s2
char strcpy(char *dest,const char *src) 将字符串src复制到dest
size_t strcspn(const char *s1,const char *s2) 扫描s1,返回在s1中有,
在s2中也有的字符个数

char strdup(const char *s) 将字符串s复制到最近建立的单元
int stricmp(const char *s1,const char *s2) 比较字符串s1和s2,并返回s1-s2
size_t strlen(const char *s) 返回字符串s的长度
char strlwr(char *s)
将字符串s中的大写字母全部转换成小写字母,并返回转换后的字符串
char strncat(char *dest,const char *src,size_t maxlen)
将字符串src中最多maxlen个字符复制到字符串dest中
int strncmp(const char *s1,const char *s2,size_t maxlen)
比较字符串s1与s2中的前maxlen个字符
char strncpy(char *dest,const char *src,size_t maxlen)
复制src中的前maxlen个字符到dest中
int strnicmp(const char *s1,const char *s2,size_t maxlen)
比较字符串s1与s2中的前maxlen个字符
char strnset(char *s,int ch,size_t n)
将字符串s的前n个字符置于ch中
char strpbrk(const char *s1,const char *s2)
扫描字符串s1,并返回在s1和s2中均有的字符个数
char strrchr(const char *s,int c)
扫描最后出现一个给定字符c的一个字符串s
char strrev(char *s)
将字符串s中的字符全部颠倒顺序重新排列,并返回排列后的字符串
char strset(char *s,int ch)
将一个字符串s中的所有字符置于一个给定的字符ch
size_t strspn(const char *s1,const char *s2)
扫描字符串s1,并返回在s1和s2中均有的字符个数
char strstr(const char *s1,const char *s2)
扫描字符串s2,并返回第一次出现s1的位置
char strtok(char *s1,const char *s2)
检索字符串s1,该字符串s1是由字符串s2中定义的定界符所分隔
char strupr(char *s)
将字符串s中的小写字母全部转换成大写字母,并返回转换后的字符串

存贮分配子程序,所在函数库为dos.h、alloc.h、malloc.h、stdlib.h、process.h
int allocmem(unsigned size,unsigned *seg)
利用DOS分配空闲的内存, size为分配内存大小,seg为分配后的内存指针
int freemem(unsigned seg)
释放先前由allocmem分配的内存,seg为指定的内存指针
int setblock(int seg,int newsize)
本函数用来修改所分配的内存长度, seg为已分配内存的内存指针,newsize为新的长度
int brk(void *endds)
本函数用来改变分配给调用程序的数据段的空间数量,新的空间结束地址为endds
char *sbrk(int incr)
本函数用来增加分配给调用程序的数据段的空间数量,增加incr个字节的空间
unsigned long coreleft() 本函数返回未用的存储区的长度,以字节为单位
void *calloc(unsigned nelem,unsigned elsize)
分配nelem个长度为elsize的内存空间并返回所分配内存的指针
void *malloc(unsigned size) 分配size个字节的内存空间,并返回所分配内存的指针
void free(void *ptr) 释放先前所分配的内存,所要释放的内存的指针为ptr
void *realloc(void *ptr,unsigned newsize)
改变已分配内存的大小,ptr为已分配有内存区域的指针,newsize为新的长度,

返回分配好的内存指针.
long farcoreleft() 本函数返回远堆中未用的存储区的长度,以字节为单位
void far *farcalloc(unsigned long units,unsigned long unitsz)
从远堆分配units个长度为unitsz的内存空间,并返回所分配内存的指针
void *farmalloc(unsigned long size)
分配size个字节的内存空间, 并返回分配的内存指针
void farfree(void far *block)
释放先前从远堆分配的内存空间, 所要释放的远堆内存的指针为block
void far *farrealloc(void far *block,unsigned long newsize)
改变已分配的远堆内存的大小,block为已分配有内存区域的指针,newzie为新的长度,返回分配好的内存指针

时间日期函数,函数库为time.h、dos.h
在时间日期函数里,主要用到的结构有以下几个:
总时间日期贮存结构tm

```
struct tm
{
  int tm_sec; /*秒,0-59*/
  int tm_min; /*分,0-59*/
  int tm_hour; /*时,0-23*/
  int tm_mday; /*天数,1-31*/
  int tm_mon; /*月数,0-11*/
  int tm_year; /*自1900的年数*/
  int tm_wday; /*自星期日的天数0-6*/
  int tm_yday; /*自1月1日起的天数,0-365*/
  int tm_isdst; /*是否采用夏时制,采用为正数*/
}
```

日期贮存结构date

```
struct date
{
  int da_year; /*自1900的年数*/
  char da_day; /*天数*/
  char da_mon; /*月数 1=Jan*/
}
```

时间贮存结构time

```
struct time
{
  unsigned char ti_min; /*分钟*/
  unsigned char ti_hour; /*小时*/
  unsigned char ti_hund;
  unsigned char ti_sec; /*秒*/
}
```

char *ctime(long *clock)
本函数把clock所指的时间(如由函数time返回的时间)转换成下列格式的字符串:
Mon Nov 21 11:31:54 1983\n\0
char asctime(struct tm *tm)
本函数把指定的tm结构类的时间转换成下列格式的字符串:
Mon Nov 21 11:31:54 1983\n\0

double difftime(time_t time2,time_t time1)
计算结构time2和time1之间的时间差距(以秒为单位)
struct tm *gmtime(long *clock)
本函数把clock所指的时间(如由函数time返回的时间)转换成格林威治时间,并以tm结构形式返回
struct tm *localtime(long *clock)
本函数把clock所指的时间(如函数time返回的时间)转换成当地标准时间,并以tm结构形式返回
void tzset()本函数提供了对UNIX操作系统的兼容性
long dostounix(struct date *dateptr,struct time *timeptr)
本函数将dateptr所指的日期,timeptr所指的时间转换成UNIX格式, 并返回自格林威治时间1970年1月1日凌晨起到现在的秒数
void unixtodos(long utime,struct date *dateptr,struct time *timeptr)
本函数将自格林威治时间1970年1月1日凌晨起到现在的秒数utime转换成DOS格式并保存于用户所指的结构dateptr和timeptr中
void getdate(struct date *dateblk)
本函数将计算机内的日期写入结构dateblk中以供用户使用
void setdate(struct date *dateblk)
本函数将计算机内的日期改成由结构dateblk所指定的日期
void gettime(struct time *timep)
本函数将计算机内的时间写入结构timep中, 以供用户使用
void settime(struct time *timep)
本函数将计算机内的时间改为由结构timep所指的时间
long time(long *tloc)
本函数给出自格林威治时间1970年1月1日凌晨至现在所经过的秒数,并将该值存于tloc所指的单元中.
int stime(long *tp)本函数将tp所指的时间(例如由time所返回的时间)写入计算机中.

.

发表评论

#1楼 2008-05-13 16:50 | 蒋振凯 [未注册用户] 回复 引用

兴奋啊！终于找到了这么多库函数！

Email (仅博主可见)

验证码 * 4995 看不清,换一张

内容(请不要发表任何与政治相关的内容)

网站首页
新闻频道
.NET频道
社区
博问
网摘
招聘
找找看

Remember Me?

提交 登录 使用高级评论 新用户注册 返回页首 恢复上次提交

[使用Ctrl+Enter键可以直接提交]

ASP.NET MVC专题发布
.NET频道上线测试中
微软2009开发者大赛正在进行
找找看

相关文章:
如何编写一个C#程序
如何配置C#命令行编译器
数据库连接字符串大全
C#下如何实现服务器+客户端的聊天程序
关于C++/CLI的困惑
C++自学，入门该读哪些书？
发现笔试考C++的多，谁能推荐两天C++的电子书
建议把C++博客也整合到一块来

相关链接:

所属分类的其他文章:
C语言函数大全(w开头)--资料收集
C语言函数大全(v开头)--资料收集
C语言函数大全(u开头)--资料收集
C语言函数大全(t开头)--资料收集
C语言函数大全(s开头) (2)--资料收集
C语言函数大全(s开头) (1)--资料收集
C语言函数大全(q,r开头)--资料收集
C语言函数大全(p开头)--资料收集
C语言函数大全(n ,o开头)--资料收集
C语言函数大全(l开头)--资料收集

最新IT新闻:
传微软将投入1亿美元宣传Kumo品牌
IBM收购Sun或下周公布 价格调至每股9到10美元
iPhone已成为游戏平台 威胁索尼任天堂

IE8"开发人员工具"使用详解下
ASP.NET MVC Training Kit发布了