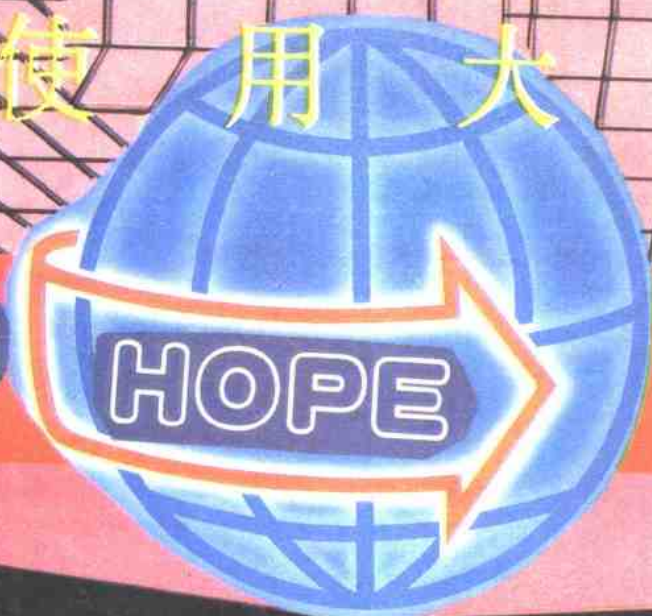


最新Microsoft QuickC

2.50版

使用大全



(共四册)

周杉 王忠 编译
晓夏 武仁

希望出版社

2

66

3

中国科学院希望高级电脑技术公司

最新Microsoft QuickC 使 用 大 全 (V2.50版)

- 安装和运行·成套工具 (之一)
- 语言参考手册 (之二)
- 库函数集锦 (之三)
- 库函数集锦 (之四)

中国科学院希望高级电脑技术公司
一九九一年元月

引 言

《Microsoft QuickC(2.5 版)使用大全》的第三、四册全面系统地介绍了包含在 2.5 版软件包中的各个库函数,按字母顺序列出了它们的使用法、功能、原型所在头文件、返回信息、相关函数等等内容,并提供了经过精心编制的多个示例程序,读者在分析示例的过程中,可以学到大量的编程知识和技巧,从而提高自己的程序设计水平。无论对于 QuickC 的初学者还是高级程序员,这两本书都是必不可少的工具书!

由于第三、第四册主要介绍库函数的内容,所以它们的页码连续编号,书中所有函数均按字母顺序排列。

第四册的最后还列出了 QuickC2.5 中的全局变量和内容说明。

祝您成功!

目 录

库函数参考

abort	1
abs	3
access	4
acos	6
acosl	6
alloca	9
_arc	9
asctime	11
asin	11
assert	11
atan,atan2	13
atof	14
atoi	16
atol,_atold	16
_bcalloc	16
bdos	18
Bessel	19
_bexpand	20
_freseg	21
_bheapadd	23
_bheapchk	23
_bheapmin	23
_bheapseg	23
_bheapset	24
_bheapwalk	24
_bios_disk	25
_bios_equiplist	27
_bios_keybrd	29
_bios_memsiz	31
_bios_serialcom	32
_bios_timeofday	33
_bmalloc	35

_bmsize	35
bsearch	39
cabs	42
cabsl	43
calloc	43
ceil	44
ceilf	46
_cexit	46
_c_exit	46
cgets	46
_chain_intr	47
chdir	53
_chdrive	55
chmod	57
chsize	57
_clear87	60
clearerr	62
_clearscreen	63
clock	65
close	65
_control87	68
cos	69
cosh	71
coshl	71
cosl	72
cprintf	72
cputs	73
creat	75
scanf	76
ctime	77
diecetomsgbin	80
difftime	81
_disable	83

_displaycursor	83	execle	119
div	85	execlp	119
dmsbintoieee	85	execlpe	119
_dos_allocmem	85	execv	120
_dos_close	87	execve	121
_dos_creat	89	execvp	121
_dos_creatnew	90	execvpe	122
dosexterr	90	exit	122
_dos_findfirst	92	exp	122
_dos_findnext	95	_expand	124
_dos_freemem	95	expl	125
_dos_getdate	96	fabs	125
_dos_getdiskfree	98	fclose	126
_dos_getdrive	98	fcloseall	128
_dos_getfileattr	99	fcvt	128
_dos_getftime	101	fdopen	128
_dos_gettime	102	feof	130
_dos_getvect	102	ferror	131
_dos_keep	104	fexpand	131
_dos_open	105	fflush	131
_dos_read	105	_ffree	133
_dos_setblock	106	fgetc,fgetchar	134
_dos_setdrive	106	fgetchar	136
_dos_setfileattr	106	fgetpos	136
_dos_setftime	107	fgets	139
_dos_settime	107	_fheapchk	139
_dos_setvect	108	_fheapmin	139
_dos_write	108	_fheapset	141
dup	109	_fheapwalk	142
dup2	110	fiectomsbin	142
ecvt	110	filelength	142
_ellipse	112	fileno	144
_emit	114	_floodfill	146
_enable	115	floor	148
eof	115	floorl	148
execl	117	flushall	148

<code>_fmemccpy</code>	149	<code>_fstrlen</code>	181
<code>_fmemchr</code>	151	<code>_fstrlwr</code>	181
<code>_fmemcmp</code>	153	<code>_fstrncat</code>	182
<code>_fmemcpy</code>	155	<code>_fstrncmp</code>	184
<code>_fmemicmp</code>	155	<code>fstrncpy</code>	184
<code>_fmemmove</code>	155	<code>_fstrniemp</code>	185
<code>_fmemset</code>	156	<code>_fstrnset</code>	185
<code>fmod</code>	157	<code>_fstrpbrk</code>	185
<code>fmodl</code>	157	<code>_fstrchr</code>	186
<code>fmsbintoieee</code>	157	<code>_fstrrev</code>	186
<code>_fmize</code>	158	<code>_fstrset</code>	186
<code>fopen</code>	158	<code>fstrspn</code>	186
<code>FP_OFF</code>	158	<code>_fstrstr</code>	187
<code>_fpreset</code>	160	<code>_fstrtok</code>	187
<code>fprintf</code>	163	<code>_fstrlwr</code>	187
<code>fputc</code>	165	<code>ftell</code>	187
<code>fputs</code>	165	<code>ftime</code>	188
<code>fread</code>	165	<code>_fullpath</code>	189
<code>_frealloc</code>	168	<code>fwrite</code>	190
<code>free</code>	168	<code>gcvt</code>	190
<code>freect</code>	168	<code>getarcinfo</code>	190
<code>freopen</code>	170	<code>_getbkcolor</code>	191
<code>frexp</code>	170	<code>getc</code>	193
<code>frexpl</code>	171	<code>getch</code>	195
<code>fscanf</code>	171	<code>getchar</code>	196
<code>fseek</code>	171	<code>getche</code>	196
<code>fsetpos</code>	172	<code>_getcolor</code>	197
<code>fsopen</code>	172	<code>_getcurrentposition</code>	201
<code>fstat</code>	175	<code>getcwd</code>	201
<code>_fstreat</code>	177	<code>_getcwd</code>	202
<code>fstrchr</code>	178	<code>_getdrive</code>	202
<code>fstrcmp</code>	178	<code>getenv</code>	202
<code>fstrep</code>	178	<code>_getfillmask</code>	204
<code>_fstrcspp</code>	179	<code>_getfontinfo</code>	204
<code>_fstrdup</code>	180	<code>_gettexttent</code>	207
<code>fstricmp</code>	180	<code>_getgtextvector</code>	207

<code>_getimage</code>	208	<code>isatty</code>	244
<code>_getlinstyle</code>	210	<code>itoa</code>	244
<code>_getphyscoord</code>	210	<code>j0</code>	246
<code>_getpid</code>	210	<code>_j0l</code>	247
<code>_getpixel</code>	211	<code>kbhit</code>	247
<code>gets</code>	212	<code>labs</code>	248
<code>_gettextcolor</code>	213	<code>ldexp</code>	248
<code>_gettextcursor</code>	214	<code>ldexpl</code>	249
<code>_gettextposition</code>	214	<code>ldiv</code>	249
<code>_getvideoconfig</code>	214	<code>lfind</code>	249
<code>_getviewcoord</code>	218	<code>_lineto</code>	250
<code>_getvisualpage</code>	218	<code>localconv</code>	252
<code>getw</code>	218	<code>localtime</code>	252
<code>_getwindowcoord</code>	219	<code>locking</code>	253
<code>_getwritemode</code>	219	<code>log</code>	256
<code>gmtime</code>	221	<code>_makepath</code>	256
<code>_grstaus</code>	222	<code>malloc</code>	257
<code>halloc</code>	225	<code>_matherr</code>	259
<code>_harderr</code>	226	<code>_matherrl</code>	261
<code>_hardresume</code>	231	<code>max</code>	261
<code>_heapadd</code>	231	<code>_memavl</code>	262
<code>_heapchk</code>	232	<code>memccpy</code>	262
<code>_heapmin</code>	232	<code>memchr</code>	262
<code>_heapset</code>	233	<code>memcmp</code>	263
<code>_heapwalk</code>	234	<code>memcpy</code>	263
<code>hfree</code>	234	<code>memicmp</code>	264
<code>hypot</code>	235	<code>_memmax</code>	264
<code>hypotl</code>	236	<code>memmove</code>	265
<code>_imagesize</code>	236	<code>memset</code>	265
<code>inpw</code>	237	<code>max,min</code>	266
<code>inpw</code>	239	<code>mkdir</code>	266
<code>int86</code>	240	<code>mktemp</code>	266
<code>int86x</code>	240	<code>mktime</code>	267
<code>intdos</code>	241	<code>modf</code>	268
<code>intdosx</code>	241	<code>modfl</code>	268
<code>isalnum</code>	242	<code>movedata</code>	268

<code>_moveto</code>	269	<code>_pg_initchart</code>	293
<code>_msize</code>	269	<code>_pg_resetpalette</code>	294
<code>_ncalloc</code>	270	<code>_pg_resetstyleset</code>	294
<code>_nexpand</code>	270	<code>_pg_setchardef</code>	294
<code>_nfree</code>	270	<code>_pg_setpalette</code>	295
<code>_nheapchk</code>	271	<code>_pg_setstyleset</code>	295
<code>_nheapmin</code>	271	<code>_pg_vlabelchart</code>	295
<code>_nheapset</code>	271	<code>_pie</code>	296
<code>_nheapwalk</code>	272	<code>_polygon</code>	296
<code>_nmalloc</code>	272	<code>pow</code>	297
<code>_nrealloc</code>	273	<code>powl</code>	298
<code>_nstrdup</code>	273	<code>printf</code>	298
<code>offset</code>	273	<code>putc</code>	299
<code>onexit</code>	274	<code>putch</code>	300
<code>open</code>	274	<code>putchar</code>	301
<code>outgtext</code>	275	<code>putenv</code>	302
<code>_outmem</code>	275	<code>_putimage</code>	302
<code>outp</code>	276	<code>puts</code>	303
<code>outpw</code>	276	<code>putw</code>	304
<code>_outtext</code>	277	<code>qsort</code>	305
<code>perror</code>	277	<code>raise</code>	305
<code>_pg_analyzechart</code>	278	<code>rand</code>	306
<code>_pg_analyzechartms</code>	280	<code>read</code>	306
<code>_pg_analyzepie</code>	280	<code>realloc</code>	307
<code>_pg_analyzescatter</code>	281	<code>_rectangle</code>	308
<code>_pg_analyzescatterms</code>	281	<code>_registerfonts</code>	309
<code>_pg_chart</code>	282	<code>_remapallpalette</code>	309
<code>_pg_chartms</code>	283	<code>remapalette</code>	313
<code>_pg_chartpie</code>	286	<code>remove</code>	313
<code>_pg_chartscatter</code>	286	<code>rename</code>	314
<code>_pg_chartscatterms</code>	288	<code>rewind</code>	314
<code>_pg_defaultchart</code>	288	<code>rmdir</code>	315
<code>_pg_getchardef</code>	289	<code>rmtmp</code>	315
<code>_pg_getpalette</code>	290	<code>_rotl</code>	315
<code>_pg_getstyleset</code>	293	<code>_rotr</code>	317
<code>_pg_hlabelchart</code>	293	<code>scanf</code>	317

<code>_scrolltextwindow</code>	318	<code>spawnl</code>	353
<code>_searchdev</code>	321	<code>spawnv</code>	357
<code>segread</code>	321	<code>_splitpath</code>	358
<code>_selectpalette</code>	321	<code>sprint</code>	358
<code>_setactivepage</code>	323	<code>sqrt</code>	359
<code>_setbkcolor</code>	326	<code>srand</code>	359
<code>setbuf</code>	326	<code>sscanf</code>	359
<code>_setcliprgn</code>	328	<code>stackavail</code>	360
<code>_setcolor</code>	331	<code>stat</code>	360
<code>_setfillmask</code>	331	<code>_status87</code>	360
<code>_setfont</code>	331	<code>stdin,stdout,stderr,stderr,</code>	
<code>_setgtextvector</code>	333	<code>stdprn</code>	360
<code>setjmp</code>	334	<code>stcreat</code>	361
<code>_setlinestyle</code>	334	<code>strchr</code>	363
<code>setlocale</code>	334	<code>strcmp</code>	363
<code>setmode</code>	335	<code>strcmp</code>	364
<code>_setpixel</code>	337	<code>strcoll</code>	365
<code>_settextcolor</code>	338	<code>strcpy</code>	365
<code>_settextcursor</code>	339	<code>strespn</code>	366
<code>_settextposition</code>	339	<code>_strdate</code>	366
<code>_settextrows</code>	340	<code>strdup</code>	367
<code>_settextwindow</code>	340	<code>strerror</code>	367
<code>setvbuf</code>	341	<code>_strerror</code>	368
<code>_setvideomode</code>	341	<code>strtime</code>	368
<code>_setvideomoderows</code>	346	<code>strcmp</code>	370
<code>_setvieworg</code>	346	<code>strlen</code>	370
<code>_setviewport</code>	347	<code>strlwr</code>	370
<code>_setvisualpage</code>	347	<code>strncat</code>	371
<code>_setwindow</code>	347	<code>strncpy</code>	372
<code>_setwritemode</code>	348	<code>strncpy</code>	372
<code>signal</code>	348	<code>strnicmp</code>	372
<code>sin</code>	350	<code>strnset</code>	373
<code>sinh</code>	351	<code>strpbrk</code>	373
<code>sinhl</code>	351	<code>strrchr</code>	374
<code>sinl</code>	351	<code>strrev</code>	374
<code>sopen</code>	352	<code>strset</code>	374

strspn	375	ungetc	387
strstr	375	ungetch	388
_strtime	375	unlink	389
strtod	375	_unregisterfont	389
strtok	378	utime	390
strtol	378	va...	391
strupr	379	vfprintf	394
strxfrm	379	_wrtapon	396
swab	379	write	397
system	380	y0,y1,yn	398
tan	381	_y0l,_y1l,ynl	398
tanh	381		
tanhf	381	全局变量	
tanl	381	_amblksize	399
tell	381	daylight,timezone,tzname	399
time	381	_doserrno,errno,sys_errlist,	
tmpfile	382	sys_nerr	400
tmpnam	382	environ	401
toasii	384	_fmode	401
tzset	384	_osmajor,_osminor,_osmode,	
ultoi	385	_osversion	401
umask	385	_psp	402

abort

▲包含在: <process.h>, <stdlib.h>

▲用 法: void abort(void);

▲参 见: execl..., execv..., exit, _exit, raise, signal, spawnl..., spawnv...

▲说 明:

abort 函数在 stderr 上输出信息:

abnormal program termination(程序异常中止)

然后, 调用 raise(SIGABRT)。对 SIGABRT 信号所采用的动作取决于前次调用 Signal 函数所作的定义。缺省的 SIGABRT 动作是以退出码 3 结束调用进程, 并将控制返回到父进程或操作系统。

about 函数不消除流缓冲器, 也不执行 atexit 或 onexit 进程。

在多个库中, abort 函数不调用 raise(SIGABRT), 而是简单地以退出代码 3 结束过程。

▲返回值:

abort 函数不将控制权返回给调用者, 而是结束该过程, 并在缺省时将退出代码 3 返回父进程或操作系统。

▲示 例:

/* SIGNAL.C 演示了信号中断例程的建立, 演示的函数包括:

signal abort raise

因为 C 的 I/O 函数在信号例程内不安全, 所以代码根据条件使用了系统级 DOS 设施。另一种方式是设置全局标志, 并在信号处理器之外进行任何 I/O 操作。该技术在 SIGFP.C 中演示。 */

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <signal.h>
```

```
#include <process.h>
```

```
#include <setjmp.h>
```

```
#include <stdlib.h>
```

```
#include <float.h>
```

```
#include <dos.h>
```

```
#include <bios.h>
```

```
void ctrlhandler( void );                  /* Prototypes */
```

```
void safeout( char *str );
```

```
int safein( void );
```

```
void main()
```

```

{
    int ch;
    /* Modify ... behavior. */
    if( signal( SIGINT, ctrlhandler ) == SIG_ERR )
    {
        fprintf( stderr, "Couldn't set SIGINT\n" );
        abort();
    }
    /* Input loop illustrates results. */
    do
    {
        ch = getch();
        if( ch == 0 )
        {
            ch = getch();
            if( ch == 46 )      /* Treat ALT+C like CTRL+C */
                raise( SIGINT );
            else
                printf( "Extended code: %X\n", ch );
        }
        else
            printf( "ASCII code: %X\n", ch );
    } while( ch != 27 );      /* ESC code */
}

/* Handles SIGINT (CTRL+C) interrupt. */
void ctrlhandler()
{
    ch;
    /* Disallow CTRL+C during handler. */
    signal( SIGINT, SIG_IGN );
    safeout( "Abort processing? " );
    ch = safein();
    safeout( "\r\n" );
    if( (ch == 'y') || (ch == 'Y') )
        abort();
    else
        /* The CTRL+C interrupt must be reset to our handler since by
        * default it is reset to the system handler.
        */

```

```

        signal( SIGINT, ctrlhandler );
    }

/* Outputs a string using system level calls. */
void safeout( char *str )
{
    union REGS inregs, outregs;
    inregs.h.ah = 0x0e;
    while( *str )
    {
        inregs.h.al = *str++;
        int86( 0x10, &inregs, &outregs );
    }
}

/* Inputs a character using system level calls. */
int safein()
{
    return ( _bios_keybrd( _KEYBRD_READ ) & 0xff );
}

```

abs

▲包含在: <stdlib.h>, <math.h>

▲用 法: int abs(int n);
 double fabs(double x);
 long labs(long n);
 long double fabsl(long double x);

▲返 回: 返回参数的绝对值。

▲参 见: cabs

▲说 明:

函数 abs, fabs, fabsl 和 labs 返回其参数<x>的绝对值。它们的返回类型如下:

函数	参数/返回值
abs	整数
fabs	浮点值
fabsl	长双精度浮点值
labs	长整数值

▲返回值: 这些函数的返回值如上, 没有错误返回值。

▲示 例:

/* MMATH.C 演示了整型数学函数的用法, 整型函数包括

abs, labs, min, max, div, ldiv.* /

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

void main()
{
    int x, y;
    long lx, ly;
    div_t divres;
    ldiv_t ldivres;
    printf( "Enter two integers: " );
    scanf( "%d %d", &x, &y );
    printf("Function\tResult\n\n" );
    printf( "abs\t\tThe absolute value of %d is %d\n", x, abs( x ) );
    printf( "min\t\tThe lesser of %d and %d is %d\n", x, y, min( x, y ) );
    printf( "max\t\tThe greater of %d and %d is %d\n", x, y, max( x, y ) );
    divres = div( x, y );
    printf( "div\t\tFor %d / %d, quotient is %d and remainder is %d\n\n",
           x, y, divres.quot, divres.rem );
    printf( "Enter two long integers: " );
    scanf( "%ld %ld", &lx, &ly );
    printf("Function\tResult\n\n" );
    ldivres = ldiv( lx, ly );
    printf( "labs\t\tThe absolute value of %ld is %ld\n", lx, labs( lx ) );
    printf( "ldiv\t\tFor %ld / %ld, quotient is %ld and remainder is %ld\n",
           lx, ly, ldivres.quot, ldivres.rem );
}
```

access

▲包含在: <io.h>, <errno.h>

▲用法: int access(char *pathname, int mode);
mode: 00 (existence) 04 (read permission)
 02 (write permission) 06 (read & write permission)

▲返回:

若 <pathname> 的存取方式为 <mode>, 则返回 0; 若文件不存在或存取方式不是 <mode>, 则返回 -1。

▲参 见: chmod, fstat, open, stat

▲说 明:

对于文件来说,函数 `access` 确定指定的文件是否存在,以及是否可以方式 `<mode>` 访问。
`<mode>` 的允许值与在 `access` 调用中的意义如下:

值	意义
00	仅检查是否存在
02	检查写权限
04	检查读权限
06	检查读写权限

对于目录来说,函数 `access` 确定指定的目录是否存在;在 MS-DOS(R)和 OS/2 中,所有目录都是可读写的。

▲返回值:

若文件具有给定的方式,则 `access` 返回 0;若指定的文件不存在或不能以给定的方式进行访问,则返回 -1,并将 `errno` 设置为 `EACCESS` 或 `ENOENT`; `EACCESS` 表示文件的权限不允许以指定的方式访问, `ENOENT` 表示文件不存在。

▲示 例:

/* `CHMOD1.C` 演示了使用下列函数对文件进行读、修改文件属性和修改时间等:

`access` `chmod` `utime`

本程序的功能更强的版本见 `CHMODE2.C`, 它使用 `_dos_` 函数。 */

```
#include <io.h>
#include <sys\types.h>
#include <sys\stat.h>
#include <sys\utime.h>
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

enum FILEATTRIB { EXIST, WRITE = 2, READ = 4, READWRITE = 6 };
/* Macro uses access */
#define EXIST( name ) !access( name, EXIST )

void main( int argc, char *argv[] )
{
    if( !EXIST( argv[1] ) )
    {
        printf( "Syntax: CHMOD1 <filename>" );
        exit( 1 );
    }
    if( !access( argv[1], WRITE ) )
    {
```

```

printf( "File %s is read/write. Change to read only? ", argv[1] );
/* NOTE: Use stdlib.h for function definition of toupper rather
 * than macro version in ctype.h. Macro side effects would cause
 * macro version to read two keys.
 */
if( toupper( getche() ) == 'Y' )
    chmod( argv[1], S_IREAD );
}
else
{
    printf( "File %s is read only. Change to read/write? ", argv[1] );
    if( toupper( getch() ) == 'Y' )
        chmod( argv[1], S_IREAD | S_IWRITE );
}
printf( "\nUpdate file time to current time? " );
if( toupper( getche() ) == 'Y' )
    utime( argv[1], NULL );
exit( 0 );
}

```

acos

- ▲包含在: <math.h>, <errno.h>
- ▲用法:


```
double acos( double x );
double asin( double x );
double atan( double x );
double atan2( double y, double x );
```
- ▲返回:

acos 返回 X 的反余弦, 若 X 大于 1 或小于 -1, 则返回 0。

asin 返回 X 的正弦, 若 X 大于 1 或小于 -1, 则返回 0。

atan 返回 X 的反正切值。

atan2 返回反正切值, 若两个参数皆为 0, 则返回 0。并置 `errno` 为 `EDOM`。
- ▲参见: cos, matherr, sin, tan
- ▲示例: 见 `acosl` 中的 `TRIG.C`。

acosl

- ▲包含在: Include: <math.h>,<errno.h>
- ▲用法:


```
long double acosl( long double x );
long double asinl( long double x );
long double atanl( long double x );
```

`long double atan2l( long double y, long double x );`

▲返 回: `acosl` 返回 X 的反余弦值, 若 X 大于 1 或小于 -1, 则返回 0。
`asinl` 返回 X 的反正弦值。若 X 大于 1 或小于 -1, 则返回 0。
`atanl` 返回 X 的反正切。

`atan2l` 返回反正切值, 若两个参数均为 0, 则返回 0。

▲参 见: `cos`, `cosl`, `matherr`, `_matherrl`, `sin`, `sinl`, `tan`, `tanl`

▲说 明:

函数 `acos`, `asin`, `atan` 和 `atan2` 分别返回 X 的反余弦、反正弦、反正切的值, X 的单位为弧度。`atan2` 使用了两个参数。以上函数的参数及返回值的范围如下。

函数	参数范围	返回值范围
<code>acos</code>	-1到1	0到 Π
<code>asin</code>	-1到1	$-\Pi/2$ 到 $\Pi/2$
<code>atan</code>	无限制	$-\Pi/2$ 到 $\Pi/2$
<code>atan2</code>	无限制	$-\Pi/2$ 到 $\Pi/2$

长双精度函数使用了 80 位长的双精度形式的参数和返回值, 其它方面和以上函数相同。

对于 `acos` 和 `asin`, X 必须在 -1 和 1 之间。若 X 小于 -1 或大于 1, 则 `acos` 和 `asin` 将 `errno` 设置为 `EDOM`, 在错误输出中显示一条 `DOMAIN` 错误信息, 并返回 0。

若 `atan2` 的两个参数皆为 0, 则将 `errno` 设置为 `EDOM`, 在错误输出中显示一条 `DOMAIN` 错误信息, 并返回 0。

▲返回值:

同上。

▲示 例:

/* TRIG.C 演示了下列三角函数的用法:

```
cos      cosh      acos
sin      sinh      asin
tan      tanh      atan
atan2    */
```

```
#include <math.h>
#include <float.h>
#include <stdio.h>
#include <stdlib.h>
```

```
void main()
{
    double x, rx, y;
    do
```