C語言常用標準庫

C語言與C++編程 今天

> **來自公眾號：strongerHuang**
>
> 素材來源：網絡

有很多工程師喜歡自己封裝一些標準庫已有的函數，其實自己封裝的函數，並不一定比標準庫好，有時候反而代碼更冗餘，且有bug。下面就來分享一下C語言常見的一些標準庫。

標準頭文件包括：

```
<asset.h>    <ctype.h>    <errno.h>    <float.h>
<limits.h>   <locale.h>   <math.h>     <setjmp.h>
<signal.h>   <stdarg.h>   <stddef.h>   <stdlib.h>
<stdio.h>    <string.h>   <time.h>
```

一、標准定義（<stddef.h>）

文件<stddef.h>裡包含了標準庫的一些常用定義，無論我們包含哪個標準頭文件，<stddef.h>都會被自動包含進來。

這個文件裡定義：

類型size_t（sizeof運算符的結果類型，是某個無符號整型）；

類型ptrdiff_t（兩個指針相減運算的結果類型，是某個有符號整型）；

類型wchar_t（寬字符類型，是一個整型，其中足以存放本系統所支持的所有本地環境中的字符集的所有編碼值。這裡還保證空字符的編碼值為0）；

符號常量NULL（空指針值）；

宏offsetor（這是一個帶參數的宏，第一個參數應是一個結構類型，第二個參數應是結構成員名。

```
offsetor(s,m)
```

求出成員m在結構類型t的變量裡的偏移量）。

注：其中有些定義也出現在其他頭文件裡（如NULL）。

二、錯誤信息（<errno.h>）

<errno.h>定義了一個int類型的表達式errno，可以看作一個變量，其初始值為0，一些標準庫函數執行中出錯時將它設為非0值，但任何標準庫函數都設置它為0。

<errno.h>裡還定義了兩個宏EDOM和ERANGE，都是非0的整數值。數學函數執行中遇到參數錯誤，就會將errno置為EDOM，如出現值域錯誤就會將errno置為ERANGE。

三、輸入輸出函數（<stdio.h>）

文件打開和關閉：

```
FILE *fopen(const char *filename, const char *mode);
int fclose(FILE * stream);
```

字符輸入輸出：

```
int fgetc(FILE *fp);
int fputc(int c, FILE *fp);
```

getc和putc與這兩個函數類似，但通過宏定義實現。通常有下面定義：

```
#define getchar()  getc(stdin)
#define putchar(c) putc(c, stdout)
int ungetc(int c, FILE* stream);  //把字符 c 退回流 stream
```

格式化輸入輸出：

```
int scanf(const char *format, ...);
int printf(const char *format, ...);
int fscanf(FILE *stream, const char *format, ...);
int fprintf(FILE *stream, const char *format, ...);
int sscanf(char *s, const char *format, ...);
int sprintf(char *s, const char *format, ...);
```

行式輸入輸出：

```
char *fgets(char *buffer, int n, FILE *stream);
int fputs(const char *buffer, FILE *stream);
char *gets(char *s);
int puts(const char *s);
```

直接輸入輸出：

```
size_t fread(void *pointer, size_t size, size_t num, FILE *stream);
size_t fwrite(const void *pointer, size_t size, size_t num, FILE *stream);
```

四、數學函數（<math.h>）

1.三角函數：

三角函數	sin	cos	tan
反三角函數	asin	acos	atan
雙曲函數	sinh	cosh	tanh

2.指數和對數函數：

以*e*為底的指數函數	exp
自然對數函數	log
以*10*為底的對數函數	log10

3.其他函數：

平方根	sqrt
絕對值	fabs
乘冪，第一個參數作為底，第二個是指數	double pow(double, double)
實數的餘數，兩個參數分別是被除數和除數	double fmod(double, double)

注：所有上面未給出類型特徵的函數都取一個參數，其參數與返回值都是double類型。

下面函數返回雙精度值（包括函數ceil和floor）。在下表裡，除其中有特別說明的參數之外，所有函數的其他參數都是double類型。

函數原型	意義解釋
ceil(x)	求出不小於*x*的最小整數（返回與這個整數對應的double值）
floor(x)	求出不大於*x*的最大整數（返回與這個整數對應的double值）
atan2(y, x)	求出 *tan-1(y/x)*，其值的範圍是[- *pai,pai]*
ldexp(x, int n)	求出*x*2n*
frexp(x, int *exp)	把 *x*分解為 *y*2n*， 是位於區間 *[1/2,1)*裡的一個小數，作為函數結果返回，整數*n* 通過指針*exp返回（應提供一個int變量地址）。當*x* 為*0*時這兩個結果的值都是*0*
modf(x, double *ip)	把*x*分解為小數部分和整數部分，小數部分作為函數返回值，整數部分通過指針*ip返回。

五、字符處理函數（<ctype.h>）

見下表：

int isalpha(c)	c是字母字符
int isdigit(c)	c是數字字符
int isalnum(c)	c是字母或數字字符

int isspace(c)	c是空格、製表符、換行符
int isupper(c)	c是大寫字母
int islower(c)	c是小寫字母
int iscntrl(c)	c是控製字符
int isprint(c)	c是可打印字符，包括空格
int isgraph(c)	c是可打印字符，不包括空格
int isxdigit(c)	c是十六進制數字字符
int ispunct(c)	c是標點符號
int tolower(int c)	當c是大寫字母時返回對應小寫字母，否則返回c本身
int toupper(int c)	當c是小寫字母時返回對應大寫字母，否則返回c本身

注：條件成立時這些函數返回非0值。最後兩個轉換函數對於非字母參數返回原字符。

六、字符串函數（<string.h>）

1.字符串函數

所有字符串函數列在下表裡，函數描述採用如下約定：s、t表示(char *)類型的參數，cs、ct表示(const char*)類型的參數（它們都應表示字符串）。n表示size_t類型的參數（size_t是一個無符號的整數類型），c是整型參數（在函數里轉換到char）：

函數原型	意義解釋
size_t strlen(cs)	求出cs的長度
char *strcpy(s,ct)	把ct複製到s。要求s指定足夠大的字符數組
char *strncpy(s,ct,n)	把ct裡的至多n個字符複製到s。要求s指定一個足夠大的字符數組。如果ct裡的字符不夠n個，就在s裡填充空字符。
char *strcat(s,ct)	把ct裡的字符複製到s裡已有的字符串之後。s應指定一個保存著字符串，而且足夠大的字符數組。
char *strncat(s,ct,n)	把ct裡的至多n個字符複製到s裡已有的字符串之後。s應指定一個保存著字符串，而且足夠大的字符數組。
int strcmp(cs,ct)	比較字符串cs和ct的大小，在cs大於、等於、小於ct時分別返回正值、0、負值。
int strncmp(cs,ct,n)	比較字符串cs和ct的大小，至多比較n個字符。在cs大於、等於、小於ct時分別返回正值、0、負值。
char *strchr(cs,c)	在cs中查尋c並返回c第一個出現的位置，用指向這個位置的指針表示。當cs裡沒有c時返回值NULL
char *strrchr(cs,c)	在cs中查尋c並返回c最後一個出現的位置，沒有時返回NULL
size_t strspn(cs,ct)	由cs起確定一段全由ct裡的字符組成的序列，返回其長度
size_t strcspn(cs,ct)	由cs起確定一段全由非ct裡的字符組成的序列，返回其長度
char *strpbrk(cs,ct)	在cs裡查尋ct裡的字符，返回第一個滿足條件的字符出現的位置，沒有時返回NULL
char *strstr(cs,ct)	在cs中查尋串ct（查詢子串），返回ct作為cs的子串的第一個出現

	的位置，ct未出現在cs裡時返回NULL
char *strerror(n)	返回與錯誤編號n相關的錯誤信息串（指向該錯誤信息串的指針）
char *strtok(s,ct)	在s中查尋由ct中的字符作為分隔符而形成的單詞

2.存儲區操作

<string.h>還有一組字符數組操作函數（存儲區操作函數），名字都以mem開頭，以某種高效方式實現。在下面原型中，參數s和t的類型是(void *)，cs和ct的類型是(const void *)，n的類型是size_t，c的類型是int（轉換為unsigned char）。

函數原型	意義解釋
void *memcpy(s,ct,n)	從ct處複製n個字符到s處，返回s
void *memmove(s,ct,n)	從ct處複製n個字符到s處，返回s，這裡的兩個段允許重疊
int memcmp(cs,ct,n)	比較由cs和ct開始的n個字符，返回值定義同strcmp
void *memchr(cs,c,n)	在n個字符的範圍內查尋c在cs中的第一次出現，如果找到，返回該位置的指針值，否則返回NULL
void *memset(s,c,n)	將s的前n個字符設置為c，返回s

七、功能函數（<stdlib.h>）

1.隨機數函數：

函數原型	意義解釋

int rand(void)	生成一個0到RAND_MAX的隨機整數
void srand(unsigned seed)	用seed為隨後的隨機數生成設置種子值

2.動態存儲分配函數:

函數原型	意義解釋
void *calloc(size_t n, size_t size)	分配一塊存儲，其中足以存放n個大小為size的對象，並將所有字節用0字符填充。返回該存儲塊的地址。不能滿足時返回NULL
void *malloc(size_t size)	分配一塊足以存放大小為size的存儲，返回該存儲塊的地址，不能滿足時返回NULL
void *realloc(void *p, size_t size)	將p所指存儲塊調整為大小size，返回新塊的地址。如能滿足要求，新塊的內容與原塊一致；不能滿足要求時返回NULL，此時原塊不變
void free(void *p)	釋放以前分配的動態存儲塊

3.幾個整數函數

幾個簡單的整數函數見下表，div_t和ldiv_t是兩個預定義結構類型，用於存放整除時得到的商和余數。div_t類型的成分是int類型的quot和rem，ldiv_t類型的成分是long類型的quot和rem。

函數原型	意義解釋
int abs(int n)	求整數的絕對值

long labs(long n)	求長整數的絕對值
div_t div(int n, int m)	求n/m，商和余數分別存放到結果結構的對應成員裡
ldiv_t ldiv(long n, long m)	同上，參數為長整數

4.數值轉換

函數原型	意義解釋
double atof(const char *s)	由串s構造一個雙精度值
int atoi(const char *s)	由串s構造一個整數值
long atol(const char *s)	由串s構造一個長整數值

5.執行控制

1）非正常終止函數abort。

原型是：

```
1  void abort(void);
```

2）正常終止函數exit。

原型是：

```
void exit(int status);
```

導致程序按正常方式立即終止。status作為送給執行環境的出口值，0表示成功結束，兩個可用的常數為EXIT_SUCCESS，EXIT_FAILURE。

3）正常終止註冊函數atexit。

原型是：

```
int atexit(void (*fcn)(void))
```

可用本函數把一些函數註冊為結束動作。被註冊函數應當是無參無返回值的函數。註冊正常完成時atexit返回值0，否則返回非零值。

6.與執行環境交互

1）向執行環境傳送命令的函數system。

原型是：

```
int system(const char *s);
```

把串s傳遞給程序的執行環境要求作為系統命令執行。如以NULL為參數調用，函數返回非0表示環境裡有命令解釋器。如果s不是NULL，返回值由實現確定。

2）訪問執行環境的函數getenv。

原型是：

```
char *getenv(const char *s);
```

從執行環境中取回與字符串s相關聯的環境串。如果找不到就返回NULL。本函數的具體結果由實現確定。在許多執行環境裡，可以用這個函數去查看“環境變量”的值。

7.常用函數bsearch和qsort

1）二分法查找函數bsearch：

```
void *bsearch(const void *key, const void *base, size_t n, size_t size, int
```

函數指針參數cmp的實參應是一個與字符串比較函數strcmp類似的函數，確定排序的順序，當第一個參數keyval比第二個參數datum大、相等或小時分別返回正、零或負值。

2）快速排序函數qsort：

qsort對於比較函數cmp的要求與bsearch一樣。設有數組base[0],...,base[n-1]，元素大小為size。用qsort可以把這個數組的元素按cmp確定的上升順序重新排列。

```
void qsort(void *base, size_t n, size_t size, int (*cmp)(const void *, cons
```

●輸入m獲取文章目錄