# C语言笔记

## 文档格式说明

## 函数printf用法

### 1、打印64位整数

|  |
| --- |
| uint64 d = 0x1234567812345678;  printf(“%I64d\r\n”, d); |

## 函数scanf用法

### 1、读取单个字节

|  |
| --- |
| scanf\_s(“%c”, &c);  fflush(stdin) |

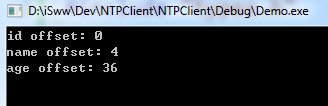
## 宏offsetof用法

 #define offsetof(s, m)   (size\_t)&(((s \*)0)->m)

s是一个结构名，它有一个名为m的成员（s和m 是宏offsetof的形参），它实际是返回结构s的成员m的偏移地址。

|  |
| --- |
| #include <stdio.h>  #include <stddef.h>  #include "types.h"  #define NAME\_STR\_LEN 30  struct student\_info  {  uint32\_t id;  uint8\_t name[NAME\_STR\_LEN];  uint32\_t age;  };  int main()  {  uint16\_t offset = 0;    offset = offsetof(struct student\_info, id);  printf("id offset: %d\r\n", offset);  offset = offsetof(struct student\_info, name);  printf("name offset: %d\r\n", offset);  offset = offsetof(struct student\_info, age);  printf("age offset: %d\r\n", offset);  scanf\_s("%d", &offset); //避免调试后，立即退出。  return 0;  } |

运行结果：



## 时间函数

本地时间（local time）：UTC的8时区

格林威治时间（Greenwich Mean Time GMT）

世界协调时间（Universal Time Coordinated UTC）：UTC的0时区

Unix时间戳：UTC时间1970.1.1 0:00:00起开始计算的秒数。

一天的总秒数为86400秒。

头文件：time.h

时间戳没有时区之分，但是日期有时区之分。

### 1、time函数

原型：time\_t time(time\_t \*t)

说明：返回从公元1970年1月1日的UTC时间从0时0分0秒算起到现在所经过的秒数。如果t 并非空指针的话，此函数也会将返回值存到t指针所指的内存。

示例：

|  |
| --- |
| #include <time.h>  #include <stdio.h>  #include "types.h"  int main()  {  time\_t timestamp;  timestamp = time((time\_t \*)NULL);  printf("Seconds from 1970:1:1 00:00:00 : %d\r\n", (uint32\_t)timestamp);  return 0;  } |

运行结果：



### 2、ctime函数

原型：char \*ctime(const time\_t \*timep)

说明： 将参数timep所指的time\_t结构中的信息转换成真实世界所使用的时间日期表示方法，然后将结果以字符串形态返回。此函数已经由时区转换成当地时间，字符串格式为"Wed Jun 30 21 :49 :08 1993/n"。若再调用相关的时间日期函数，此字符串可能会被破坏。

示例：

|  |
| --- |
| int main()  {  time\_t timestamp;  struct tm now;  char buf[50];  timestamp = time((time\_t \*)NULL);  ctime\_s(buf, sizeof(buf), &timestamp);  printf("%s\r\n", buf);  } |

运行结果：



### 3、asctime函数

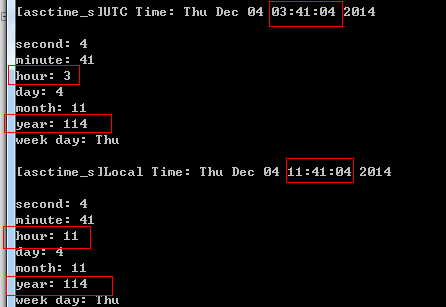
原型：char \*asctime(const struct tm \*timep)

说明：将参数timep所指的tm结构中的信息转换成真实世界所使用的时间日期表示方法，然后将结果以字符串形态返回，字符串格式为:"Wed Jun 30 21:49:08 1993/n"。只是将timep表示的日期以字符串的形式表示。

示例：

|  |
| --- |
| int main()  {  time\_t timestamp;  struct tm now;  char buf[50];  timestamp = time((time\_t \*)NULL);  gmtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]UTC Time: %s\r\n", buf);  print\_tm(&now);  localtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]Local Time: %s\r\n", buf);  print\_tm(&now); } |

运行结果：



### 4、gmtime函数

原型：struct tm \*gmtime(const time\_t \*timep)

说明： 将参数timep 所指的time\_t 结构中的信息转换成真实世界所使用的时间日期表示方法，然后将结果由结构tm返回。返回的是UTC时间。

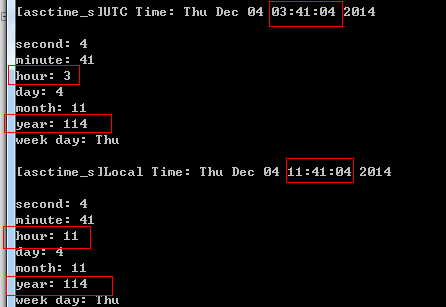
struct tm   
{   
 int tm\_sec;   
 int tm\_min;   
 int tm\_hour;   
 int tm\_mday;   
 int tm\_mon;   
 int tm\_year;   
 int tm\_wday;   
 int tm\_yday;   
 int tm\_isdst;   
};

int tm\_sec： 代表目前秒数，正常范围为0-59，但允许至61秒   
int tm\_min ：代表目前分数，范围0-59   
int tm\_hour ：从午夜算起的时数，范围为0-23   
int tm\_mday ：目前月份的日数，范围01-31   
int tm\_mon： 代表目前月份，从一月算起，范围从0-11   
int tm\_year ：从1900 年算起至今的年数   
int tm\_wday ：一星期的日数，从星期一算起，范围为0-6   
int tm\_yday ：从今年1月1日算起至今的天数，范围为0-365   
int tm\_isdst ：日光节约时间的旗标   
此函数返回的时间日期未经时区转换，而是UTC时间。

示例：

|  |
| --- |
| char \*wday[] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};  void print\_tm(struct tm \*t)  {  printf("second: %d\r\n", t->tm\_sec);  printf("minute: %d\r\n", t->tm\_min);  printf("hour: %d\r\n", t->tm\_hour);  printf("day: %d\r\n", t->tm\_mday);  printf("month: %d\r\n", t->tm\_mon);  printf("year: %d\r\n", t->tm\_year);  printf("week day: %s\r\n\r\n", wday[t->tm\_wday]);  }  int main()  {  time\_t timestamp;  struct tm now;  char buf[50];  timestamp = time((time\_t \*)NULL);  gmtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]UTC Time: %s\r\n", buf);  print\_tm(&now);  localtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]Local Time: %s\r\n", buf);  print\_tm(&now); } |

运行结果：



### 5、localtime函数

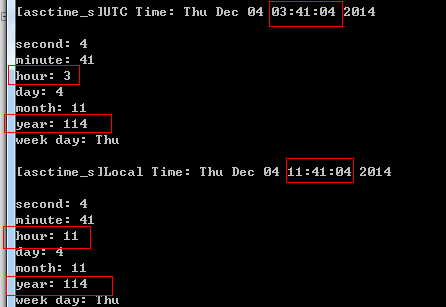
原型：struct tm \*localtime(const time\_t \* timep)

说明：将参数timep所指的time\_t结构中的信息转换成真实世界所使用的时间日期表示方法，然后将结果由结构 tm返回。结构tm的定义请参考gmtime()。此函数返回的时间日期已经转换成当地时区。返回的是当地时间。

示例：

|  |
| --- |
| char \*wday[] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};  void print\_tm(struct tm \*t)  {  printf("second: %d\r\n", t->tm\_sec);  printf("minute: %d\r\n", t->tm\_min);  printf("hour: %d\r\n", t->tm\_hour);  printf("day: %d\r\n", t->tm\_mday);  printf("month: %d\r\n", t->tm\_mon);  printf("year: %d\r\n", t->tm\_year);  printf("week day: %s\r\n\r\n", wday[t->tm\_wday]);  }  int main()  {  time\_t timestamp;  struct tm now;  char buf[50];  timestamp = time((time\_t \*)NULL);  gmtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]UTC Time: %s\r\n", buf);  print\_tm(&now);  localtime\_s(&now, &timestamp);  asctime\_s(buf, sizeof(buf), &now);  printf("[asctime\_s]Local Time: %s\r\n", buf);  print\_tm(&now); } |

运行结果：



### 6、mktime函数

原型：time\_t mktime(struct tm \*timep)

说明：用来将参数timep所指的tm结构数据转换成从公元1970年1月1日0时0分0 秒算起至今的UTC时间所经过的秒数。 timep表示的是当地时间。  
示例：

|  |
| --- |
| int main()  {  time\_t timestamp;  struct tm now;  char buf[50];  timestamp = time((time\_t \*)NULL);  printf("[time]current timestamp: %d\r\n", (uint32\_t)timestamp);  localtime\_s(&now, &timestamp);  now.tm\_sec += 10;  printf("[mktime]timestamp: %d, 10s later\r\n", (uint32\_t)mktime(&now));  gmtime\_s(&now, &timestamp);  now.tm\_sec += 10;  printf("[mktime]timestamp: %d, 10s later\r\n", (uint32\_t)mktime(&now));    return 0;  } |

运行结果：



原型：

说明：

示例：

|  |
| --- |
|  |

运行结果：