

2020九月暑期复习

笔记本: 我的第一个笔记本

创建时间: 2020/9/9 20:09

更新时间: 2020/9/9 21:59

作者: 1272209351@qq.com

练习题:

- 1、32位平台下 strlen ----- 遇到 \0 结束
 sizeof ----- 大小

(1)

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int func( int a[] )
5 {
6     //如果函数func参数传递数组Array, 那么sizeof(a)呢?
7     printf("%d\n", sizeof(a));
8 }
9
10 void main()
11 {
12     char str[] = "Welcome to Bit";
13     int Array[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
14     char *p = str;
15     int n;
16     printf("len = %d\n", strlen(str));
17     printf("size = %d\n", sizeof(str));
18     printf("%d\n", sizeof(Array));
19     printf("%d", sizeof(p));
20     printf("%d\n", sizeof(n));
21
22     func(Array);
23
24 }
```

结果:

14 (遇到 \0 结束)

15 4*9=36 4 4 4

(2)

```
char str1[] = "HelloBit";
printf("len = %d\n", strlen(str1));
printf("size = %d\n", sizeof(str1));

char str2[10] = "HelloBit";
printf("len = %d\n", strlen(str2)); //
printf("size = %d\n", sizeof(str2)); //
```

结果:

8 9 8 10

(3)

```
char str3[10] = {'H', 'e', 'l', 'l', 'o', 'B', 'i', 't'};
printf("len = %d\n", strlen(str3)); //
printf("size = %d\n", sizeof(str3)); //
```

结果:

8 10

(4)

```
char str4[] = {'H', 'e', 'l', 'l', 'o', 'B', 'i', 't'};
printf("len = %d\n", strlen(str4)); //
printf("size = %d\n", sizeof(str4)); //8 I
```

```
char str5[10];
for(int i=0; i<5; ++i)
    str5[i] = 'a' + i;
printf("len = %d\n", strlen(str5)); //
printf("size = %d\n", sizeof(str5)); //
```

结果:

~~24 (随机, 未完全初始化后边赋值 \0)~~ 8
~~44 (随机, 因为未进行初始化找不到 \0)~~ 10

全局数据 && 局部数据 :

未进行初始化时, 系统给出的初始化不一样

全局数据初始化为 \0

局部数据初始化为 随机值

(5)

```
char str7[10];
for(int i=0; i<5; ++i)
    str7[i] = '0'; I
printf("len = %d\n", strlen(str7)); //随机
printf("size = %d\n", sizeof(str7)); //10
```

字符 '0' 不等于 '\0'

'\0' == '0' ; 反斜杠起到转义的作用

char ch='\5' =====存放的是 5

char ch='5' =====存放的是53 ('0' 字符ASCII 码值为 0, 5---ASCII 值为53)

(6)

```
short *par[10][10]; //数组
printf("size = %d\n", sizeof(par)); //400
```

数组类型, 10*10=100个元素, short* ==4 字节

2、运行是否出问题? 指出问题 / 循环次数?

- 2、请观察下列程序运行时是否会出现问题，若有，请指出问题，若不存在问题，请指出程序循环的次数

```
#define MAX_SIZE 255
void main()
{
    unsigned char buff[MAX_SIZE + 1];
    unsigned char i;
    for(i=0; i <= MAX_SIZE; i++)
    {
        buff[i] = i;
    }
}
```

有问题 --- 死循环

原因：变量 i 为无符号char 类型 [0,255] 总小于 MAX_SIZE+1

- 3、添加下面粗斜体部分代码的初衷是为了给 gui show image 这句代码加上限制条件，请问这样修改有什么隐患？该如何修改？

```
.....
gui_push_clip();
#ifdef AAA
    if (show_status == MMI_TRUE)
#endif
#ifdef BBB
    gui_show_image(x, y, image_id);
#endif
    gui_pop_clip();
    update_dt_display();
    .....
```

修改方案：

```
#ifdef BBB
    #ifdef AAA
    if (show_status == MMI_TRUE)
    #endif
    gui_show_image(x, y, image_id);
#endif
    gui_pop_clip();
    update_dt_display();
    .....
```

```

+ .....↵
  gui_push_clip();↵
  #ifdef AAA↵
    if (show_status == MMI_TRUE)↵
  #endif↵
  #ifdef BBB↵
    gui_show_image(x , y , image_id);↵
  #endif↵
  ↵ ; ↵
  gui_pop_clip();↵
  update_display();↵
  .....↵

```

或者加个 ; 号 （不建议）

4、请问代码运行问题？

(1)

```

1 #include<stdio.h>
2 #include<string.h>
3 #include<malloc.h>
4
5 void GetMemory(char *p)
6 {
7     p = (char*)malloc(57);
8 }
9 void main()
10 {
11     char *str = NULL;
12     GetMemory(str);
13     strcpy(str, "HelloBit");
14     printf(str);
15 }
16

```

形参改变不能改变实参，开辟的空间无法使用 ---- 用二级指针

(2)

```

1 char* GetMemory(void)
2 {
3     char p[] = "HelloBit";
4     return p;
5 }
6 void main()
7 {
8     char *str = NULL;
9     str = GetMemory();
10    printf(str);
11 }

```

p 是个临时数组，使用完之后会被释放

(3)

```

4
5 void GetMemory(char **p)
6 {
7     *p = (char *)malloc(57);
8 }
9 void main()
10 {
11     char *str = NULL;
12     GetMemory(&str);
13     strcpy(str, "HelloBit");
14     printf(str);
15 }
16

```

申请成功判断，内存释放
改进：

```

void GetMemory(char **p)
{
    *p = (char *)malloc(57);
    assert(*p != NULL);
}

void main()
{
    char *str = NULL;
    GetMemory(&str);
    strcpy(str, "HelloBit");
    printf(str);

    free(str);
}

```

(4)

```

void main()
{
    char *str = (char *)malloc(57);
    strcpy(str, "Hello");
    free(str);
    if (str != NULL)
    {
        strcpy(str, "C++");
        printf(str);
    }
}

```

str 被释放，成为野指针，空间不能被正常使用因此不可以将 'C++' 写入
改进：

```

void main()
{
    char *str = (char *)malloc(57);
    strcpy(str, "Hello");
    free(str);
    str = NULL; //预防野指针
    if (str != NULL)
    {
        strcpy(str, "C++");
        printf(str);
    }
}

```

5、编程题

5、请编码实现以下功能的函数

功能：实现对一个 8 bit 数据（unsigned char 类型）的指定位（例如第 n 位）的置 0 或者置 1 操作，并保持其他位不变。

函数原型：

void bit_set(unsigned char *p_data, unsigned char position, bool flag)

函数参数说明：

p_data 是指定的源数据，position 是指定位（取值范围 1~8）；flag 表示是置 0 还是置 1 操作，true：置 1 false：置 0

解答：

```

void bit_set(unsigned char *p_data, unsigned char position, bool flag)
{
    if(flag) //1
    {
        *p_data |= (0x01<<(position-1));
    }
    else //0
    {
        *p_data &= ~(0x01<<(position-1));
    }
}

void main()
{
    unsigned char data = 123; //0 ~255 //0111 1011 double
    unsigned char pos = 3;
    bool flag = true;

    bit_set(&data, pos, flag);
}

```

如果未告知取值要考虑边界条件：

```

void bit_set(unsigned char *p_data, unsigned char position, bool flag)
{
    assert(p_data != NULL || (position>=1 && position<=8));

    if(flag) //1
    {
        *p_data |= (0x01<<(position-1));
    }
    else //0
    {
        *p_data &= ~(0x01<<(position-1));
    }
}

```

6、

- 6、请实现字符串右循环移位函数，比如：“abcdefghi”循环右移 2 位就是“hiabcdefg”。
- 函数原型：void RightLoopMove(char *pStr, unsigned short steps)
- 函数参数说明：
- pStr: Point to a ‘\0’ terminated string
- steps: The rotate shift numbers

解答：

法一：将字符串划分为两部分，前边部分拼接到后边字符之后

```
void RightLoopMove(char *pStr, unsigned short steps)
{
    assert(pStr != NULL && *pStr != '\0');
    int len = strlen(pStr);
    char *tmp = (char*)malloc(sizeof(char) * (len+1)); //空间复杂度O(n)
    assert(tmp != NULL);

    steps %= len;

    strcpy(tmp, pStr+(len-steps));
    strncat(tmp, pStr, len-steps);

    strncpy(pStr, tmp, len);
    free(tmp);
    tmp = NULL;
}

void main()
{
    char str[] = "abcdefghi";
    printf("str = %s\n", str);
    RightLoopMove(str, 20);
    printf("str = %s\n", str);
}
```

法二：将最后一个字符提前保存在临时空间变量，其余字符整体后移

```
void RightLoopMove(char *pStr, unsigned short steps)
{
    assert(pStr != NULL && *pStr != '\0');
    int len = strlen(pStr);
    steps %= len;
    for(int i=0; i<steps; ++i)
    {
        char tmp = pStr[len-1]; //空间复杂度O(1)
        for(int end=len-1; end>0; --end) //时间复杂度O(n^2)
            pStr[end] = pStr[end-1];
        pStr[0] = tmp;
    }
}

void main()
{
    char str[] = "abcdefghi";
    printf("str = %s\n", str);
    RightLoopMove(str, 40);
    printf("str = %s\n", str);
}
```

