C 语言的深度挖掘 (一)

Warm-up

是否入门?

```
void  fun( char *t,  char *s )
{
    while (*t++ = *s++);
}
```

改进

```
char *strcpy(char *t, const char *s)
{
    assert( t!=NULL && s!=NULL);
    char *temp = t;
    while(*t++ = *s++)
            NULL;
    return temp;
}
```

有错吗?

```
#include <stdio.h>
#include <stdlib.h>
void getmemory(char *p)
{
    p = (char *) malloc(100);
    strcpy(p, "hello world");
}
int main( )
{
    char *str = NULL;
    getmemory(str);
    printf("%s/n", str);
    free(str);
    return 0;
}
```

为什么在每个头文件都能见到

```
#if !defined(AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79FE967FE8B6_
#define AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79FE967FE8B6__INCL

/*
 文件内容
*/

#endif // !defined(AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79FE967
```

```
#ifndef AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79FE967FE8B6
#define AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79FE967FE8B6

/*
 文件内容
*/

#endif // #ifndef AFX_STDAFX_H__8OED7A19_965B_4FB7_B54A_79F
```

浮点数的内存表示问题

```
main()
{
    float x = 3.0/5;
    if(0.6 == x)
        printf("x==0.6\n");
    else
        printf("x!=0.6\n");
}
```

```
main()
{
    double x1, x2;
    x1 = (3.14 + 1e20) - 1e20;
    x2 = 3.14 + (1e20 - 1e20);
    printf("%f\n", x1);
    printf("%f\n", x2);
}
```

可以少传一个参数吗?

```
#define TOTAL_ELEMENTS(a) (sizeof(a)/sizeof(a[0]))

void init1(char array[], int len)
{
    for(int i=0; i <len; i++)
        array[i] = i*2;
}

void init2(char array[])
{
    int len = TOTAL_ELEMENTS(array);
    for(int i=0; i <len; i++)
        array[i] = i*2;
}
```

结构体变量的内存表示

```
#include <stdio.h>

typedef struct _A {
    int a;
    char b;
    float c;
} A;

void main()
{
    printf("%d\n", sizeof(A));
    return;
}
```

联合类型变量的内存表示

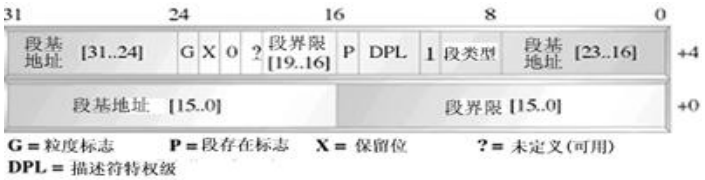
```
#include <stdio.h>

union A {
    unsigned int a;
    unsigned char b[4];
} g;

void main()
{
    g.b[1] = 1;
    printf("%d, %d\n", sizeof(A), g.a);
    return;
}
```

位域

```
struct A {
    unsigned char a:1;
    char b:3;
    unsigned char c:1;
    unsigned char d:1;
    unsigned char e:2;
};
void main()
{
    A x;
    char *p = (char*)&x;
    *p = 0xdb;
    printf("%d,%d,%d,%d,%d,%d\n",sizeof(A),x.a,x.b,x.c,x.d,x.e);
    return;
}
```

应用位域的例子—段描述符

```
typedef struct {
    unsigned int seg_limit_0_15 : 16;
    unsigned int base0_15 : 16;
    unsigned int base16_23: 8;
    unsigned int type : 4;
    unsigned int s : 1;
    unsigned int dpl : 2;
    unsigned int p : 1;
    unsigned int seg_limit16_19 : 4;
    unsigned int avl : 1;
    unsigned int unused : 1;
    unsigned int d_b : 1;
    unsigned int g : 1;
    unsigned int base24_31 : 8;
} SegmentDescriptor;
```



G = 粒度标志　　P = 段存在标志　　X = 保留位　　? = 未定义(可用)
DPL = 描述符特权级

关键字*volatile*的作用

```
int flag;

void onInterupt()
{
    flag = 0;
}

void thread1()
{
    flag = 1;
    while(flag)
    {
        // do something
    }
}
```