c深度剖析-符号

1.注释符号

```
int/*...*/i;                                这里看作空格
    char* s = "abcdefgh      //hijklmn"; 这里是字符串整体

    //Is it a \                              \代表换行
    valid comment?

    in/*...*/t i;                           类型不能拆
```

规则：

编译器在编译过程中删除注释，采用空格代替

编译器认为双引号括起来的都是字符串，包括//

/* */不能嵌套

```
y=x/*p              这里/*被看作注释，应该+括号
*/
y=x/(*p)            //更正
```

2.接续符和转义符

符号\

编译器会自动剔除\,跟在后面的字符自动接到前一行

接续单词时，\后不能有空格，下一行之前也不能有空格，

适合在宏代码块中使用，因为#define限制在一行内完成

```
#include <stdio.h>

#define SWAP(a,b) \
{                 \
    int temp = a; \
    a = b;        \
    b = temp;     \
}//直接替换可交换

void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
} //并不能交换
```

```c
int main()
{
    int a = 1;
    int b = 2;

    SWAP(a, b);

    printf("a=%d, b=%d\n", a, b);

    return 0;
}
```

还可以表示无回显字符，如\n , \t，也可表常规字符

\d 8进制,\x 16进制

3.单引号，双引号

```c
#include <stdio.h>

int main()
{
    char* p1 =  1 ;
    char* p2 = '1';
    char* p3 = "1";

    printf("%s, %s, %s", p1, p2, p3);  //error,p1,p2野指针问题
    printf('\n');                      //error，printf(char *)转换也是野指针
    printf("\n");

    return 0;
}
```

'a'字符常量，大小为1，'a'+1='b'

"a"字符串常量，大小为2,"a"+1='\0'结束符

```c
#include <stdio.h>

int main()
{

    char c = " ";      //其实是将字符串起始地址赋值给c

    while (c == "\t" || c == " " || c == "\n")
    {
        scanf("%c", &c);
    }

    return 0;
}
//只需将""全换成''不会报错
```

4.逻辑运算符

```
#include <stdio.h>

int main()
{
    int i = 0;
    int j = 0;

    if( ++i > 0 || ++j > 0 )
    {
        printf("%d\n", i);
        printf("%d\n", j);
    }

    return 0;
}
//answer 1 0 ,j未执行，换成&&都为1
```

程序短路规则：

||从左向右开始计算， 遇到第一个真停止计算

&&从左向右计算，遇到第一个假停止计算

三目运算符

(a?b:c)，a为真，返回b的值,否则返回c的值

```
int a = 0;
int b = 1;

(a < b ? a : b) = 3; //报错，因为等价于1=3
*(a < b ? &a : &b) = 3;
```

5.运算符

尽量使用括号来避免优先级错误

```
0x1<<2+3 结果为32
```

异或可用于实现交换

```
#define SWAP3(a,b) \
{                  \
    a = a ^ b;     \
    b = a ^ b;     \
    a = a ^ b;     \
} //采用了结合律
```

6.++，--操作符

```
int i = 3;
int x;
int j = (++i) + (++i) + (++i);  //18
x=(++i,i++,i+10);//15
```

逗号表达式从左到右顺序求值，返回最后一个表达式的值

++ ， --贪心原则：从左向右一次 尽可能多地读入字符

```
a+++b ; //等价于 a++   + b
++i++;  //报错
```

7. 优先级

[] ,(),. 的优先级都高于*

== !=高于位操作和赋值符号

算数运算高于移位

逗号优先级最低

```
int * ap[];  //等价于int * (ap[]) , ap是一个指针数组
int * fp();  //等价于int * (fp()) , fp返回值为int*型
*p.f      ;  //等价于*(p.f) ,  使用->代替
(val & mask !=0); //等价于 val & (mask!=0)
c=getchar() != EOF; //等价于 c=(getchar!=EOF)
msb<<4+lsb;   //等价于msk<<(4+lsb)
i=(1,2);      //等价于(i=1),2
```

8.类型转换

算数运算，低类型转换为高类型 char, short, int ,u int,long,u long,double,float

赋值中，右边转为左边类型