1.全局变量

1》定义：所有函数外部定义的变量

2》默认值为0

3》作用：长期储存内存

4》特点

生命周期跟随程序，程序启动创建，程序退出销毁

作用域是其所在位置的之下的所有函数，

就近原则

没有初始化，默认值是0

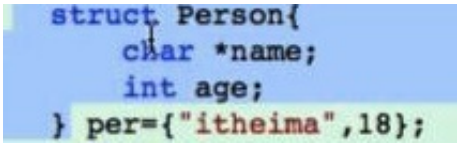
2.结构体

1》定义：构造类型，存不同类型的数据

2》用法：定义: 第一种:

```
struct 结构体类型{数据类型1 名称1；数据类型2 名称2}
struct person{
        char *name;
         int age;
};
struct perpon p1 = {"heima",18};
```

第二种:



```
struct Person{
    char *name;
    int age;
} per={"itheima",18};
```

取值：p1.name

赋值：p1.name = "hello";

p1.age = 8;

3》什么时候用：相关联的不同数据类型

4》特点：

可定义匿名类型结构体

同一个作用域内不可以重复定义，匿名结构体除外

可在函数外和函数内定义，效果等同于全局变量和局部变量，也可以嵌套定义

可在函数内外定义同一个类型，使用就近原则，结构体也是如此

3.结构体与数组

一,是什么?
是数组,只不是数组里每一个元素是结构体类型。
二,有什么用?
用于存储多个有关联的结构体.
三,怎么用?
定义及初始化赋值: struct 结构体类型 数组名[长度]={结构体1,结构体2};
如: struct Person{char *name;int age;};
struct Person per={"itheima",2}; struct Person per1={"itcast",3};
struct Person arr[2]={per,per1};
取值:数组名[角标];单独赋值:数组名[角标]=新的结构体;
如: struct Person per3=arr[0];
四,什么时候用?
当需要存储多个有关联的结构体,并作为一个整体使用时.
五,有什么特点?
1.遵循数组的特点

4.结构体的指针

1》使用: struct 结构体类型 *指针名 = &结构体变量

```
struct Person{
    char *name;
    int age;
};
struct Person per={"itheima",2};
struct Person *p=&per;
```

2》取值: (*指针名).结构体元素名 (*p).name

*指针名->结构体元素名 (*p)->name

3》赋值: p -> name ="heheh";

p -> age = 18;

4》特点: 指针移动对结构体本身没有意义,结构体名并不包含第一个元素的地址

5.结构体的内存分析

1》原则:

结构体分配内存空间遵循两个原则 -对齐原则或对齐算法.(增加cpu寻址效率)
1.每个元素的偏移量(当前元素首地址距离整个结构体首地址的字节数)必须是自已所占有字节数的整数倍.如果不够补齐上一个元素的字节数.
2.整个结构体的字节数必须是其最大元素所占字节的整数倍.如果不够,最后一个元素填充字节.

6.枚举

7.预处理指令: 文件包含

宏定义

文件编译

8。宏定义

1》用法: #define 被替换的内容 替换的内容: #define K 6

2》结束: 出现#undef: #undef K

3》特点

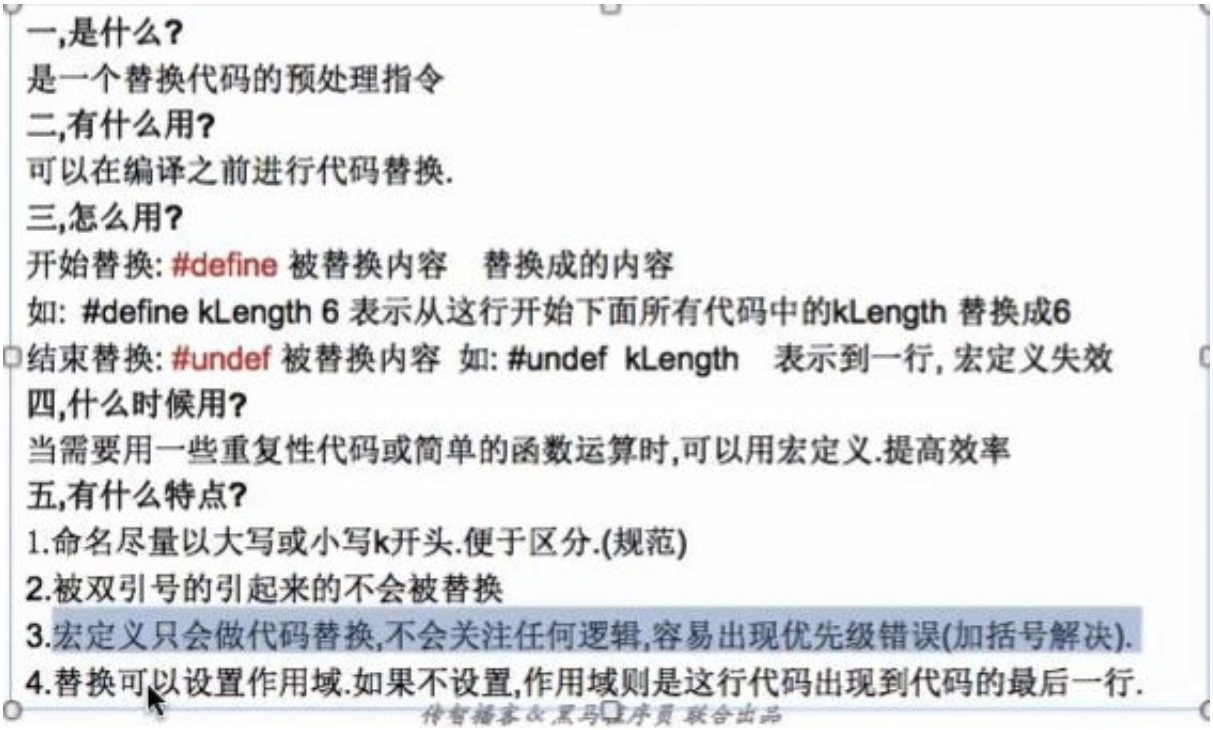
命名尽量以大写或者小写开头(规定)

被双引号引起来的不会被替换

只会做代码替换,不会出现任何逻辑,容易出现优先级错误(加括号解决)

替换可以设置作用域,如果不设置,作用域是这行代码出现代码的最后一行

9.预处理指令



10.补充

1》指针与地址（指针不是地址）

```
//指针与地址:
1.指针在内存中占用8(64位),并且,指针里面能存地址,指针有自己的地址
2.指针里面的存的地址可以改变,但是地址本身不能变化.
3.指针有类型概念.地址只是一个16进制的常量
4.指针本身可以移动指向新的数据空间.并且指针移动一位字节数不一样(由类型决定)
5.指针有两层含义:1.表示一个能存地址变量(等效于指针变量).2.指还含有数据类型的概念
```

2》 字符串

```
char* c="heheh";//表示字符串
char  *arr[2]={"hhahah","heheh"};//表示字符串数组
char (*arr)[3];表示一个长度为3的char类型数组的指针
```

3》指针字符串和字符数组及字符串的区别

```
指针字符串及字符数组字符串区别的补充
char *p="haha";
p="heihei";//正确
//p[0]='q';//错误
char arr[]="haha";
arr[0]='q';//正确
arr="heihei";//错误
arr[0]='h';//正确
```