# 一、BASE

## 变量

int 整形 %d

float/double 浮点型 %f

short 65535-长度

long

字符类型：char

每一个字符都有自己的编码ACSII，如果字符串按数字输出，将输出ACSII编码

char a = ‘12’ %c

scanf（“%d”）获取用户输入数据

## 数组

int Array[20] //定义20长度的数组

int Arr[10]={0,1,....} //定义一个10个大小的数组,连续空间

int Array[]={1,2,3,4} //定义并赋值

printf(“%d”,Arr[1]) //输出

## 字符串

Char name[]=”holle” //字符串，有\0

Char name[]={“h”,”o”,”l”,”l”,”e”} //字符串数组，没有\0

// \0字符串结束标准，字符串的内容要循环修改，将字符一个一个赋值，不能直接等于赋值

char name[20]={‘a’,’b’,’\0’}

**使用内置函数拷贝**

//主要是因为，C语言中字符串是存放在数组中的，拷贝字符串需要把数组中每一个元素拷贝一遍

#include <string.h>

strcpy(a,b)

## 函数

int getName(int a, int b)

{

return 0; 必须有返回值

}

## 引入文件

头文件：将类似功能的的函数放到一个.h文件中

使用#incloude “header.h” 引入当前目录下的文件

# 二、scanf

scanf(“%s”,input); 字符串不用添加&

scanf(“%d”,&input);获取数字时要添加&

# 三、指针

**在1个等式上不能同时出现\*,&;**

**当定义一个变量是指针类型时，它唯一多出来的属性是 \* ，可以使用\*获取变量中地址的内容，其他的变量属性是不变的，为了防止内存溢出，\*p声明时就应赋值**

int a=5

int \*p // \*p指针变量，4个字节

## 情况一

p=a // p保存的是a中的值，相当于普通变量

## 情况二

p=a;

printf(\*p) //这里会溢出，因为P变量存的值是5,\*p是找 内存中指向5的位置

## 情况三

\*p=a / p=&a //这时2种方式，都是获取a的地址，变量p中保存的都是地址

printf（\*p）

## 情况四

\*p=&a // 这里会溢出，不知道指到哪里了

## 情况五

int p, b=5;

p=&b; //a中存放地址

printf(\*p); //语法错误，只有指针类型才能使用 \*

printf(p); // 打印b的地址

## 情况六

int \*p, b=5

p = &b; // 存放b的地址，普通变量也可存放地址

\*p = 15 //给 b 赋值

## 情况七-指向数组的指针

int stu[2] = {1,2}, \*p

// 这3中写法都指向了数组stu，stu存放的是数组的首地址，

// stu[0]存放的是首元素，so，要使用 &stu[0]

p = &stu[0] / p=stu / \*p=stu

printf(\*(p+3)) // 指针下滑，指向第3个数

## 情况八-指向函数的指针

int func(int a){}

int (\* pfun)(int a)； // 定义一个函数指针

pfun=func; // 指向一个函数

pfun=&func; // 指向一个函数

printf（pfun（1）） // 调用函数

# 四、结构体

结构体的存在是为了节省代码，相同的变量，不用多次声明，通过结构体声明多个具有相同属性的变量

## 定义方式一

1. 声明结构体

struct Player{

int a;

}

1. 使用结构体

struct Player p1; // p1包含Player中声明的所有变量

p1.a = 1; // 给结构体的属性赋值

## 定义方式二

typedef struct \_Player{

int a;

}Player;

Player p1;

p1.a = 1 // 给结构体的属性赋值

## 指针使用结构体

typedef struct \_Player{

int a;

}Player;

Player \* nowp //声明一个**机构体**指针，这个指针（nowp）指向结构体（Player）

nowp=&p1 // 将结构体指针，指向p1

nowp->a // -> 指针的成员操作符与.意思相同

## 结构体数组-没确定使用方式

typedef struct \_player{

int a;

}Player;

Struct Player info[30] // 使用结构体Player 声明一个30容量的info

info[1].a=1;

# 五、联合体，结构体数组

typedef struct \_Play{

int a;

union { // 联合体，类似枚举类型,只保存联合体中最大的容量

int ID;

char name[20];

}job;

}

# 六、枚举类型

enum SYS{ // 枚举类型的值是根据顺序排列的

ADD=1, // 1

QUERY=2, // 2

}

enum SYS input; // 声明一个枚举变量input

scanf(“%d”, &input) //输入信息放到input中,提示用户输入1、2、3、4

if (input == ADD) {} // 枚举类型会自动管理相关变量

# 七、动态分配空间

# C语言-Player结构体动态分配空间

Player \* None=（Player \*）malloc(sizeof(Player));

#

Player \* None = new Player;

free //释放空间

# 八、文件操作

FILE \* f1; //定义一个文件指针

f1=fopen(“路径”, “w”)

# 九、杂项

变量存储内省

堆区 存放动态分配的数据

栈区 存放局部数据，局部变量

全局数据区 全局数据和静态数据，全局变量

程序代码区 程序的各个函数代码

变量存储类型

Auto自动类

register寄存器类 仅局部变量

static静态类

extern外部类