#define \_CRT\_SECURE\_NO\_WARNINGS

////博客

//1、总结

//2、记录学习的过程和心得

//3、愿意分享的

//gitee-代码的托管，维护网站，个人品牌

//大公司喜欢，了解+注册 ，三板斧+数据连接

//印象笔记（有道云笔记）

//xmind-思维导图

//原文件的第一行：使用scanf而非scanf\_s,只要在vs的环境

//sacnf\_s 是vs编译器自己的提供的函数，非标准C提供的函数，只有VS 编译器认识，其他编译器不认

// 建议使用scanf，scanf\_s要使用需要了解，略有不同

//> 应用软件 上层软件

//

//> 操作系统 下层软件

//> 驱动层 C语言

//> 硬件

//

上

//> 计算机语言的发展:低级---- > 高级

//> 二进制的指令--->汇编指令(助记符)----->B语言----->C语言（高级语言）

//低级语言

//

//C语言的国际标准：

//ANSI C(美国英语总计)

//

//编译器：Clang （刷题）、GCC（主流）、win - TC、MSVC、Turbo C

//VS2019

//SUBLIME(编辑器)

//编辑器——编辑

//编译器——编译

//C / C++ 是编译型的语言

//test.c---- > test.exe

//编译、链接

//python解释器语言

//

//.h 头文件

//.c 源文件

//.c\.cpp:

//.cpp编译器会按照C++的语法编译代码

//.c编译器会按照C的语法编译

//为什么写代码？

//解决问题

//C语言中一定要有 main 函数

//主函数

//printf是一个库函数

//专门用来打印数据

#include <stdio.h>//使用头文件

//给printf的打的招呼

//std-标准

//i—input

//o—output

//运行代码：(fn)+crtl+F5

//int main()

////int整型类型

////int main(void)

////告诉main无参数

////规定：main函数是程序的入口

////main有且仅有一个

//{

// printf("hehe\n");

// //hehe字符串

// return 0;

// //0是整数，前后呼应;

// //习惯：异常返回非0，正常返回0

//}

//古老的写法

//void main()

//{

//

//}

//数据类型：

// 1、内置的

//整数—整型short int long long long

//小数—浮点型—小数点可以浮动float double

//字符—字符型char

//为什么出现这么的类型？

//类型的设计来源生活

//2、自定义的

//struct....//结构体

//int main()

//{

// printf("%zu\n", sizeof(char));//sizeof谁的大小，在内存中的所占空间的大小 %zu sizeof返回的无符号整型u-无符号，z返回值

// printf("%zu\n", sizeof(short));

// printf("%zu\n", sizeof(int));

// printf("%zu\n", sizeof(long));

// printf("%zu\n", sizeof(long long));

// printf("%zu\n", sizeof(float));

// printf("%zu\n", sizeof(double));

//

// return 0;

//}

//返回的1为1字节

//计算机中的单位：最小的单位bit—比特位

// byte-字节

//kb mb gb tb pb 1024

//计算机能理解二进制

//十进制的世界：0123456789

//二进制的世界：01

//1个bite位：存储一个1|0的大小

//1B=8b

//C语言的规定：

//sizeof(long)>=sizrof(int)

//空间利用率高，使用灵活，适当的时候更加贴切

//int main()

//{

// int age = 20;//类型的用处：创建变量、-向内存索要空间

// double price = 66.6;

// return 0;

//}

//double、float都有可能不精准保存

//变量，空间，容纳常量 和常量

//int b = 203;//全局变量

//

//int main()

//{

// //short age = 20;//年龄

// //int height = 180;//身高

// //float weight = 88.8;//体重

// //变量与变量名

// int a = 10;//局部变量

//

//}

//变量的分类：局部（大括号外部），全局变量(大括号外部)

//同一下一个变量只能出现一次，否则会报错

//当全局变量和局部变量名字相同的情况下，局部优先

//建议：不要讲全局和局部的名字写成一样的

//计算两个整数的和

//sacnf 是一个输入函数

//printf 是一个输出函数

//int main()

//{

// int u1 = 0;

// int u2 = 0;//建议初始化，否则可能报错或者随机值

// //输入 计算 输出

// scanf("%d%d",&u1,&u2);//&取地址符

// int sum = u1 + u2;//变量的创建，使用

// printf("%d\n",sum );

//

//

// return 0;

//}

//在VS的安装路径下有一个文件:

//newc++.cpp的文件

//在vsd工程中创建的.c或者.cpp文件的时候，都是拷贝news++.cpp这个文件

//变量的作用域

// 1、局部变量\_所在局部范围

//

// 2、全局变量\_z整个工程

//

//

//extern int a;//声明外自外部的变量，与include不同，只是申明变量

//

//void test()

//{

// printf("test-->%d\n",a);

//}

//int main()

//{

// test();

// {

//

// printf("%d\n", a);

// }

// printf("%d\n", a);

// return 0;

//}

//变量的生命周期//生命周期约等于作用域

//局部变量:进入作用域

//全局变量：整个程序的生命周期

//常量，

//1、字面常量

//2、const修饰的常变量，const起锁定专业

//3、define定义的标识符常量

//4、枚举常量，枚举类型

//

//int main()

//{

// 30;

// 3.14;

// 'w';//字符

// "abc";//字符串常量

// const int a = 100;//常变量不可修改，常属性不可修改

// //在C语言中，const修饰的a本质是变量，但是不能被修改，具有常属性

//

// int arr[10] = { 0 };//数组。连续的空间，可以储存10个整型。【】只能放常量

//

// return 0;

//#define max 100//没有类型

//#define str "abcdef"

//int main()

//{

// printf("%d\n", max);

// int arr[max] = { 0 };

// printf("%s", str);//s取字符串

// return 0;

//}

//enum Color //枚举的关键字,枚举常量大写

//{

// //三个可能取值就是枚举常量

// RED,

// BLUE,

// GREEN

//};

////性别

//enum sex

//{

// MALE,

// FEMALE,

// SECRET//枚举常量不可改

//};//枚举常量只是一个整型

//int main()

//{

// //三原色 RGB

// enum Color c = RED ;//变量不会像内存申请开辟空间

// printf("%d", c);

// return 0;

//}

#include<string.h>

//int main()

//{

// //char字符类型

// // 'a';

// //char ch = 'w';

// //字符串"abcdef"

// //C语言没有字符串类型，""

// char arr1[10] = { "abcdef" };//存在数组中，内存数字可以省略，自动识别；字符串的结束标志：\0,但不算做字符串内容

// char arr2[] = { 'a','b','c','d','e','f'};

// int len = strlen(arr2);

// //求字符串函数的一个函数，string length

// //使用头文件<string.h>

// printf("%d\n",len);

// printf("%s\n", arr1);

//

// printf("%s\n", arr2);//没有结束标志,在字符中难以打印出\0

//

// return 0;

//}

//转义字符，转变字符原来的意思

//int main()

//{

// printf("abc\n");

// return 0;

//}

//\n换行字符

//int main()

//{

// printf("abc\0def");

//}

//\0字符串的结束标志

//转义字符

// \?防止成为三字母词的？

//三字母词

//??）---->]

//??（---->[

//%d-打印整型

//%c-字符

//%s-字符串

//%f-float

//%lf-double

//%zu-sizeof的返回值

//int main()

//{

// //printf("%s\n", "(are you ok \?\?)");

// ////(are you ok]

// ////现在不支持三次母词

// //printf("%c\n",'\'');

// //printf("asdf\n");

// //printf("%s\n","asdf");

// //printf("\"");

// //printf("abcd\\0ef");

// //printf("c:\\test\\test.c");

// ////printf("\a\a\a\a\a");//报一个警告

// //printf("aba\raba");

// printf("%c\n",'\130');//八进制的130——> 十进制的88

// printf("%d\n",'\x063');//转出的十进制数不能大于127

// //转义字符转出的只是一个字符

// //空格 也是一个字符

// printf("%d\n", strlen("c:\test\628\test.c"));

// //\t只是一个字符，相当于TAB四个空格

//

// return 0;

//}

//ASCII 编码

//a-97

//A-65

//0-48

//注释：梳理思路；对复杂的代码进行解释；帮助自己，帮助别人的；

//int main()

//{

// /\*

// int a=10;

// int b=10;

// 早期的c语言注释

// \*/

// //cpp的注释风格，不支持嵌套注释

// //创建指针变量p,并赋值为NULL

// int\* p = NULL;

// return 0;

//}

//int main()

//{

// printf("加入比特\n要好好学习吗(1/0)\n");

// int a = 0;

// scanf("%d", &a);

// if (a == 1)

// {

// printf("好offer");

// }

// else

// {

// printf("卖红薯");

// }

// return 0;

//}

//选择语句：if else

// switch

//循环：while for do while

//int main()

//{

// int line = 0;

// printf("加入比特\n");

// while (line<20000)//真循环

// {

// printf("写代码%d\n",line);

// line++;

// }

// if (line >= 20000)

// {

// printf("好offer");

//

// }

// else

// {

// printf("继续加油");

// }

//}

//C语言是【程序化】的程序设计语言

//顺序、选择、循环

//函数

//求两个任意整数的和

//int Add(int x, int y)//函数参数

//{

// int z = 0;

// z = x +y;

// return z;//函数体，return (x+y);

//}//函数 参数--->结果

//// 输入 输出

////函数名 返回类型

////简化代码，反复使用

//int main()

//{

// int a = 0;

// int b = 0;

// scanf("%d%d", &a,&b);

// int sum = Add(a,b);

// printf("%d", sum);

// return 0;

//}

//数组：一组相同类型元素的集合

//int main()

//{

// int arr[10] = { 0,1,2,3,4,5,6,7,8,9 };

// //数组可以是int char double

// //数组的小标，下标是从0开始的

// //[]中的数字可以不写

// int i = 0;

// while(i<10)

// {

// printf("%d ",arr[i]);//通过访问下标，访问数组的元素。下标需要通过循环实现

// i++;

// }

// return 0;

//}

//在线OJ

// OJ-online judge

// 很流行

// 很多多联网公司在笔试环节都采用在线OJ的形式

//1、io型 所有的代码都是自己来完成和实现

//输入、计算、输出

//2、接口型，补充型

//只需要完成一个函数

//假设其他需要的数据都是准备好的

//int main()

//{

// printf("Name Age Gender\n");

// printf("---------------------\n");

// printf("Jack 18 man\n");

// return 0;

//}

//int main()

//{

// printf("I lost my cellphone!");

// return 0;

//}

//int main()

//{

// int a = 40, c = 212;

// int r = (-8 + 22) \* a - 10 + c / 2;

// printf("%d",r);

// return 0;

//}

//

// 0-数字0

// '0'-字符0，ASCII值是48

// '\0'-字符-ASCII值是0

// EOF-end of file 文件的结束标志 值是-1

//int main()

//{

// int arr[10];

// //初始化时，不填写内容必须标注大小或者填写初始化内容（变化）

// //整形数组没有/0

// //不完全初始化，剩余部分默认初始化为0，其等价于‘\0’

// sttrlen求字符串大小的

// int arr[10]={1};

// 第一个是1，其他默认不完全初始化

// C99的标准之前，数组的大=大小都是用常量或者常量表达式来制定

// C99的标准之后，支持了变长数组，这个时候允许数组的大小是变量，但这种这种指定方式的数组是不能初始化的

//}

//vs对C99中的一写语法支持不是很好，不支持边长数组

//windows 操作系统

//linux 开源 操作系统，适合技术操作、技术教学

//国内服务器比较多的Linux

//int Max(int x, int y)//标注函数的返回值的类型

//{

// if (x > y)

// return x;

// else

// return y;

//}

//

//int main()

//{

// /\*char ch[] = { "Hello,bit" };\*/

// int a = 0, b = 0;

// scanf("%d%d", &a, &b);

// int r = Max(a, b);

// printf("两个整数的较大值%d", r);

// return 0;

//}

//int f(int x)

//{

// if (x < 0)

// {

// return 1;

// }

// else

// {

// if (x = 0)

// return 0;

// else

// return -1;

// }

//}

//

//int main()

//{

// int a = 0;

// scanf("%d",&a);

// printf("%d",f(a));

// return 0;

//}

//操作符

//灵活性——丰富的操作符

//对操作符熟悉

// / %

//int main()

//{

// float a = 7 / 2.0;

// //除号的两端都是整数的时候，执行的是整数的除法，如果两端只要有一个浮点数就执行浮点数的除法

// int b = 7 % 2;

// //取模操作符的两个操作数只能是整数

// printf("%.2f\n", a);

// printf("%d\n", b);

//

// return 0;

//}

//移位操作符——涉及二进制的运算

//右移>> 左移<<

//位操作符 & ^ |

//赋值操作符

//int main()

//{

// int a = 0;//初始化是赋值吗？不是

// a = 20;//赋值改变原有值

// a = a + 3;//a=23 //a += 3

// a -= 3;

// a %= 2;

// return 0;

//}

//单目操作符

//!逻辑反操作，0表示假，非0表示真

// - +（没什么意义） 正负值

//&跟指针有关

//sizeof是单目操作符

//~ \*

//a+b

// + 双目操作符

//单目操作符——只有一个操作数的操作符

//int main()

//{

// /\*int flag = 0;

// if (!flag)

// {

// printf("hehe");

// }\*/

// int a = 4;

// printf("%d\n", sizeof(a));//4

// printf("%d\n", sizeof a);//4 操作符

//

// printf("%d\n", sizeof(int));//4

// //类型还是要加上括号

// int arr[10] = { 0 };//整形数组没有/0！！！

// printf("%d\n",sizeof(arr));//40 计算整个数组的大小，单位是字节

// printf("%d\n",sizeof(arr[0]));

// printf("%d",sizeof(arr)/sizeof(arr[0]));//10-数组的元素个数——数组的总大小/数组的第一个大小

// return 0;

//}

//前置的++，——后置的

//int main()

//{

// int a = 10;

// //左向理解+ -

// //int b = a ++;//后置加加，先使用，后++

// // int b = a;a=a+1

// //int b = ++a;//前置++，前++，后使用

// //a+=1,b=a;

// printf("%d\n",b);

// printf("%d\n",a);

// return 0;

//}

//强制类型转换

//int main()

//{

// int a = (int)3.14;//(强制类型转换)

// //3.14字面浮点数，编译器默认理解为double类型

// //强制类型转换

// printf("%d\n",a);

// return 0;

//}

//大一学扎实

//

// 纠结读研

// 有没有能力考研——评估能力——英语、数学、专业课技能、家庭支持；研究生>本科生

// 20-25%考录比

// 10—15%保研

// 10%考成功的1/10

// 能力弥补学历不足

//

// 计算机行业——公平，不怎么看重学历学校层次，看重技能、项目

//

//

//关系操作符

//> >= < <= !=

// == 判断相等

//int mian()

//{

// int a = 10;

// if (a = 3)//赋值，表达式答案为3，为真

// {

// printf();

//

// }

// return 0;

//}

//逻辑操作符

//&& 逻辑与（且）

//|| 逻辑或

//int main()

//{

// //int a = 0;

// //int b = 20;

// //if (a && b)

// // //a b 逻辑与：都真才真，一假则假

// //{

// // printf("hehe");

// //}

// int a = 0;

// int b = 0;

// if (a || b)

// //a b 逻辑或：有真则真，

// {

// printf("hehe");

// }

// return 0;

//}

//条件操作符（三目操作符）

// e1 ? e2 : e3

// 真 √ ×

// 假 × √

//有三个操作数

//int main()

//{

// int a = 10;

// int b = 20;

// int r = (a > b ? a : b);//画图理解

// printf("%d",r);

// return 0;

//}

//逗号表达式：从左向右依次计算，整个表达式的结果是最后一个表达式的结果

// ， ， ， ，， ，；

//下标引用、函数调用、结构成员

//[]、()、.->

//int main()

//{

// int arr[10] = { 1,23,4,5,67,78,88,9 };//定义数组的语法形式 ，可初始化数组的大小必须是常量规定

// int a = 8;

// arr[a];//下标引用操作符，arr、3就是【】的操作数,访问数组（此处）可以是变量，变量是程序运行时才有的

// return 0;

//}

//函数调用操作符

//f（x,y）//()就是函数调用操作符，add，2,3,都是（）的操作数

//常见关键字——C语言本身内置的

//关键字不是自己创造的，也不能自己创建

//auto-所有局部前都有被省略的auto continue 循环

//break-跳出循环

// for while /do/ while default if else goto

//case-循环 switch case

//内置类型 char short int long float double\_关键字 signed 有符号的 unsigned 无符号的 void\_无（函数的返回类型，函数参数）

// sizeof 计算大小的 typedef类型重命名

//const 修饰变量 常属性

//enum\_枚举 struct-结构体 union\_联合体（共用体）

//extern 声明外部符号

//register 寄存器 static 静态的

//return 函数返回值

//volatile\_操作系统

//变量名不能是关键字

//变量的命名：

//1、有意义；2、名字必须是字母、数字、下划线组成 不能有特殊字符，不能以数字开头 3、变量名不能是关键字

//int main()

//{

// auto int a = 10;//自动创建自动销毁

// return 0;

//}

//关键字typedef-类型重命名——简化名字

//类型重命名、定义

//typedef unsigned int uint;

//typedef struct Node

//{

// int data;

// struct Node\* next;

//}Node;//定义链表节点

//int mian()

//{

// unsigned int num1 = 0;

// uint num2 = 0;

// struct Node n;

// Node n2;

// return 0;

//}

//static\_静态的

//修饰变量（局部、全局）、函数

//void test() //test()执行任务 应该有个return void不需要任何返回

//{

// static int a = 1;//static 修饰局部变量的时候，局部变量出了作用域，不销毁

// //本质上，static修饰局部变量时，改变了变量的存储位置，影响了变量的生命周期，和程序的生命周期一样

// //内存：栈区（局部变量...）+堆区（动态内存开辟、函数malloc、freecalloc realoc）+静态区（静态变量、全局变量，程序销毁才销毁）

// //没有对应的汇编语言，不参与执行的，不会重复创建a，a的地址不发生变化

// //全局变量、静态变量在编译期间已经分配了空间、指派地址，并不是在运行期间

// //局部变量在运行期间创建、销毁

// a++;

// printf("%d\n",a);

//

//}

//

//int main()

//{

// int i = 0;

// while (i < 10)

// {

// test();//循环十次

// i++;

// }

// return 0;

//}

//修饰全局变量

//外部文件

//全局变量具有外部链接属性，编译+链接-->可执行程序

//static修饰全局变量的时候，这个全局变量的外部链接属性就变成了内部链接属性，改变链接属性

//其他源文件（.c）就不能在使用到这个全局变量

//全局变量存储在静态区,也称为全局区

//静态存储区包括BSS段和数据段，其中未初始化的全局变量位于BSS段，而已初始化的全局变量则位于数据段。这些存储位置在程序开始执行时就已经确定，与局部变量在栈空间的不固定地址形成对比。

//这是在程序编译时就已经分配好的。全局变量一直到程序结束才释放。全局变量可以被任何函数访问，如果没有特别声明，全局变量的默认访问权限是外部的，也就是说，可以在其他的C文件中访问。

//被static修饰的全局变量，感觉作用域变小了

//静态局部变量在全局数据区分配内存，且在内存中的位置固定不变。其作用域为局部作用域，即只在定义它的函数内有效，在函数外部不能调用。

//静态全局变量也是在全局数据区分配内存，但是作用域只在定义它的文件内有效，不能被其他文件引用

//安全性、减伤冲突性

//extern int g\_val;

//int main()

//{

// printf("%d", g\_val);

// return 0;

//}

//static修饰函数

//函数本来具有外部连接属性

//静态函数，外部链接属性就变成了内部链接属性，其他源文件（.c）就不能在使用这个全局变量

//extern int Add(int x , int y);//声明外部函数

//int main()

//{

// int a = 10;

// int b = 20;

// int r = Add(a, b);

// printf("%d\n", r);

// return 0;

//}

//register 寄存器关键字

//电脑上的存储空间（存储体系）：（金字塔结构，访问读写速度快，空间小，造价高）

//寄存器（集成到CPU上）

//高速缓存（cache）

//内存

//硬盘

//CPU中央处理器计算能力快，硬件造价高、内存跟不上

//创建寄存器变量

//int main()

//{

// register int num = 3;//建议 3存放在寄存中；编译器决定是否放在寄存器

// return 0;

//}

//define定义常量和宏

#define num 1//标识符常量

//#define ADD(x,y) ((x)+(y)) //宏是有参数的 宏是完成替换的

////宏和函数体有点区别

//// 宏名宏的参数（参数无类型），宏体

//int main()

//{

// int a = 10;

// int b = 10;

// int c = ADD(a,b);

// return 0;

//}

//.h头文件（函数的声明、类型的声明、头文件的包含）

//.c源文件（函数的实现）

//浮点数的四舍五入，不能用肉眼看到的数值来计算

//浮点数在内存中有可能不能精确保存 12.455 12.4549999.....\_12.45

//如果在v1.c中写了#include "v2.c"，那么v2.c会被当成v1.c的一部分，此时虽不在同一个源文件中，但二者仍像在同一个文件中一样，是不能称为“外部”的。

//指针

//内存：程序运行是在内存中 ——内存会划分为一个个的内存单元，一个内存单元的大小1字节，每个内存单元都有一个编号（地址）

// 32位电脑 生成地址

// 地址线（电线）电信号 1/0

//32根地址线：2^32个地址序列，管理一个内存单元，管理4,294,967,296个字节 管理4GB

//bit-比特位

//byte-字节 字节跳动（今日头条、抖音） 菜鸟网络 美团北京三快科技有限公司 对公司有一定了解

// KB

// MB

// GB

// TB

// PB

//64位电脑

//char c://1byte bit 地址会太多，综合考虑字节更合适

//int main()

//{

// int a = 10;//向内存申请4个字节存放10 ，取地址取首地址

// &a ;// 取地址符

// printf("%p",a);//%p打印地址,

// int\* p = &a;//指针变量，存储地址（地址）的变量

// //int\*是类型，\*代表是指针变量，int说明的是p指向的对象是int类型的

// //变量名是p

// \*p=20;//解引用操作符，意思是通过p中存放的地址，找到p所指向的对象，\*p就是p指向的对象

// //地址不能随意改动

// printf("%d", a);

// char ch = 'w';

// char\* d = &ch;

// return 0;

//}

//内存单元

// 编号->地址（又称为指针）

//内存：地址+内存中的数据+内存数据解析的不准确（文本解析）

// 一个数字-32进制数-16进制数//倒存

//GO语言在北上广用得比较多

//int main()

//{

// //不管是什么类型的指针，都是在穿件指针变量

// //指针变量都是用来存放地址的

// //一个指针变量的大小取决于一个地址存放的时候需要多大发空间

// //32位机器上的地址：32bit位——4字节，指针变量大小就是4byte

// //64位 64bit位——8字节，指针变量大小就是8byte

//

// printf("%zu\n", sizeof( int\*));

// printf("%zu\n", sizeof (char\*));

// printf("%zu\n", sizeof (double\*));

//

// int\* p1,p2,p3;//p1是指针，p2、p3是int

// int \*p1,\*p2,\*p3；

// return 0;

//}

//人：名字+年龄+性别+地址+电话

// 书：书名+作者+出版社+定价+书号

// 复杂对象

// C语言有自定义类型的能力

// 自定义类型中有struct

//结构体：把一些单一类型组合在一起的做法

//学生

struct Stu //用类型创建变量

{

//成员，

char name[20];

int age;

char sex[10];

char tale[12];

};

void print(struct Stu\* ps)

{

printf("%s %d %s %s", ps->name, ps->age, ps->sex, ps->tale);

//结构体指针变量->成员名

printf("%s %d %s %s", (\*ps).name, (\*ps).age, (\*ps).sex, (\*ps).tale);

}

int main()

{

struct Stu s = {"zhangsan",20,"man","13286948411"};//初始化

printf("%s %d %s %s", s.name , s.age , s.sex, s.tale);

// . ：左边结构体对象.成员名

print(&s);

return 0;

}

//continue 属于循环结构，跳出某次循环

//define 不是关键字，是预处理指令