进制的转换BWG7X-J98B3-W34RT-33B3R-JVYW9

二进制转10进制，1000101 从右往左从2的0次方开始，相乘。

二进制转8进制，三个一组同样成2的次方，结果依次排序；

二进制转16，四个一组，同上；

十进制转2进制/2，转16进制/16，转8进制/8；

八进制转2进制，每个数字拆开/2，分成三个一组，不足补0；

十六进制转2进制，每个数字拆开/2，分成4个一组，不足补0；

&符号，相同为1有0为0；||有1为1，相同为0 ; ^，相同为0，不同为1；~求补，只作用于二进制，每个数取反，1变0，0变1； <<左位移，向左移动N为，相当于原数乘以2的N次方。 >>右位移，相当于原数除以2的n次方。

负数，先换成二进制，补0（8位）,取反，补1； 二进制最高位为0表示此数为正数，为1表示为负数。

随机数的计算

-要取得[a, b)的随机整数，使用rand() % (b - a) + a;

要取得[a, b]的随机整数，使用rand() % (b - a + 1) + a;

要取得(a, b]的随机整数，使用rand() % (b - a) + a + 1;

if判断

if(i=0)

{

这是表示这个判断永远都是false 为1的话是为true; 如果为其它，则表示赋值。

}

循环语句

For(1;2;3)

{

4 //执行顺序1-2-4-3-2；  
}

Int x=0;

如果(int i=1;i<=100;x+=y,i+=1)

{

X+=1;

}

输出结果是5150,因为x每次都+1，循环100次.

1

While（2）

｛ 4

3

执行方式 2-4-2-4；

｝

do while 执行方式 4-2-4-2-4；

1

Do

{

4

3

}

While(2);

数组

数组地址的算法，arrd(i)=arrd[0]+i\*sizeof(数据类型)//很重要

求数组元素的个数方法：sizeof(arr[])/sizeof(arr[0]);

数组是由一组顺序排列的，具有相同类型的变量组合成的集合。（内存地址紧连，连续，数据类型必须一样，可以储存一个或多个元素。）

如果给数组赋值，只赋值一部分，剩下的默认为0。

通过直接赋值，可以不用给数组元素个数int array[]={1,2,3};

strlen输出char字符长度，不计\0；如果想输出其它类型的数组，则用printf(“%x\n”,数组名)；

除了char类型，直接输出都是地址，而char是字符串。Sizeof求的是字符的长度。如果是char类型，则计算的是容器大小。

如果是二维数组，没有strlen, sizeof(数组名)表示的是大小，如果是数组名[0]表示的是数组里面第一组的大小。

函数

能够完成某些特定功能的独立代码段，称之为一个函数。提高了可重用性。

函数重载， 参数列表一样，返回值必须一样。参数列表可以赋值也可以不赋值，赋值只能靠右边。如果赋值，在调用的时候不赋值就是默认。

函数重载，函数名称一样，但是参数和返回类型不一样或参数个数不一样是为函数重载。只要参数一样，都为重复声明。

形参只有在调用的时候才创建，在调用结束会删除这个空间。

常用库函数 floor 取一个不大于该数的整数，ceil取一个不小于该数的整数。Gets puts输出字符串。Abs绝对值pow是平方。 atoi 转int,atol,atof, itoa转成字符串，char sar[10]={0}; itoa(内容，数组名称，进制);

memcpy用于拷贝内存空间 memcpy(存储的空间从0开始计算, 要拷贝的空间, 大小);

memcmp用于比较内存大小，memcmp数组名1，数组名2，要比较的大小)；1，0，-1；

memset内存设置函数，绝对数用于清空缓存区memset(数组名，字符编码 0，大小sizeof(数组名) )；

Strcpy字符串拷贝函数strcpy(数组名，要拷贝的内容，可以是字符串，也可以是数组)；

Strcat字符串连接函数strcat(数组名，要加的内容，字符串或者数组)；

Strcmp字符串比较strcmp(数组名1，数组名2)；

Strstr查找字符串函数（需要查找的数组，查找的内容）；

递归函数

1自己调用自己，

2一般情况下，需要有一个终止条件，使得递归不会无限循环。

3一般条件下，需要有一个递归条件，使得递归可以无限循环。

指针

指针是储存内存地址的变量，指向一个数据结构的首地址。

未赋值的指针不能使用，为负数；

取地址运算符&，取得一个变量的内存地址

解引用运算符\*，取得某一个指针变量所指向的内存中的数据 直接取值；

指针常量，指针本身是常量.int const \*p=&a;能通过 \*p修改a的值

常量指针，指向常量的指针。Const int \* point=&a;

+

Int(\*p)[10]数组指针，指向数组的指针。

Int\* p[10]指针数组，本身是数组，数组里的元素都是指针。

程序内存空间 代码区，全局数据区（运行开始时就存在，全局变量，静态变量），堆区（动态变量），栈区（局部变量如int a=10;）。

创建新的内存：Int\* point=new int(100); 判断内存是否为空 ：if(point!=NULL); 删除数组：deldete[] point 如果是数组则delete[] point; 初始化memset(point,0,sizeof(int)\*数字大小)

内存操作事项：1创建后，判断是否为空，2初始化内存，3避免越界，4new和delete数量要相等 5delete后将指针设置为空；

Void类型指针，可以指向任何一个，但是在使用时要强转如：\*（数据类型\*）数组名称。

引用的声明方式：数据类型& 引用变量名=目标变量名。引用不会创建新的对象，只是多了个名字。引用不会占用内存。

Int& 函数名（int& a,int& b）｛return a｝特殊引用 函数名（a,b）=100;给a赋值。 不能返回局部变量，不能返回new类型的成员。either指针or引用。如返回类成员引用，最好const;

Const数据类型&引用变量名=目标变量名;常量引用，不能通过引用修改原变量的值。如果形参用const引用，只能通过实参来传值，不能在内部修改。

数组最为参数传递的时候，相当于一个指针，sizeof求的是首地址的大小 数据类型 函数名(数据类型& 数组名[],int b)

指针和引用的区别

1引用创建时必须初始化，指针不用，并且能空指针，但不能空引用。

2指针的大小永远为4，引用为引用的目标大小。

3对引用的修改就是对目标的修改。

4引用不会产生新的内存空间。

Void num(int\* p,int size){p=new int[size]}，这样会导致内存泄漏，因为函数里又重新new了一次，没有指针能指向，导致内存无法施放。加引用的话就可可以了 int\*& p;或者 \*\*p;里面变成\*p=new;

函数指针，指向函数的指针，返回值类型（\*指针变量）（函数参数列表）。无需解引用，直接用指针代替函数。

Int get(int,1,int,2,int(\*get1)(int,1,int,2)){return get1(a,b)}函数的指针作为函数的参数，可以传递某些条件。

函数指针数组，本身是个数组，里面所有元素都是指针，指向一个函数。Int(\*p[])(int,a,int b)={数组名1，数组名2}

返回值为指针的函数，就是指针函数。函数的返回数据类型\*函数名称（形式参数表）

复合数据类型

结构体struct 名字｛｝这里可以直接定义结构体；里面用分号隔开，使用的话直接用名字定义 struct 名字 sda={}初始化，或者使用.运算符初始化。但是必须要用memset,如果初始化字符串，只能用strcpy;

使用结构体指针来访问对象成员可以是结构体变量名->成员名。

结构体的内存对齐，整个的字节大小是最大的整数倍。如果两个其它类型紧邻，则可以打包，否则不可以。

Union联合体，和结构体声明方式一样。联合体的大小，为最大成员的整数倍，只能初始化第一个成员，同一时间只能用一个成员。好处是节省内存。

结构体和联合体的区别，结构体成员独立内存，联合体共享一段内存

关键字不一样。

enum 只能定义这个类，配合switch使用。

typedef类型定义符，给数据类型起别名。

Typedef int INT36; typede char NAME[];

C++新11特性

1 Auto自动数据类型推导 auto i=100, auto i=2.5;

2 decltype 获取指定变量的类型 int a=9; decltype(a) c=20;

3 nullptr 专门用于表示空指针；只能作用于指针。预防NULL二义性。

多文件结构

用< >引用头文件，是标准库，用“”是非标准库头文件，多半是自己写的

头文件4部分， 1版权声明，2引用的头文件，3函数声明

#pragma once表示只包含一次头文件。

源文件，1版权声明，2引用头文件，3函数实现。

1预处理->2编译->3链接

1预处理后，头文件包含进源文件，只剩下源文件。

2程序中所有.CPP文件编译为obj二进制文件。

3程序的obj和lib文件一起参与链接，最后生成可执行的exe文件。

宏定义在预处理阶段#define SIZE 20; 一改全改。

#ifdef SIZE表示用这个，#endif,结束用这个， #undef表示不用这个宏 #else,#elif相当与else ，else if;#if是判断这个

面向对象

数据（性质，状态）+操作（描述对象的行为）=类（表示相同或相似性质的一组对象）

对象则是类的具体实例。面向对象的三大特征，封装，继承，多态。

构造函数（在程序运行时调用）析构函数（程序结束时运行）拷贝构造函数（）格式 类名（const<类名>&<引用名>）别人的属性全部拿过来用。 调用 类名=目标命 或者 目标名（类名）

内联函数

可以提高效率，不用调用，直接使用。类里的函数都是内联，在声明的时候加inline

必须满足以下所有条件，才使用内联

1函数的代码非常短小，且使用频繁

2函数中没有非常消耗性能的

Static

静态局部：生命周期整个函数，只能在函数内部访问，在函数执行的时候创建；

全局静态变量：只能在当前的文件中访问。

静态函数：只能在当前文件中访问。

This是指针，在重名的时候可以指向当前类的成员。

友元打破封装和数据隐藏的手段。Friend

特点，单方向，不继承，不传递

Operator运算符重载 类名 operator运算符（类名&名）；

虚基类，解决多重继承问题，子类只继承一次。Class类名：virtual继承方式 类名。

虚函数非静态virtual，非友元，虚函数无论继承多少次，都保持虚函数特性。

同一个函数，在子类和父类中实现不同的内容。

纯虚函数，有声明没实现，在子类中实现。virtual 返回类型 函数名（参数表）=0；

有纯虚函数就是抽象类，不能实例化，不能创建对象。

动态调用时，删除时不会调用派生类析构，所以要定义父类为虚析构函数。

模版

函数模版 Template<typename T> T类名（参数列表） 都写在.h中，解决函数重载的问题。如不同类型比较大小。

针对指针的特化template<>T 类名<T>(参数列表)

类模板是在.h中定义，模板类是在.CPP中使用。

Template<typename T> class 类名｛｝ 实现： 类名（返回类型 int， float） 命名（数值）；

线性数据结构

顺序表：元素与元素之间连续，（查询访问效率高，向头部添加删除元素效率低 ）

链表：可以是连续的也可以是不连续的，每个元素都有数据域和指针域，数据域自身的地址，指针域，元素指针，（元素的类型），指向下一个元素的位置。（查询访问效率低，插入和删除效率高）

栈，先进后出，只允许一端插入或者删除。如同挤地铁。

队列，只能在头部插入和删除，在队尾添加，先进先出，如同排队买票

树的结构

自然数，树根在下，继承树，树根在上，度是最大子节点，叶子节点，除度外的所有的节点。高度为有几层

二叉树的遍历方式：1前序遍历（根左右），2中序遍历（左根右），3后序遍历（左右根）

插入排序，本身是一个有序的数组，插入后还是有序的。

插入是每次都是让数组变成有序的。有序越来越多，无序越来越少，从第一个元素开始

快速排序，选数组第一个数，比这个数大的放前面，小的放后面（先从最后找比他小的，调换，在从前面找比他大的交换）

二分搜索必须是有序的