c++学习笔记

1. 第一个c++程序

|  |
| --- |
| //引入头文件  #include <iostream>  //入口  int main()  {  //输出一句话  /\*\*  1.std::cout 向控制台输出内容  2.<< 输出运算符  3."" 输出内容  4.hello world  5.\n 回车  6.; 结束  \*/  std::cout << "Hello World!\n";  } |

1. C++输出数据
   1. 数据是信息的载体
   2. 数据分类
      1. 数字：0-9
      2. 字符：单引号括起来的
      3. 字符串：双引号括起来的
   3. 输出数据
      1. Std::cout可以输出各种类型的数据
      2. 操作符拼接多个数据项
      3. Std::endl换行使用
      4. Using namespace std;缺省命名空间
2. 程序的注释
   1. 程序添加说明文字，
      1. 单行注释
      2. 多行注释
3. 使用变量
   1. 变量是存储内存变量的简称，用于存放临时数据的。
   2. 属于之前需要先声明

语法：数据类型 变量名

常用的数据类型有：int、float、double、char、string、bool

* 1. 代码

|  |
| --- |
| #include <iostream>  int main()  {  char c = 'A';  int num = 10;  std::string name = "垃圾";  bool sex = false; //ture为1 false为0  std::cout << c << " " << num << " " << name << " " << sex;  std::cout << "Hello World!\n";  } |

1. 标识符命名
2. C++输入数据
   1. 输出类型
      1. 控制台节目输入
      2. 文件中读取
      3. 数据库中读取
      4. 网络中读取
      5. 输入方式
   2. 输出案例
      1. Std::cin >>变量名
         1. bool只能填入0和非0
         2. 输入数据和变量数据类型不匹配，导致结果不确定
         3. 代码

|  |
| --- |
| string name;  cin >> name;  std::cout << "Hello World!\n"<<name; |

1. 算术运算
   1. 算术运算包含了：+ - \* / %
   2. 只适用于整数和浮点数
   3. 取模只能使用于整数
2. 自增和自减运算
   1. ++变量名、变量名++、--变量名、变量名--
3. 赋值运算
   1. +=
4. 函数
5. 函数的位置
   1. 在main上面定义，下来写内容
   2. 在main的上面写内容
   3. 函数可以多次声明
   4. 函数的声明也可以写main函数中，必须放在使用之前，否则会报错。
6. 函数参数的传递
   1. 函数调用，调用这将参数传递给函数的参数
      1. 实参：可以是常量变量表达式
      2. 形参：函数调用的参数
   2. 函数中修改形参的值不会影响实参。
   3. 代码

|  |
| --- |
| #include <iostream>  int max(int a, int b);  int min(int a, int b);  int main()  {  int a = 10;  int b = 11;  std::cout << "max:" << max(a,b) << std::endl;  std::cout << "Hello World!\n";  }  int max(int a, int b) {  return a > b ? a : b;  }  int min(int a, int b) {  return a < b ? a : b;  } |

1. 函数分文件编写
   1. 使用头文件和源文件在编写代码
   2. 在使用的时候通过#将头文件加进去
   3. 可以写到头文件中的东西：头文件、命名空间、全局变量、函数声明，数据结构和类的声明等
   4. 源文件：函数的定义、类的定义。
2. Sizeof运算符
   1. Sizeof(数据类型) 得到数据类型或者变量占用的内存空间
   2. 32位和64为可能结果不一样
   3. String不是基本数据类型，sizeof(string)没意义
   4. 代码

|  |
| --- |
| cout << sizeof(int) << endl;  cout << sizeof(a) << endl; |

1. C++的基本数据类型
   1. Int float double char bool string(不是基本数据类型，但是比较重要)

第一节：int类型

1. 整型的基本概念
   1. C++使用int关键字声明变量，实际是integer的简写
   2. 声明的时候可以在int关键字之前加signed、unsigned、short、long四种修饰符
      1. Signed：有符号的，可以表示正数和负数
      2. Unsigned：无符号的
      3. Short：短的
      4. Long：长的
   3. C++中没有整数的取值范围，取值返回根据组合来计算得到
2. 整数的书写
   1. 二进制：书写必须以0b（B）开头
   2. 八进制:
      1. 必须以0开头
      2. 0-7
   3. 十进制
   4. 十六进制：
      1. 0x开头
3. 整数的坑
   1. 书写十进制不要在前面加上0，会被认为是八进制

第二节：longlong类型

1. vs中long是4个字节
2. liunx中long是8个字节
3. long long 类型的整数，至少是8个字节，并且与long的长度一样长

第三节：浮点型

1. 浮点型包括float 、double、long double
2. 字符数据类型；
   1. 内存占用一个字节，使用单引号
   2. 不存放字符本身，只存放编码
   3. 内存中存储都是二进制
3. ASCII：33个控制字符 95个可显示字符
4. 字符的本质
   1. 字符本质是整数，取值范围是0-127
   2. 书写的时候使用单引号，也可以使用整数
   3. 显示的时候可以是整数，也可以是整数
   4. 可以和整数进行运算
5. 原始字面量：是不需要转义，直接使用他的直接含义

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  string path = "c:\\user";  string path = R"(c:\user)";  //前后必须一致 否则会报错  string path1 = R"aaa(c:\user;)aaa";  cout << "Hello World!\n";  } |

1. 字符串
   1. C++风格：string var = “字符串”; 类
   2. C风格：char[] ch = “字符串”; 字符数组
2. 数据类型的抓换
   1. 计算机进行运算的时候，要求操作数类型相同的大小和存储方式
   2. 转换方式：
      1. 自动转换
      2. 强制转换
3. 数据类型的别名
   1. 别名的目的
      1. 为负值的类型创建别名，方便书写和记忆
      2. 创建与平台无关的数据类型，提高程序的兼容性
         1. 兼容性的写法

|  |
| --- |
| typedef long int lint;判断使用的机器 |

* + 1. 别名可以原原本本的代替原名
  1. 语法
     1. Typedef 数据类型名 别名

第五部分 指针

1. 指针
   1. C++通过&得到变量的其实地址
   2. 语法
      1. &变量名
   3. 指针用于存放变量的地址
2. 函数中使用
   1. 代码

|  |
| --- |
| void printInfo(int\* a) {  cout << \*a << endl;  \*a = -1;  }  int main()  {  int a = 0;  printInfo(&a);  cout << a << endl;  } |

可以提升性能，较少数据的copy

1. 使用常量

常量是程序中固定不变的数据

* 1. Const修饰
     1. 在程序的任何地方都可以的
     2. 使用它修饰指针
        1. 常量指针
           1. 语法：const 数据类型 \*变量名

不可以使用解指针的方法设置值

注意

指向的变量可以改变，值不可以改变（\*p =xx值不可以改变）

一般用于修饰函数的形参，表示不希望改变值

如果用于型参，虽然指向对象可以改变，但是没意义

* + - 1. 指针常量
         1. 语法：数据类型 \*const 变量名
         2. 指向的变量对象不可以改变
         3. 注意

定义的时候必须初始化，否则没意义

可以通过解引用方法修改地址中的值

代码

|  |
| --- |
| Int\* const p = &a;  \*p = 13;初始化必须设置值 并且值可以修改 |

* + - 1. 常指针常量
         1. 语法：const int \*const int
  1. 宏常量
     1. 一般在main函数之上声明的，使用大写字母
     2. 语法：

|  |
| --- |
| #define NAME C++; |

1. -void关键字
   1. C++中，void表示无类型
   2. 作用：
      1. 函数返回值void
      2. 参数设置为void,表示不需要参数
      3. -void\*作为函数参数，表示可以使用任意类型的参数
   3. 注意：
      1. 不能使用void声明变量，不可以代表一个真实的变量
      2. 不能对void\*指针直接解引用
      3. 把其它类型的指针赋值给void\*指针不需要转换
      4. 把void\*指针赋值给其他类型的需要转换
2. 内存空间
   1. 内核空间 栈空间 堆空间 常量 代码
3. 堆和栈的区别
   1. 管理方式不同：栈是自己回收的，出作用域就会自己回收，堆需要自己释放
   2. 空间大小不同：堆是物理空间决定的，栈一般只有8M
   3. 分配方式不同
   4. 是否存在碎片
4. 动态分配内存new和delete
   1. 使用堆内存的步骤
      1. 声明指针
      2. 使用new向系统申请一个内存，让指针指向这块内存
      3. 通过解引用的方法，使用这一块内存
   2. 语法：new 数据类型（初始值）
   3. 申请成功分配一块地址，如果申请失败，就是一块空地址
   4. 释放delete 地址
   5. 注意
      1. 动态分配内存没有变量名，只能通过指针来操作内存中的数据
      2. 如果动态分配内存不使用了，必须要释放
      3. 动态分配的内存生命周期与程序相同
      4. 指针作用域失效，指向的内存也不会释放
      5. 指针跟踪分配内存的时候，不可以跟丢。
5. 二级指针
6. 空指针
   1. 在c或者是c++中，都可以使用0或者NULL表示空指针，在没有赋值之前给它设置空，代表没有指向任何地址
   2. 使用空指针的后果
      1. 程序崩溃
      2. 对空指针执行delete，系统会忽略该操作，不会出现异常，内存释放后，也应该设置为null。
   3. 为什么指向空指针会崩溃
      1. Null指针分配的分区，范围是0x0000-0x0000ffff这个区域是空闲的，没有任何物理存储与之对应，任何读写都会发生异常
   4. C++11的nullptr
      1. 使用0或者NULL会产生歧义，所以建议使用（void\*)0
7. 野指针
   1. 野指针就是一个没有指向一个有效合法的地址
   2. 如果访问野指针就会发生崩溃
   3. 出现野指针的情况
      1. 指针定义的时候没有初始化，值是不确定的
      2. 使用了动态分配的内存，内存释放了，指针不会置空，但是指向已经失效
      3. 指针指向变量已经超过了作用域
   4. 规避方法
      1. 指针在定义的时候，如果无指向，就指向nullptr
8. 函数指针
   1. 函数的二进制代码存放在内存四区中代码段，函数的地址是它在内存中的起始地址，如果将函数地址作为参数传给函数，就可以灵活的调用其他函数
   2. 使用函数指针的步骤
      1. 声明函数指针
      2. 让函数指针指向函数的地址
      3. 通过函数指针调用函数
   3. 代码演示
      1. 声明函数指针
         1. 必须提供指针类型（返回值和函数列表）