|  |  |
| --- | --- |
| **组号:** | **15** |

****

信息科学与工程学院课程实验报告

《面向对象程序设计》

|  |  |
| --- | --- |
| 姓名： | 杨保亮 |
| 学号： | 201711010539 |
| 班级： | 计工一 |
| 教师： | 张庆科 |
| 时间： | 2018/10/12 |

**面向对象程序设计实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | 杨保亮 | 班级 | 计工一 | 学号 | 201711010539 | 组号 | 15 |
| 时间 | 2018/10/12 | 地点 | E312 | 周次 | 第八周 | 页码 |  |
| 源码 | □ 无源码 □ 文档源码 □ 托管源码 | | | | | | |
| 报  告  内  容  报  告  内  容  报  告  内  容 | **实验报告要求**：请围绕实验目的、实验内容、实验过程及步骤(可添加文字、矢量图)、实验结论与分析进行撰写。凡涉及源代码内容可给出完整源码并附上源码Github托管网址（请务必按照条目书写）。   1. **实验目的**   **总结第二章的知识点，更好地掌握学习第二章的知识。**   1. **实验内容**   **C++对C的改进及扩展：**  **2.1函数中一些基本控制的区别**  **2.11输入输出**   1. **函数是构成C++程序的基本单位。** 2. **程序生成的四个步骤：编辑、编译、连接、执行。** 3. **#include<iostream>是预处理命令，在编译时就执行，没有对应的机器指令。** 4. **对于计算机硬件一般使用“兼容”，对于软件一般使用“移植”。** 5. **C++中，将一个数据从一个对象到另一个对象的流动抽象为流，数据的输入输出是通过I/O流来实现的。C++中输入输出分别用cin/cout,使用这两个流类对象必须含有正确的文件包含#include<iostream>及using namespace std;或者#include<iostream.h>。输入要使用提取符”>>”,输出使用插入符”<<”。** 6. **输入时提取符后面只能跟变量，用户在键盘输入时，两个数据之间默认要用空白符（空格、回车、Tab建）分隔，** 7. **换行符有两种”\n”和”endl”**   **2.12注释方式：”//”是C++新增的单行注释方式，注释内容从”//”开始到本行结束，只对单行有效，适合注释内容不超过一行的情况。C语言中的”/\*.....\*/”在C++中仍然适用。嵌套使用遵循以下四条原则：**   1. **/\*.......\*/方式的注释不能嵌套。** 2. **//方式的注释可以嵌套。** 3. **//方式下可以嵌套/\*....\*/注释。** 4. **/\*.....\*/方式下可以嵌套//注释。**   **2.13 const定义常量**  **1、宏定义存在一个缺陷，对常量只做简单的替换，不进行类型检查。Const则不会出现这种效果。**  **2、const的效果有三种：保护常量、保护指针、双保护。**  **Const定义的形式为：**  **Const [常量类型]符号常量=表达式；**   1. **三种指针用法：** 2. **指向常量的指针：**   **Const int \*p=&x**  **（常量不可改变）**   1. **常指针**   **Int \*const p=&x;**  **指针不可改变**   1. **指向常量的常指针：**   **Const int \* const p=&x;**  **指针，常量都不可改变。**  **2.14 强制类型转换**  **（1）将目标类型名称用一对圆括号括起，后面接需要转换的源表达式，源表达式有运算符，则源表达式本身也需要用一对圆括号括起，例如：x=(int)(f+23.4)。**  **（2）目标类型无需加括号，源表达式必须加括号。例如：x=int(f)。**  **（3）类型转换时，由所占字节数大的类型转换为所占字节数小的类型可以。但是由所占字节数小的转换为所占字节数大的则会造成精度缺失。**  **2.15 新增bool（布尔）类型**  **bool类型用常量ture表示逻辑真，用false表示逻辑假。所有的关系运算、逻辑运算都产生bool类型的结果。在默认情况下，bool表达式的值为ture时输出1，值为false时输出0.可以使用C++标准库中提供的boolalpha操纵符使其输出为ture或者false。也可以使用noboolapha操纵符是输出恢复为0/1.**  **2.16 名字空间**  **1、名字空间的定义为：namespace 名字空间 {....; }**  **2、名字空间的用法**  **（1）先用 *using namespace 名字空间名称*，之后程序中可以直接使用名字空间中的内容而无需加任何限定标记。**  **（2）无需事先声明，在需要用的地方以*名字空间称：：局部内容名*的形式使用**  **“：：”称为域解析符或作用域运算符。**  **（3）先用 *using namespace 名字空间名称：：局部内容名*；声明这一局部内容，之后便可以直接使用这一局部内容名。**   1. **函数内部变量储存在栈内，后进后出。在程序中定义多个变量，全局变量最后输出，局部变量优先输出。**   **2.2 函数区别**  **2.2.1 局部变量**  **局部变量随用随定义，局部变量的定义和声明可以在程序块的任何位置，变量的作用域为从定义点到该变量所在的最小程序块末的范围。也可以出现同名变量，按最近范围变量优先的原则处理。**  **2.2.2 域解析符**  **两个同名变量一个是全局变量，一个是局部变量。在同名变量前加上域解析符“：：”，对被隐藏的同名全局变量进行访问。**  **2.2.3 带默认值的形参**  **允许在函数原型声明为一个或多个形式参数指定默认参数值。在调用函数时，如果不提供实际参数，形参直接使用默认值。实参的个数小于等于形参。**  **2.2.4 内联函数**  **功能较为简单的函数可以采用宏定义，但预处理进行宏扩展时，只进行简单的文本替换，不考虑代码的语义。**  **在C++中增加了内联函数，在一个函数定义首部的最前面增加关键字inline，该函数就变成了内联函数。内联函数在进行编译时，必须考虑代码语义。**  **内联函数内部不允许使用循环语句和开关语句，还不能使用递归。定义在类内的成员函数都自动理解为内联函数，前面无需加inline关键词。**  **2.2.5 函数重载**  **只是在形式参数的个数、类型、顺序方面有区别的不同函数以相同的函数名来命名，该同名函数被称作重载**  **特别注意：重载函数与带默认值的函数一起使用时，有可能引起二义性。**  **2.3 引用**  **2.3.1 变量的引用：**  **引用：就是某一变量（目标）的一个别名，对引用的操作与对变量直接操作完全一样。**  **引用的声明方法：类型标识符 &引用名=目标变量名；**  **例：char ch;**  **char &rp=ch;**  **1、引用仅是变量的别名，而不是实实在在地定义了一个变量，因此引用本身并不占用内存，而是和目标变量共同指向目标变量的内存地址.**  **2、表达式中的取地址符&不再是取变量的地址，而是用来表示该变量是引用类型的变量。**  **3、定义一个引用时，必须对其初始化。**  **2.3.2 引用作为参数**  **引用的一个重要作用就是作为函数的参数。以前的C语言中函数参数传递是值传递，如果有大块数据作为参数传递的时候，采用的方案往往是指针，因为这样可以避免将整块数据全部压栈，可以提高程序的效率。但是现在（C++中）又增加了一种同样有效率的选择（在某些特殊情况下又是必须的选择），就是引用。**  **（1）使用引用传递函数的参数，在内存中并没有产生实参的副本，它是直接对实参操作；而使用一般变量传递函数的参数，当发生函数调用时，需要给形参分配存储单元，形参变量是实参变量的副本；如果传递的是对象，还将调用拷贝构造函数。因此，当参数传递的数据较大时，用引用比用一般变量传递参数的效率和所占空间都好。**  **（2）使用指针作为函数的参数虽然也能达到与使用引用的效果，但是，在被调函数中同样要给形参分配存储单元，且需要重复使用\"\*指针变量名\"的形式进行运算，这很容易产生错误且程序的阅读性较差；另一方面，在主调函数的调用点处，必须用变量的地址作为实参。而引用更容易使用，更清晰。**  **如果既要利用引用提高程序的效率，又要保护传递给函数的数据不在函数中被改变，就应使用常引用。**  **就如同上边的例子一样。**  **2、常引用**  **常引用声明方式：const  类型标识符  &引用名 = 目标变量名；**  **用这种方式声明的引用，不能通过引用对目标变量的值进行修改,从而使引用的目标成为const，达到了引用的安全性。**  **3、引用作为返回值**  **要以引用返回函数值，则函数定义时要按以下格式：**  **类型标识符  &函数名 （形参列表及类型说明）**  **｛  函数体  ｝**  **说明：**  **1、以引用返回函数值，定义函数时需要在函数名前加&**  **2、用引用返回一个函数值的最大好处是，在内存中不产生被返回值的副本。**  **以下程序中定义了一个普通的函数fn1（它用返回值的方法返回函数值），另外一个函数fn2，它以引用的方法返回函数值。**  **2.4 异常处理**  **异常是程序在执行期间产生的问题.C ++异常是指在程序运行时发生的特殊情况，比如尝试除以零的操作。**  **异常提供了一种转移程序控制权的方式.C ++异常处理涉及到三个关键字：try，catch，throw。**  **抛出：当问题出现时，程序会抛出一个异常这是通过使用。扔关键字来完成的。**  **catch：在您想要处理问题的地方，通过异常处理程序捕获异常。捕获关键字用于捕获异常。**  **尝试： 尝试块中的代码标识将被激活的特定异常。它后面通常跟着一个或多个catch块。**  **如果有一个块抛出一个异常，捕获异常的方法会使用try和catch关键字.try块中放置可能抛出异常的代码，尝试块中的代码被称为保护代码。** | | | | | | |
|  |  | | | | | | |

：可根据内容自行拓展页面，作业内容尾部尽量不要留有空白