**中国矿业大学计算机学院实验报告**

|  |
| --- |
| 课程名称  高级语言程序设计           实验名称 类与对象  班级 计算机科学与技术2019-3班 姓名 王杰永 学号 03190886  仪器组号\_\_\_\_\_\_\_  实验日期 2020.12.3  实验报告要求：1.实验目的   2.实验内容（题目描述，源代码，运行截图，调试情况）    3.实验体会 |
| 一、实验目的  （1）掌握类的声明、对象定义以及类对象的访问权限。  （2）掌握访问类的数据成员和成员函数的方法。  （3）掌握构造函数和析构函数的定义以及使用。  二、实验内容  1、第一题  1.1题目描述  题目描述  自定义一个复数类型Complex，其中含有若干成员函数，使用该类可以完成复数的加法以及对复数的输出。请完成类定义，并编制主函数，说明Complex类对象，对定义的各成员函数进行调用。  class Complex {         double real; //复数实部         double imag; //复数虚部  public:         Complex (); //无参构造函数，将复数对象的实部和虚部均置为0         Complex (double r, double i); //有参构造函数，设置对象的实部和虚部         Complex AddCom(Complex c2); //调用者对象与对象c2相加，返回Complex类对象         void OutCom () ; //输出调用者对象的有关数据（各分量）  };  具体 要求如下：  1、实现有参构造函数 Complex (double r, double i);  2、实现 Complex AddCom(Complex c2); 调用者对象与对象c2相加，返回Complex类对象  3、实现void OutCom () ;实现输出调用者对象的有关数据分量（一定要输出虚部的符号i），如果该数为纯虚数时,不需要输出实部，当虚部为0时，不需要输出虚部。  4．编制主函数main，作用有参函数说明类对象cx,cy，使用 Complex 调用AddCom实现复数加法，并将相加的结果调用 OutCom方法以复数的形式输出。  输入  输入包括a,b,c,d四个整数，第一个复数为a+bi，第二个复数为c+di  样例输入 [Copy](javascript:CopyToClipboard($('#sampleinput').text()))  1 2 3 4  样例输出 [Copy](javascript:CopyToClipboard($('#sampleoutput').text()))  4+6i  1.2 源代码  #include<iostream>  using namespace std;  class Complex {  private:  double real;  double imag;  public:  Complex() { real = 0; imag = 0; }  Complex(double r, double i);  Complex AddCom(Complex c2);  void OutCom();  };  Complex::Complex(double r, double i) {  real = r;  imag = i;  }  Complex Complex::AddCom(Complex c2) {  this->real += c2.real;  this->imag += c2.imag;  return \*this;  }  void Complex::OutCom() {  if (real != 0 && imag != 0)  cout << real << "+" << imag << "i";  else if (real == 0 && imag != 0)  cout << imag << "i";  else if (real != 0 && imag == 0)  cout << real;  }  int main() {  int a, b, c, d;  cin >> a >> b >> c >> d;  Complex cx(a, b), cy(c, d);  cx.AddCom(cy);  cx.OutCom();  return 0;  }  1.3 运行截图    1.4 调试情况  本题运行较为顺利。  2、第二题  2.1题目描述      2.2 源代码  #include<iostream>  using namespace std;  int month[12] = { 31,28,31,30,31,30,31,31,30,31,30,31 };  class DateType {  int y, m, d;  public:  DateType(int y = 1, int m = 1, int d = 1) {  this->y = y;  this->m = m;  this->d = d;  }  void IncrementDay();  bool Equal(DateType dt2);  void PrintDate();  };  void DateType::IncrementDay() {  if (y % 4 == 0 && y % 100 != 0 || y % 400 == 0)  month[1] = 29;  if (d == month[m - 1]) {  d = 1;  if (m == 12) {  m = 1;  y++;  }  else  m++;  }  else  d++;  month[1] = 28;  }  bool DateType::Equal(DateType dt2) {  if (this->y == dt2.y && this->m == dt2.m && this->d == dt2.d)  return true;  else  return false;  }  void DateType::PrintDate() {  cout << y << ":" << m << ":" << d;  }  int main() {  int y1, m1, d1, y2, m2, d2;  cin >> y1 >> m1 >> d1 >> y2 >> m2 >> d2;  DateType dt1(y1, m1, d1), dt2(y2, m2, d2);  dt1.PrintDate(); cout << endl;  dt2.PrintDate(); cout << endl;  if (dt1.Equal(dt2))  cout << "True" << endl;  else  cout << "False" << endl;  dt1.IncrementDay();  dt2.IncrementDay();  dt1.PrintDate(); cout << endl;  dt2.PrintDate(); cout << endl;  return 0;  }  2.3 运行截图    2.4 调试情况  在日期的自增函数中，需要注意进位情况。不仅有一个月的31号需要进位，还需要考虑每一年的最后一天也需要进位。  3、第三题  3.1题目描述      3.2 源代码  #include<iostream>  using namespace std;  class CStudent {  unsigned long reg\_num;  char name[30];  float math, eng, comp;  public:  float Sum();  float Average();  void Display();  void SetData();  };  float CStudent::Sum() {  return math + eng + comp;  }  float CStudent::Average() {  return Sum() / 3;  }  void CStudent::Display() {  cout << reg\_num << " " << name << " " << math << " " << eng << " " << comp;  }  void CStudent::SetData() {  cin >> reg\_num >> name >> math >> eng >> comp;  }  int main() {  CStudent stu[150];  int TOTAL;  cin >> TOTAL;  for (int i = 0; i < TOTAL; i++) {  stu[i].SetData();  }  int idx\_max = 0;  cout << "TOTAL=" << TOTAL << endl;  for (int i = 0; i < TOTAL; i++) {  cout << "CStudent " << i + 1 << " : ";  stu[i].Display(); cout << endl;  }  for (int i = 0; i < TOTAL; i++) {  if (stu[idx\_max].Sum() < stu[i].Sum())  idx\_max = i;  cout << "CStudent" << i + 1 << ".Sum=" << stu[i].Sum();  cout << "," << "CStudent" << i + 1 << ".average=" << stu[i].Average();  cout << endl;  }  cout << "class\_Sum\_max=" << stu[idx\_max].Sum() << endl;  cout << "The infomation of the CStudent with class\_Sum\_max : ";  stu[idx\_max].Display();  system("pause");  return 0;  }  3.3 运行截图    3.4 调试情况  这道题做起来并不难，最开始提交的一次错误是理解错了题意。  最初认为，首先输出一个”TOTAL=”,之后再cin>>TOTAL;，从而在控制台上出现 TOTAL=3 这样的信息，下面也是这样的理解，导致输出错误。  4、第四题  4.1题目描述  OJ上无法看到原题目，找不到题目描述了。  大概就是创建一个Point类，和一个Circle类，其中Circle类的数据成员中包含一个Point类的对象。  最后要求重载运算符<<和>>，实现对两个类的对象的输出。  4.2 源代码  #include<iostream>  using namespace std;  class Point {  int x, y;  public:  Point() {};  Point(int x, int y) {  this->x = x;  this->y = y;  }  friend istream& operator>>(istream& is, Point& p);  friend ostream& operator<<(ostream& os, Point p);  };  istream& operator>>(istream& is, Point& p) {  is >> p.x >> p.y;  return is;  }  ostream& operator<<(ostream& os, Point p) {  os << "(" << p.x << "," << p.y << ")";  return os;  }  class Circle {  Point O;  int r;  public:  Circle(int a, int b, int r) {  O = \*new Point(a, b);  this->r = r;  }  friend istream& operator>>(istream& is, Circle& c);  friend ostream& operator<<(ostream& os, Circle c);  };  istream& operator>>(istream& is, Circle& c) {  cin >> c.O >> c.r;  return is;  }  ostream& operator<<(ostream& os, Circle c) {  cout << c.O << endl << c.r;  return os;  }  int main() {  Point p(0, 0);  cin >> p;  cout << p << endl;  Circle c(0, 0, 0);  cin >> c;  cout << c << endl;  return 0;  }  4.3 运行截图    4.4 调试情况  这道题把自己大一时没有学精的运算符重载重新复习了一下，尤其是从未尝试过对流运算符重载。  通过这道题深刻理解了为什么<<和>>只能重载为非成员函数，即友元函数。  在调试的过程中，最初重载时传递的参数均为值传递，忽略了需要引用传递，导致<<重载成功，但是>>得不到正确答案。最后才明白，重载>>必须使用引用传递。  三、实验体会  本次实验加深了我对类与对象的理解，熟练了如何使用类的对象以及其成员函数。对流运算<<和>>有了更深刻的理解。收获颇丰！ |