**实验二 类和对象的基本概念和基础知识（一）**

* 1. **实验目的：**

1、熟悉C++语言的运行环境，程序的编辑、编译和运行的过程。

2、掌握类和对象的定义方法，对象成员的表示方法及对象初始化的方法。

* 1. **实验设备：**装有VC6.0运行环境的计算机。
  2. **实验要求：**

1**、**运行课上例题（例5.1）“日期类的定义和对象的定义”程序。

2、运行课上例题（例5.3）“日期类用构造函数实现对象初始化”程序。

3、运行课上例题（例5.5）“复制构造函数”程序。

**四、实验报告：**

1、分析程序的理论运行结果，并与实际运行结果进行比较。

2、总结如何实现对象的初始化。

**实验报告**



学院（系）名称：机械工程学院

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 张春晓 | 学号 | 20190541 | 专业 | 机器人工程 |
| 班级 | 一班 | 实验项目 | 实验二 类和对象的基本概念和基础知识（一） |
| 课程名称 | VC++语言程序设计 | 课程代码 | 0106306 |
| 实验时间 | 10.5 | 实验地点 | 15—601 |
| 批改意见 | 成绩 |  |
| 教师签字： |
| 实验目的：  实验内容  1、运行课上例题“日期类的定义和对象的定义”程序。  2、运行课上例题“日期类用构造函数实现对象初始化”程序。  3、运行课上例题（例5.5）“拷贝初始化构造函数”程序。  程序清单及结果：  #include<iostream.h>  #include"tdate.h"  void main()  {  TDate date1,date2;  date1.SetDate(1996,5,4);  date2.SetDate(1998,4,9);  int leap=date1.IsLeapYear();  cout<<leap<<endl;  date1.Print();  date2.Print();  }  class TDate  {  public:  void SetDate(int y,int m,int d)  {  year=y;  month=m;  day=d;  }  int IsLeapYear()  {  return(year%4==0&&year%100!=0)||(year%400==0);  }  void Print()  {cout<<year<<"."<<month<<"."<<day<<endl;}  private:  int year,month,day;  };  #include<iostream.h>  #include"tdate1.h"  void main()  {  TDate1 today(1998,4,9),tomorrow(1998,4,10);  cout<<"today is ";  today.Print();  cout<<"tomorrow is ";  tomorrow.Print();  }  class TDate1  {  public:  TDate1(int y,int m,int d);  ~TDate1();  void Print();  private:  int year,mouth,day;  };  TDate1::TDate1(int y,int m,int d)  {  year=y;  mouth=m;  day=d;  cout<<"Constructor called.\n";  }  TDate1::~TDate1()  {  cout<<"Constructor called.\n";  }  void TDate1::Print()  {  cout<<year<<"."<<mouth<<"."<<day<<endl;  }  #include<iostream.h>  #include"tpoint1.h"  TPoint f(TPoint Q);  void main()  {  TPoint M(20,35),P(0,0);  TPoint N(M);  P=f(N);  cout<<"P="<<P.Xcoord()<<","<<P.Ycoord()<<endl;  }  TPoint f(TPoint Q)  {  cout<<"ok\n";  int x,y;  x=Q.Xcoord()+10;  y=Q.Xcoord()+20;  TPoint R(x,y);  return R;  }  class TPoint  {  public:  TPoint(int x,int y)  {X=x;Y=y;}  TPoint(TPoint & p);  ~TPoint()  {cout<<"Destructor Called.\n";}  int Xcoord()  {return X;}  int Ycoord()  {return Y;}  private:  int X,Y;  };  TPoint::TPoint(TPoint & p)  {  X=p.X;  Y=p.Y;  cout<<"Copy\_initialization Constructor Called.\n";  }  实验总结： |