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	高级语言程序设计实践	
班级	信息安全 2019-1 班	姓名	李春阳	学号	10193657
仪器组号			实验日期	2020. 12. 16	
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. 认识了解 c++基本语法。
2. 掌握条件语句和循环语句
3. 掌握数组和字符串的表达。
4. 掌握类的基本用法。

二、实验内容

1、第一题

1.1 题目描述

A. 问题描述

对本章示范题的用于管理商店商品的实现程序进行完善:完成 Wardrobe 立柜类的具体定义与使用，并添加“帽子仓库类”以及“立柜仓库类”的定义及使用，以使程序能够对商店的这三种商品（衬衣、帽子、立柜）进行简单的管理与应用。

要对商品实现的操作有:商品的进库（增加某类商品及其库存量），商品的出库（减少某类商品及其库存量），以及某类商品总价格的计算。

最终显示效果：(红色部分是输入数据)

```
5 * shirt data in: price/place/material =>

60 Tianjin Cotton

3 * shirt data in: price/place/material =>

80 Beijing Wool

60 Tianjin Cotton

60 Tianjin Cotton

60 Tianjin Cotton

60 Tianjin Cotton
```

```
60 Tianjin Cotton

80 Beijing Wool

80 Beijing Wool

80 Beijing Wool

shiSto.TotalPrice()=540

60 Tianjin Cotton

60 Tianjin Cotton

60 Tianjin Cotton

60 Tianjin Cotton

shiSto.TotalPrice()=240

5 * Cap data in: price/place/material/style =>

40 Suzhou Cotton M

3 * Cap data in: price/place/material/style =>

30 Wuxi Wool S

40 Suzhou Cotton M

40 Suzhou Cotton M

40 Suzhou Cotton M

40 Suzhou Cotton M

40 Suzhou Cotton M

30 Wuxi Wool S

30 Wuxi Wool S

30 Wuxi Wool S

capSto.TotalPrice()=290
```

```
40 Suzhou Cotton M

40 Suzhou Cotton M

40 Suzhou Cotton M

40 Suzhou Cotton M

capSto.TotalPrice()=160

5 * Wardrobe data in: price/place/material/color =>

160 Guangzhou Pine Yellow

3 * Wardrobe data in: price/place/material/color =>

200 Suzhou Oak Brown

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

200 Suzhou Oak Brown

200 Suzhou Oak Brown

200 Suzhou Oak Brown

WarSto.TotalPrice()=1400

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

160 Guangzhou Pine Yellow

WarSto.TotalPrice()=640
```

B.　输入

```
60 Tianjin Cotton
80 Beijing Wool
40 Suzhou Cotton M
30 Wuxi Wool S
160 Guangzhou Pine Yellow
200 Suzhou Oak Brown
```

C. 输出

```
5 * shirt data in: price/place/material =>
3 * shirt data in: price/place/material =>
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
80 Beijing Wool
80 Beijing Wool
80 Beijing Wool
shiSto.TotalPrice()=540
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
shiSto.TotalPrice()=240
5 * Cap data in: price/place/material/style =>
3 * Cap data in: price/place/material/style =>
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
30 Wuxi Wool S
30 Wuxi Wool S
30 Wuxi Wool S
capSto.TotalPrice()=290
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
capSto.TotalPrice()=160
5 * Wardrobe data in: price/place/material/color =>
3 * Wardrobe data in: price/place/material/color =>
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
```

```
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
200 Suzhou Oak Brown
200 Suzhou Oak Brown
200 Suzhou Oak Brown
WarSto.TotalPrice()=1400
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
160 Guangzhou Pine Yellow
WarSto.TotalPrice()=640
```

1.2 源代码

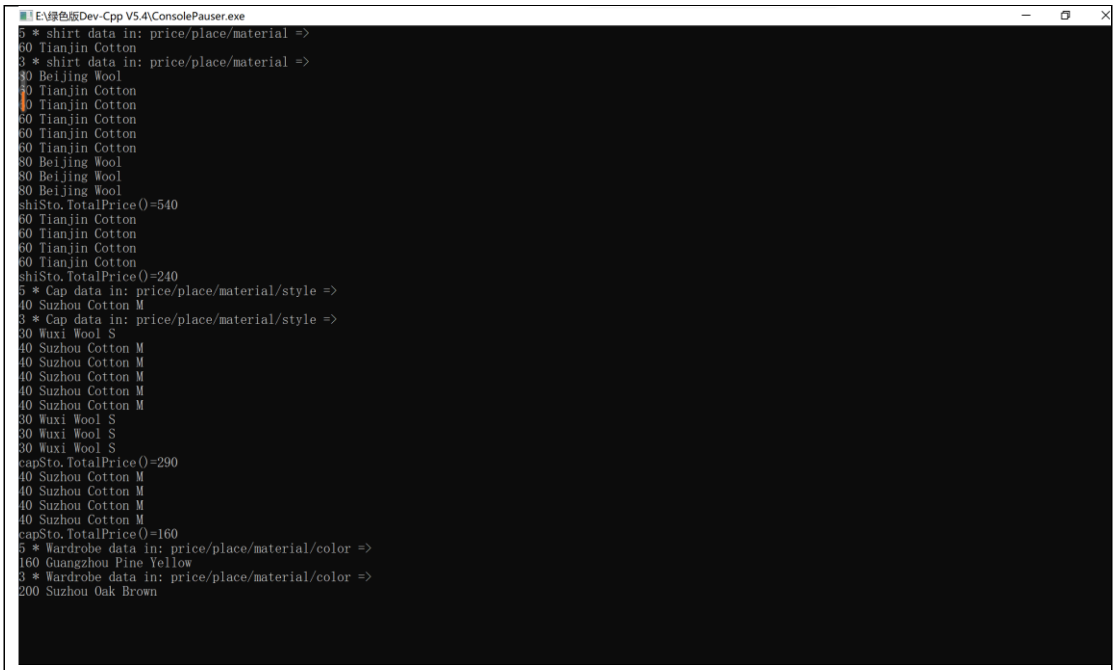
```
#include<iostream>
#include<cstring>
using namespace std;
class Base {
private:
    double price;
    char place[20];
    int count;
public:
    Base(double pr, char* pl, int cnt)
    {
        price = pr;
        strcpy(place, pl);
        count = cnt;
    }
    void display()
    {
        cout << price << " " << place << " ";
    }
    void InSomething(int add_cnt)
    {
        count += add_cnt;
    }
    void OutSomething(int del_cnt)
    {
        count -= del_cnt;
    }
    double TotalPrice()
    {
        return price;
    }
};
```

```
class Shirt :public Base {
private:
    char material[20];
public:
    Shirt(double pr, char* pl, int cnt, char* mat) :Base(pr, pl, cnt)
    {
        strcpy(material, mat);
    }
    void display()
    {
        Base::display();
        cout << material << " ";
    }
};
class Cap :public Shirt {
private:
    char style;
public:
    Cap(double pr, char* pl, int cnt, char* mat, char sty) :Shirt(pr, pl, cnt, mat)
    {
        style = sty;
    }
    void display() {
        Shirt::display();
        cout << style;
    }
};
class Wardrobe :public Base {
private:
    char material[20];
    char color[20];
public:
    Wardrobe(double pr, char* pl, int cnt, char* mat, char* col) :Base(pr, pl, cnt)
    {
        strcpy(material, mat);
        strcpy(color, col);
    }
    void display()
    {
        Base::display();
        cout << material << " " << color;
    }
};
int main()
```

```
{
    int price;
    char place[20];
    char material[20];
    char style;
    char color[20];
    cout << "5 * shirt data in: price/place/material =>" << endl;
    cin >> price >> place >> material;
    Shirt s1(price, place, 5, material);
    cout << "3 * shirt data in: price/place/material =>" << endl;
    cin >> price >> place >> material;
    Shirt s2(price, place, 3, material);
    for (int i = 0; i < 5; i++)
    {
        s1.display();
        cout << endl;
    }
    for (int i = 0; i < 3; i++)
    {
        s2.display();
        cout << endl;
    }
    cout << "shiSto.TotalPrice()=" << (s1.TotalPrice()) * 5 + (s2.TotalPrice()) * 3
<< endl;
    for (int i = 0; i < 4; i++)
    {
        s1.display();
        cout << endl;
    }
    cout << "shiSto.TotalPrice()=" << (s1.TotalPrice()) * 4 << endl;
    cout << "5 * Cap data in: price/place/material/style =>" << endl;
    cin >> price >> place >> material >> style;
    Cap c1(price, place, 5, material, style);
    cout << "3 * Cap data in: price/place/material/style =>" << endl;
    cin >> price >> place >> material >> style;
    Cap c2(price, place, 3, material, style);
    for (int i = 0; i < 5; i++)
    {
        c1.display();
        cout << endl;
    }
    for (int i = 0; i < 3; i++)
    {
        c2.display();
        cout << endl;
    }
    cout << "capSto.TotalPrice()=" << (c1.TotalPrice()) * 5 + (c2.TotalPrice()) * 3
<< endl;
    for (int i = 0; i < 4; i++)
    {
        c1.display();
        cout << endl;
    }
    cout << "capSto.TotalPrice()=" << (c1.TotalPrice()) * 4 << endl;
    cout << "5 * Wardrobe data in: price/place/material/color =>" << endl;
    cin >> price >> place >> material >> color;
    Wardrobe w1(price, place, 5, material, color);
    cout << "3 * Wardrobe data in: price/place/material/color =>" << endl;
    cin >> price >> place >> material >> color;
    Wardrobe w2(price, place, 3, material, color);
    for (int i = 0; i < 5; i++)
    {
        w1.display();
        cout << endl;
    }
    for (int i = 0; i < 3; i++)
    {
        w2.display();
        cout << endl;
    }
    cout << "WarSto.TotalPrice()=" << (w1.TotalPrice()) * 5 + (w2.TotalPrice()) * 3
<< endl;
    for (int i = 0; i < 4; i++)
    {
        w1.display();
        cout << endl;
    }
    cout << "WarSto.TotalPrice()=" << (w1.TotalPrice()) * 4;

    return 0;
}
```

1.3 运行截图



E:\绿色版Dev-Cpp V5.4\ConsolePauser.exe

```
5 * shirt data in: price/place/material =>
60 Tianjin Cotton
3 * shirt data in: price/place/material =>
80 Beijing Wool
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
80 Beijing Wool
80 Beijing Wool
80 Beijing Wool
shiSto.TotalPrice()=540
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
60 Tianjin Cotton
shiSto.TotalPrice()=240
5 * Cap data in: price/place/material/style =>
40 Suzhou Cotton M
3 * Cap data in: price/place/material/style =>
30 Wuxi Wool S
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
30 Wuxi Wool S
30 Wuxi Wool S
30 Wuxi Wool S
capSto.TotalPrice()=290
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
40 Suzhou Cotton M
capSto.TotalPrice()=160
5 * Wardrobe data in: price/place/material/color =>
160 Guangzhou Pine Yellow
3 * Wardrobe data in: price/place/material/color =>
200 Suzhou Oak Brown
```

1.4 调试情况

2、第二题

2.1 题目描述

A. 问题描述、

利用继承性与派生类来管理学生教师档案:由 Person（人员）类出发（作为基类），派生出 Student（学生）及 Teacher（教师）类；而后又由 Student（学生）类出发（作为基类），派生出 GraduateStudent（研究生）类。可假定这几个类各自具有的数据成员为:

Person（人员）类: 姓名、性别、年龄；

Student（学生）类: 姓名、性别、年龄、学号、系别；

Teacher（教师）类: 姓名、性别、年龄、职称、担任课程；

GraduateStudent（研究生）类: 姓名、性别、年龄、学号、系别、导师。

为简化起见，每个类可只设立构造函数以及显示类对象数据的成员函数 Print。而后编制简单的主函数，说明上述有关的类对象，并对其类成员函数进行简单使用（调用）

B. 输出

```
== per1.Display() => name,age,sex
sun 42 M
== stu1.Display() => name,age,sex,Reg_Number,department
guo 22 F 1001 comp
== teach1.Display() => name,age,sex,course,post
fang 38 M english professor
== gStu.Display() => name,age,sex,Reg_Number,department,advisor
```

wu 25 M 1021 comp wei

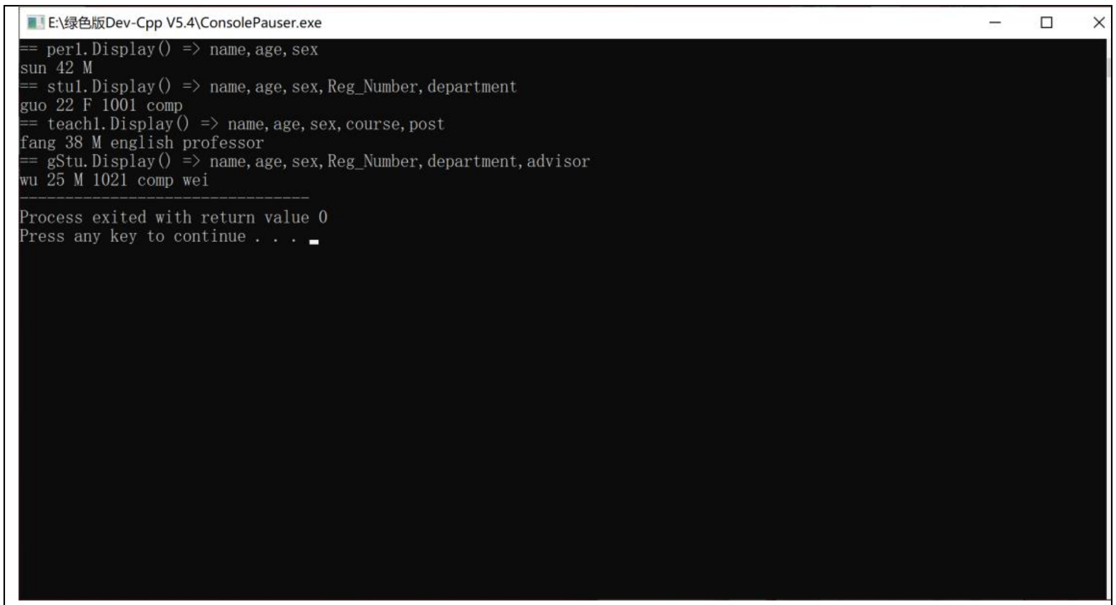
2.2 源代码

```
#include<iostream>
#include<string>
using namespace std;

class Person {
protected:
    string name;
    int age;
    string sex;
public:
    Person() {}
    Person(string n, int a, string s) :name(n), age(a), sex(s) {}
    void Display()
    {
        cout << name << " " << age << " " << sex;
    }
};
class Student :public Person {
protected:
    int regnum;
    string department;
public:
    Student() {}
    Student(string n, int a, string s, int r, string d)
    {
        name = n; age = a; sex = s; regnum = r; department = d;
    }
    void Display()
    {
        Person::Display();
        cout << " " << regnum << " " << department;
    }
};
class GraduateStudent :public Student {
protected:
    string advisor;
public:
    GraduateStudent() {};
    GraduateStudent(string n, int a, string s, int r, string d, string
ad) :Student(n, a, s, r, d), advisor(ad) {}
    void Display()
    {
        Student::Display();
        cout << " " << advisor;
    }
};
class Teacher :public Person {
protected:
    string course;
    string post;
public:
    Teacher(string n, int a, string s, string c, string p) :Person(n, a, s),
course(c), post(p) {}
    void Display()
    {
        Person::Display();
        cout << " " << course << " " << post;
    }
};

int main()
{
    cout << "== per1.Display() => name,age,sex" << endl;
    Person per1("sun", 42, "M");
    per1.Display(); cout << endl;
    cout << "== stu1.Display() => name,age,sex,Reg_Number,department" << endl;
    Student stu1("guo", 22, "F", 1001, "comp");
    stu1.Display(); cout << endl;
    cout << "== teach1.Display() => name,age,sex,course,post" << endl;
    Teacher teach1("fang", 38, "M", "english", "professor");
    teach1.Display(); cout << endl;
    cout << "== gStu.Display() => name,age,sex,Reg_Number,department,advisor" <<
endl;
    GraduateStudent gStu("wu", 25, "M", 1021, "comp", "wei");
    gStu.Display();
    return 0;
}
```

2.3 运行截图


E:\绿色版Dev-Cpp V5.4\ConsolePauser.exe

```
== per1.Display() => name,age,sex
sun 42 M
== stu1.Display() => name,age,sex,Reg_Number,department
guo 22 F 1001 comp
== teach1.Display() => name,age,sex,course,post
fang 38 M english professor
== gStu.Display() => name,age,sex,Reg_Number,department,advisor
wu 25 M 1021 comp wei
--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

2.4 调试情况

3、第三题

3.1 题目描述

A. 问题描述

自定义一个日期时间类 DateTimeType，它含有类 DateType 与类 TimeType 的类对象作为其数据成员，并具有所列的其他几个成员函数。而后编制主函数，说明 DateTimeType 的类对象，并对其成员函数以及二对象成员所属类的公有成员函数进行使用。

```
class DateTimeType {   //自定义的日期时间类 DateTimeType

DateType date; //类 DateType 的类对象 date 作为其数据成员

TimeType time; //类 TimeType 的类对象 time 作为其另一个数据成员

public:

DateTimeType(int y0=1, int m0=1, int d0=1, int hr0=0, int mi0=0, int se0=0);

//构造函数，设定 DateTimeType 类对象的日期时间，并为各参数设置了默认值

DateType& GetDate(){ return date; } //返回本类的私有数据对象 data

TimeType& GetTime(){ return time; } //返回本类的私有数据对象 time

void IncrementSecond(int s);   //增加若干秒，注意“进位”问题

void PrintDateTime(); //屏幕输出日期时间对象的有关数据
```

};

注意，每一个 DateTimeType 类对象中总包含有一个 DateType 类对象（对象成员）以及一个 TimeType 类对象（对象成员），编制与实现本程序时，也必须包含 DateType 与 TimeType 自定义类（类型）。

之所以设置了公有的类成员函数 GetDate 与 GetTime，是为类外如主函数处使用该类的私有数据成员 date 与 time 提供方便（否则的话，类外无法直接访问该类的私有数据成员）。另外，两成员函数返回的都为引用，为的是可将返回对象当作一个独立变量来使用（如可以继续作左值等）。例如，假设编制了如下形式的主函数:

```
void main() {

DateTimeType dttm1(1999,12,31,23,59,59), dttm2;

(dttm1.GetDate()).PrintDate(); //调用对象成员所属类的公有成员函数

cout<<endl;

dttm1.PrintDateTime(); //调用本派生类的成员函数  PrintDateTime

dttm2.PrintDateTime();

dttm1.IncrementSecond(30) ; //调用本派生类成员函数

dttm1.PrintDateTime();

}
```

B. 输出

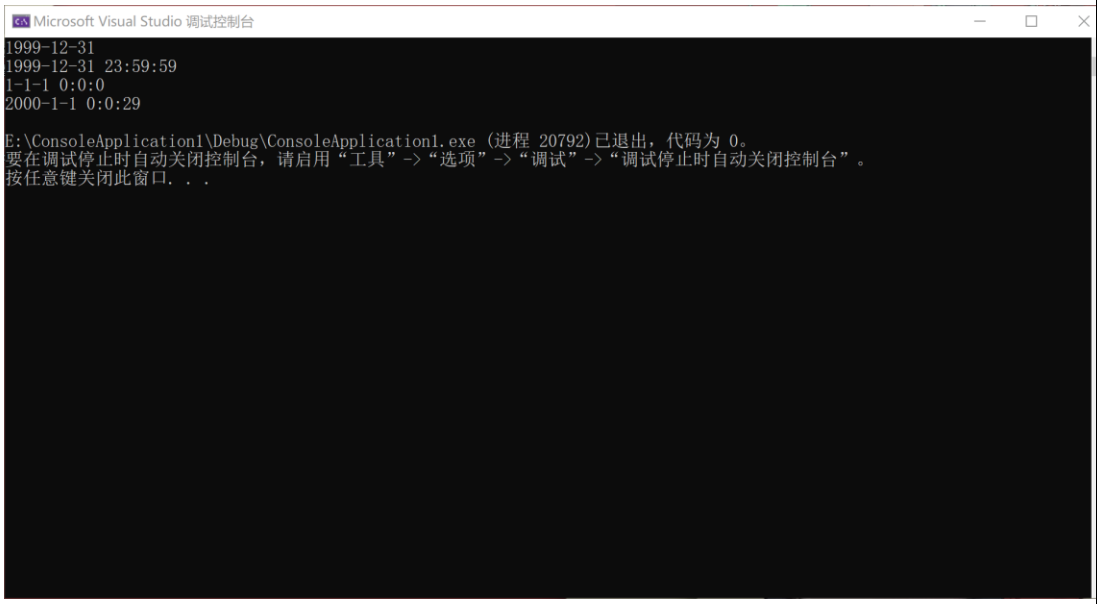
```
1999-12-31
1999-12-31 23:59:59
1-1-1 0:0:0
2000-1-1 0:0:29
```

3.2 源代码

```
#include<iostream>
using namespace std;
class DateType
{
public:
    int year;
    int month;
    int day;
    DateType(int y, int m, int d)
    {
        year = y; month = m; day = d;
    }
    void PrintDate()
    {
        cout << year << "-" << month << "-" << day << endl;
    }
};
class TimeType
{
public:
    int hour;
    int minute;
    int second;
    TimeType(int h, int m, int s)
    {
        hour = h; minute = m; second = s;
    }
};
class DateTimeType
{
    DateType date;
    TimeType time;

public:
    DateTimeType(int y0 = 1, int m0 = 1, int d0 = 1, int hr0 = 0, int mi0 = 0, int
se0 = 0) : date(y0, m0, d0), time(hr0, mi0, se0) {}
    DateType& GetDate()
    {
        return date;
    }
    TimeType& GetTime()
    {
        return time;
    }
    void IncrementSecond(int s);
    void PrintDateTime()
    {
        cout << date.year << "-" << date.month << "-" << date.day << " " << time.hour
<< ":" << time.minute << ":" << time.second << endl;
    }
};
void DateTimeType::IncrementSecond(int s)
{
    int& m = date.month;
    time.second += s;
    if (time.second >= 60)
    {
        time.minute += time.second / 60;
        time.second = time.second % 60;
        if (time.minute >= 60)
        {
            time.hour += time.minute / 60;
            time.minute = time.minute % 60;
            if (time.hour > 23)
            {
                time.hour = time.hour - 24;
                date.day++;
                if ((m == 1 || m == 3 || m == 5 || m == 7 || m == 8 || m == 10) &&
date.day == 32)
                {
                    m++;
                    date.day = 1;
                }
                if ((m == 4 || m == 6 || m == 9 || m == 11) && date.day == 31)
                {
                    m++;
                    date.day = 1;
                }
                if (m == 12 && date.day == 32)
                {
                    m = 1;
                    date.year++;
                    date.day = 1;
                }
                if ((date.year % 4 == 0 && date.year % 100 != 0) || date.year % 400
== 0)
                {
                    if (m == 2 && date.day == 30)
                    {
                        m++;
                        date.day = 1;
                    }
                }
                else
                {
                    if (m == 2 && date.day == 29)
                    {
                        m++;
                        date.day = 1;
                    }
                }
            }
        }
    }
}
int main()
{
    DateTimeType dttm1(1999, 12, 31, 23, 59, 59), dttm2;
    (dttm1.GetDate()).PrintDate();
    dttm1.PrintDateTime();
    dttm2.PrintDateTime();
    dttm1.IncrementSecond(30);
    dttm1.PrintDateTime();
}
```

3.3 运行截图



```
1999-12-31
1999-12-31 23:59:59
1-1-1 0:0:0
2000-1-1 0:0:29

E:\ConsoleApplication1\Debug\ConsoleApplication1.exe (进程 20792)已退出，代码为 0。
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口. . .
```

3.4 调试情况

三、实验体会

通过这次实验,我更加明白了 c++理论中的一些编程规范和 c++语言特性，掌握了基本编程知识，以后会更加认真的学习 c++理论知识，并不断实践和练习，在 debug 中不断学习。