

## 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验六(12月3日5-8节)-薛猛老师  
班级 信息安全 19-01 班 姓名 许万鹏 学号 05191643  
仪器组号 66 实验日期 12月3日

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) 掌握构造函数与析构函数的定义及使用。

### 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，不需要输出实部。
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】

1 2 3 4

1

#### 【输出】

4+6i

#### 1.2 源代码

```

#include<iostream>

using namespace std;

class Complex
{
private:

```

```
    double real;

    double imag;
public:
    Complex();

    Complex(double, double);

    Complex AddCom(Complex);

    void OutCom();
};

Complex::Complex(double r=0, double i=0)
{
    real = r;
    imag = i;
}

Complex Complex::AddCom(Complex c2)
{
    (*this).imag += c2.imag;

    (*this).real += c2.real;

    return *this;
}

void Complex::OutCom()
{
    if (real == 0)

        cout << imag << "i";
```

```

        else if (imag == 0)

            cout << real;

        else

            cout << real << "+" << imag << "i";
    }
}

int main()
{

    int a, b, c, d;

    cin >> a >> b >> c >> d;

    Complex x(a, b);

    Complex y(c, d);

    x.AddCom(y);

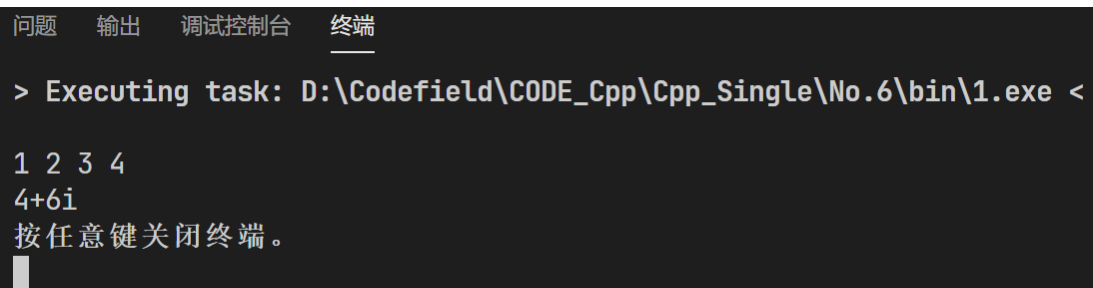
    x.OutCom();

    return 0;

}

```

### 1.3 运行截图



```

问题  输出  调试控制台  终端
> Executing task: D:\Codefield\CODE_Cpp\Cpp_Single\No.6\bin\1.exe <

1 2 3 4
4+6i
按任意键关闭终端。

```

### 1.4 调试情况

Accepted, 如上图。

## 2、第二题

### 2.1 题目描述

题目有问题，描述不清

### 2.2 源代码

无

### 2.3 运行截图

无

### 2.4 调试情况

无

## 3、第三题

### 3.1 题目描述

#### 【题目描述】

```
class DateType {  
//自定义的日期类 DateType  
  
    int y,m,d; //数据成员，表示当前日期的年、月、日  
public:  
  
    DateType(int y0=1, int m0=1, int d0=1);  
  
    //构造函数，设定年、月、日；并设置参数默认值  
  
    void IncrementDay(); //增加 1 天  
  
    bool Equal(DateType dt2); //判断二日期是否相等  
  
    void PrintDate(); //屏幕输出日期对象的有关数据（年、月、日）  
};
```

1、完成有参构造函数

2、完成 void IncrementDay() 函数，计算天数加 1 后的日期。

3、完成 Equal (DateType dt2) 函数，判断两个日期是否相等。

4、完成打印输出函数 PrintDate(), 在屏幕上输出日期对象的有关数据（年、月、日）

5、编写并完成主函数，实现输入包含六个整数，说明 DateType 类对象 dt1, dt2, 分别是 dt1 和 dt2 的年月日。请先输出 dt1, dt2, 然后判断 dt1, dt2 是否相等，再对 dt1, dt2 分别增加一天。最后输出 dt1, dt2。

注意：在 IncrementDay 成员函数中，当对日期增加 1 天后，要注意所谓的“进位”问题：首先算出本“日”所在的月份具有的天数 N（注意闰年与平年的 2 月份天数不一样），若加 1 之后的“日”数值超过所在的月份具有的天数 N 时，“进位”到月，而月份若超过 12 时还要“进位”到年等。

**【输入】**

1 1 1 1999 12 31

**【输出】**

1:1:1

1999:12:31

False

1:1:2

2000:1:1

**3.2 源代码**

```
#include <iostream>
```

```
using namespace std;
```

```
class DateType {
```

```
//自定义的日期类 DateType
```

```
    int y,m,d; //数据成员，表示当前日期的年、月、日
```

```
public:

    DateType(int y0=1, int m0=1, int d0=1);

    //构造函数，设定年、月、日；并设置参数默认值

    void IncrementDay(); //增加 1 天

    bool Equal(DateType dt2); //判断二日期是否相等

    void PrintDate(); //屏幕输出日期对象的有关数据（年、月、日）
};

DateType::DateType(int y0, int m0, int d0)
{

    y=y0;

    m=m0;

    d=d0;

}

int dpm[13]={0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

void DateType::IncrementDay()
{

    if(m==2)

    {

        if((y%4==0 && y%100!=0) || y%400==0)

        {

            if(d==29)

            {

                m++;
```

```
        d=1;

    }

    else d++;

}

else

{

    if(d==28)

    {

        m++;

        d=1;

    }

    else d++;

}

}

else

{

    if(d==dpm[m])

    {

        if(m==12)

        {

            y++;

            m=1;

            d=1;

        }

    }

}
```



```

        }

        else

        {

            m++;

            d=1;

        }

    }

    else d++;

}

}

bool DateType::Equal(DateType dt2)

{

    if(y==dt2.y && m==dt2.m && d==dt2.d) return true;

    else return false;

}

void DateType::PrintDate()

{

    cout<<y<<' :'<<m<<' :'<<d<<endl;

}

int main()

{

    int dt1_y,dt1_m,dt1_d,dt2_y,dt2_m,dt2_d;

    cin>>dt1_y>>dt1_m>>dt1_d>>dt2_y>>dt2_m>>dt2_d;

```

```

    DateType dt1(dt1_y, dt1_m, dt1_d);

    DateType dt2(dt2_y, dt2_m, dt2_d);

    dt1.PrintDate();

    dt2.PrintDate();

    if(dt1.Equal(dt2)) cout<<"True"<<endl;

    else cout<<"False"<<endl;

    dt1.IncrementDay();

    dt2.IncrementDay();

    dt1.PrintDate();

    dt2.PrintDate();

    return 0;
}

```

### 3.3 运行截图

```

> Executing task: D:\Codefield\CODE_Cpp\Cpp_Single\No.6\bin\3new.exe <
1 1 1 1999 12 31
1:1:1
1999:12:31
False
1:1:2
2000:1:1

```

### 3.4 调试情况

Accepted, 如上图。

## 4、第四题

### 4.1 题目描述

**【题目描述】**

设计一个学生类（CStudent），其私有数据成员：注册号、姓名、数学、外语、计算机课程的成绩。公有成员函数是：求三门课总成绩的函数 Sum；求三门课平均成绩的函数 Average；显示学生数据信息的函数 Display；设置学生数据信息的函数 SetData。

1. 可按如下样式设计 CStudent 类的各数据成员以及成员函数

```
class CStudent { //学生类 CStudent

unsigned long reg_num; //数据成员：注册号

char name[30]; //数据成员：姓名

float math, eng, comp; //数据成员：数学、英语、计算机成绩

public: //公有成员函数

float Sum(); //求三门课总成绩的函数 Sum

float Average(); //求三门课平均成绩的函数 Average

Display(); //显示学生数据信息的函数 Display

SetData (unsigned long r, char* n, float m, float e, float c) ;

//设置学生数据信息的函数 SetData

};
```

在主函数，通过使用“CStudent stu[150];”的语句，来说明一个 CStudent 类对象的数组 stu，而后通过各对象 stu[i]来处理并求取每一学生的总成绩、平均成绩等。

（1）输入本次欲处理的学生人数 TOTAL（小于等于 150 的正整数）；

（2）输入全班 TOTAL 个学生的有关信息，依次放入对象数组的各元素 stu[i]中（通过使用“stu[i].SetData(···);”形式的语句来实现）；

（3）对全班 TOTAL 个学生，依次通过对象 stu[i]来求出其总成绩、平均成绩等（其中要使用形如“stu[i].Sum()”以及“stu[i].Average()”式样的对成员函数进行调用的语句），并同时求出全班学生总成绩最高者处于 stu 数组的下标位置 idx\_max，而后通过使用“stu[idx\_max].Display();”来输出该学生有关的全部数据信息。

3. 程序执行后的输入输出界面样式可设计为：

TOTAL=3

CStudent 1 : 100001 ma 78 86 90 (注意空格)

CStudent 2 : 100002 li 85 91 88

CStudent 3 : 100003 hu 82 89 88

CStudent1.Sum=254,CStudent1.average=84.6667

CStudent2.Sum=264,CStudent2.average=88

CStudent3.Sum=259,CStudent3.average=86.3333

class\_Sum\_max=264

The infomation of the CStudent with class\_Sum\_max : 100002 li 85 91 88

#### 【输入】

TOTAL=3

CStudent 1 : 100001 ma 78 86 90 (注意空格)

CStudent 2 : 100002 li 85 91 88

CStudent 3 : 100003 hu 82 89 88

#### 【输出】

CStudent1.Sum=254,CStudent1.average=84.6667

CStudent2.Sum=264,CStudent2.average=88

CStudent3.Sum=259,CStudent3.average=86.3333

class\_Sum\_max=264

The infomation of the CStudent with class\_Sum\_max : 100002 li 85 91 88

#### 4.2 源代码

```
#include<iostream>

#include<string.h>

using namespace std;

class CStudent { //学生类 CStudent

    unsigned long reg_num; //数据成员：注册号

    char name[30]; //数据成员：姓名

    float math, eng, comp; //数据成员：数学、英语、计算机成绩

public: //公有成员函数

    float Sum(); //求三门课总成绩的函数 Sum

    float Average(); //求三门课平均成绩的函数 Average

    void Display(); //显示学生数据信息的函数 Display

    void SetData (unsigned long r, char* n, float m, float e,
float c) ;

    //设置学生数据信息的函数 SetData

};

void CStudent::Display()

{

    cout << reg_num << " " << name << " " << math << " " << eng << "
" << comp << endl;

}

float CStudent::Sum()

{

    return(math + eng + comp);

}
```

```
float CStudent::Average()

{

    return (*this).Sum() / 3;

}

void CStudent::SetData(unsigned long r, char* n, float m, float e,
float c)

{

    reg_num = r;

    strcpy(name, n);

    math = m;

    eng = e;

    comp = c;

}

int main()

{

    int n;

    unsigned long num;

    char name[30];

    float math, eng, comp;

    float max = 0;

    int max_stu = 0;
```

```
cin >> n;

CStudent* stu = new CStudent[n];


int j;

for (j = 0; j < n; j++)

{

    cin >> num >> name >> math >> eng >> comp;

    stu[j].SetData(num, name, math, eng, comp);

}

for (j = 0; j < n; j++)

{

    if (max < stu[j].Sum())

    {

        max_stu = j;

        max = stu[j].Sum();

    }

}

cout << "TOTAL=" << n << endl;

for (int i = 0; i < n; i++)

{

    cout << "CStudent " << i + 1 << " : ";

    stu[i].Display();

}
```

```

    }

    for (int i = 0; i < n; i++)

    {

        cout << "CStudent" << i + 1 << ".Sum=" << stu[i].Sum();

        cout << ",CStudent" << i + 1 << ".average=" <<
stu[i].Average() << endl;

    }

    cout << "class_Sum_max=" << max << endl;

    cout << "The infomation of the CStudent with class_Sum_max : ";

    stu[max_stu].Display();

    delete[] stu;

}

```

#### 4.3 运行截图

问题 输出 调试控制台 终端

---

```

> Executing task: D:\Codefield\CODE_Cpp\Cpp_Single\No.6\bin\4.exe <

3
100001 ma 78 86 90
100002 li 85 91 88
100003 hu 82 89 88
TOTAL=3
CStudent 1 : 100001 ma 78 86 90
CStudent 2 : 100002 li 85 91 88
CStudent 3 : 100003 hu 82 89 88
CStudent1.Sum=254,CStudent1.average=84.6667
CStudent2.Sum=264,CStudent2.average=88
CStudent3.Sum=259,CStudent3.average=86.3333
class_Sum_max=264
The infomation of the CStudent with class_Sum_max : 100002 li 85 91 88

按任意键关闭终端。

```

#### 4.4 调试情况



Accepted，如上图。

### 三、实验体会

通过本次实验，巩固了类与对象的相关知识，并通过新的 CUMTOJ 平台的在线排行功能体会到了自身实力的不足以及来自同学的速度压力和对程序设计的热情。