C++程序期末复习指南

详细代码

闫忠

2017\6\22 Thursday

- 写在前面的话：本文档使用的环境为 VC++6.0 与 DEV-C++，全部经过上机程序验证

  二〇一七年六月二十三日星期五早经过修改，最终版本 V1.4

一、选择结构

```
//输入成绩输出等级
#include<iostream>
using namespace std;
int main()
{
    int grade;  //默认为整型

    cout<<"请输入成绩：";

    cin>>grade;
    if(grade >= 0 && grade <= 100) //去除掉负数
    {
        switch(grade/10)
        {
        case 10:cout<<"等级为 A\n";break;
        case 9:cout<<"等级为 A\n";break;
        case 8:cout<<"等级为 B\n";break;
        case 7:cout<<"等级为 C\n";break;
        case 6:cout<<"等级为 D\n";break;
        case 5://cout<<"等级为 E\n";break;
        case 4:
        case 3:
        case 2:
        case 1:
        case 0:cout<<"等级为 E\n";break;
        default :cout<<"error\n";break;
        }
    }
    else
    {
        cout << "input error!" << endl;
    }
}
```

```
//购物金额<200 元,不打折
//500 元>购物金额>=200 元,9 折
//1000 元>购物金额>=500 元,8 折
//购物金额>1000 元,7.5 折
#include <iostream>
using namespace std;
void main()
{
    int i=500,j=1000,r=200;//if 语句条件判断需用字母
    double a,b;//顾客购物金额为 a,实际支付金额为 b
    cout<<"请输入购物金额，按回车键确认(Enter):";
    cin>>a;
    if(a>j)
    {
        b=a*0.75;
        cout<<"顾客购物打 7.5 折，实际支付金额为："<<b<<"元"<<endl;
     }
     else if(i<=a&&a<=j)
     {
        b=a*0.80;
        cout<<"顾客购物打 8 折，实际支付金额为："<<b<<"元"<<endl;
     }
     else if(r<=a&&a<i)
    {
        b=a*0.90;
        cout<<"顾客购物打 9 折，实际支付金额为："<<b<<"元"<<endl;
    }
    else
    {
        b=a;
        cout<<"顾客购物不打折，实际支付金额为："<<b<<"元"<<endl;
    }
}
```

```
//输入数字输出星期几
#include<iostream>
#include<string>
using namespace std;
int main()
{
    int x;
    cout << "请输入数字：";
    cin >> x;
    switch(x)
    {
        case 1: cout<<"星期一"<<endl; break;
        case 2: cout<<"星期二"<<endl; break;
        case 3: cout<<"星期三"<<endl; break;
        case 4: cout<<"星期四"<<endl; break;
        case 5: cout<<"星期五"<<endl; break;
        case 6: cout<<"星期六"<<endl; break;
        case 7: cout<<"星期天"<<endl; break;
    }
    return 0;
}
//如果不加 default; 程序会有安全的问题。
```

```
//输入数字输出星期几
#include<iostream>
#include<string>
using namespace std;
void main()  //使用 void 作为返回类型的话就不需要后面添加 return 0
{
    int x;
    cout << "请输入数字：";
    cin >> x;
    switch(x)
    {
        case 1: cout<<"星期一"<<endl; break;
        case 2: cout<<"星期二"<<endl; break;
        case 3: cout<<"星期三"<<endl; break;
        case 4: cout<<"星期四"<<endl; break;
        case 5: cout<<"星期五"<<endl; break;
        case 6: cout<<"星期六"<<endl; break;
        case 7: cout<<"星期天"<<endl; break;
        default: cout<<"error!"<<endl; break;
    }
}
```

二、循环结构

```
//求销售总额
#include <iostream>
using namespace std;
void main()
{
        double sum = 0, i;
        int n = 1; //n 作为次数变量不应该是 double 类型。
        cout << "请输入第"<<n<<"次购物总额：";
        cin >> i;
        while(i >0) //这里结束的条件是输入一个负数
        {
                sum += i;
                n++;
                cout << "请输入第"<<n<<"次购物总额：";
                cin >> i;
        }
        cout << "销售总额为：" << sum << endl;
}
```

三、简单的类与对象

```
//教师类
#include<iostream>
#include<string>
using namespace std;
class Teacher
{
private:
    int num;
    string name;
    string sex;
    string title;
public:
    Teacher(){} //空构造函数
    Teacher(int nu,string na,string se,string
ti):num(nu),name(na),sex(se),title(ti){}//参数初始化列表（书中有）
    void set_data()
    {
        cout<<"请输入职工号："; cin>>num;
        cout<<"请输入姓名：";  cin>>name;
        cout<<"请输入性别：";  cin>>sex;
        cout<<"请输入岗位：";  cin>>title;
    }
    void display()
    {
        cout<<"职工号："<<num<<" "<<"姓名："<<name<<" "<<"性别："<<sex<<" "<<"岗位：
"<<title<<endl;
    }
};
void main()
{
    Teacher t1,t2(007,"沃德天","男","教师");
    t1.set_data();
    t1.display();
    t2.display(); //输出默认构造函数的值
}
```

四、继承与派生

基类中的成员	在公用派生类中的访问属性	在私有派生类中的访问属性	在保护派生类中的访问属性
私有成员	不可访问	不可访问	不可访问
公用成员	公用	私有	保护
保护成员	保护	私有	保护

```
//继承与派生，日期到时间
#include<iostream>
using namespace std;
class Date
{
 public:
        Date(){}
        Date(int y,int mo,int d):year(y),month(mo),day(d){}
        void show(){cout<<year<<"年"<<month<<"月"<<day<<"日"<<endl;}
protected:  //不继承的时候，保护和私有属性是一样的
        int year;
        int month;
        int day;
};
class Time:public Date
{
 public:
        Time(){}//参数初始化列表使用继承的时候的格式
        Time(int y,int mo,int d,int h,int m,int s):Date(y,mo,d),hour(h), minute(m),
sec(s){}
        void displayall(){show();cout<<hour<<":"<<minute<<":"<<sec<<endl;}
protected:
        int hour;
        int minute;
        int sec;
};
void main()
{
         Date d(2009,6,8);
         d.show();
         Time t(2009,6,8,19,47,58);
         t.displayall();
}
```

五、运算符重载

```
//友元函数的声明放在类中，VC 不支持语法，使用.h 头文件
#include <iostream.h>
//using namespace std;
class complex
{
public:
    complex(double real = 0.0, double imag = 0.0): m_real(real), m_imag(imag){ };
public:                                          //构造函数后面有分号结束，同样可以运行
    friend complex operator+(const complex & A, const complex & B);
    friend complex operator-(const complex & A, const complex & B);
    friend complex operator*(const complex & A, const complex & B);
    friend complex operator/(const complex & A, const complex & B);
    friend istream & operator>>(istream & in, complex & A);
    friend ostream & operator<<(ostream & out, complex & A);
private:
    double m_real;  //实部
    double m_imag;  //虚部
};

//重载加法运算符
complex operator+(const complex & A, const complex &B)
{
    complex C;
    C.m_real = A.m_real + B.m_real;
    C.m_imag = A.m_imag + B.m_imag;
    return C;
}

//重载减法运算符
complex operator-(const complex & A, const complex &B)
{
    complex C;
    C.m_real = A.m_real - B.m_real;
    C.m_imag = A.m_imag - B.m_imag;
    return C;
}

//重载乘法运算符
complex operator*(const complex & A, const complex &B)
{
    complex C;
    C.m_real = A.m_real * B.m_real - A.m_imag * B.m_imag;
```

```
    C.m_imag = A.m_imag * B.m_real + A.m_real * B.m_imag;
    return C;
}
//重载除法运算符
complex operator/(const complex & A, const complex & B)
{
    complex C;
    double square = B.m_real * B.m_real + B.m_imag * B.m_imag;
    C.m_real = (A.m_real * B.m_real + A.m_imag * B.m_imag)/square;
    C.m_imag = (A.m_imag * B.m_real - A.m_real * B.m_imag)/square;
    return C;
}
//重载输入运算符
istream & operator>>(istream & in, complex & A)
{
    in >> A.m_real >> A.m_imag;
    return in;
}
//重载输出运算符
ostream & operator<<(ostream & out, complex & A)
{
    out << A.m_real <<" + "<< A.m_imag <<" i ";
    return out;
}

int main()
{
    complex c1, c2, c3;
    cin>>c1>>c2;
    c3 = c1 + c2;
    cout<<"c1 + c2 = "<<c3<<endl;
    c3 = c1 - c2;
    cout<<"c1 - c2 = "<<c3<<endl;
    c3 = c1 * c2;
    cout<<"c1 * c2 = "<<c3<<endl;
    c3 = c1 / c2;
    cout<<"c1 / c2 = "<<c3<<endl;

    return 0;
}
```

```
1  #include <iostream>
2  using namespace std;
3  class complex
4  {
5  public:
6      complex(double real = 0.0, double imag = 0.0): m_real(real), m_imag(imag){ }
7  public:
8      friend complex operator+(const complex & A, const complex & B);
9      friend complex operator-(const complex & A, const complex & B);
10     friend complex operator*(const complex & A, const complex & B);
11     friend complex operator/(const complex & A, const complex & B);
12     friend istream & operator>>(istream & in, complex & A);
13     friend ostream & operator<<(ostream & out, complex & A);
14 private:
15     double m_real;  //实部
16     double m_imag;  //虚部
```

```
编译结果...
--------
- 错误: 0
- 警告: 0
- 输出文件名: C:\Users\Administrator\Desktop\运算符输出.exe
- 输出大小: 1.83400630950928 MiB
- 编译时间: 1.03s
```

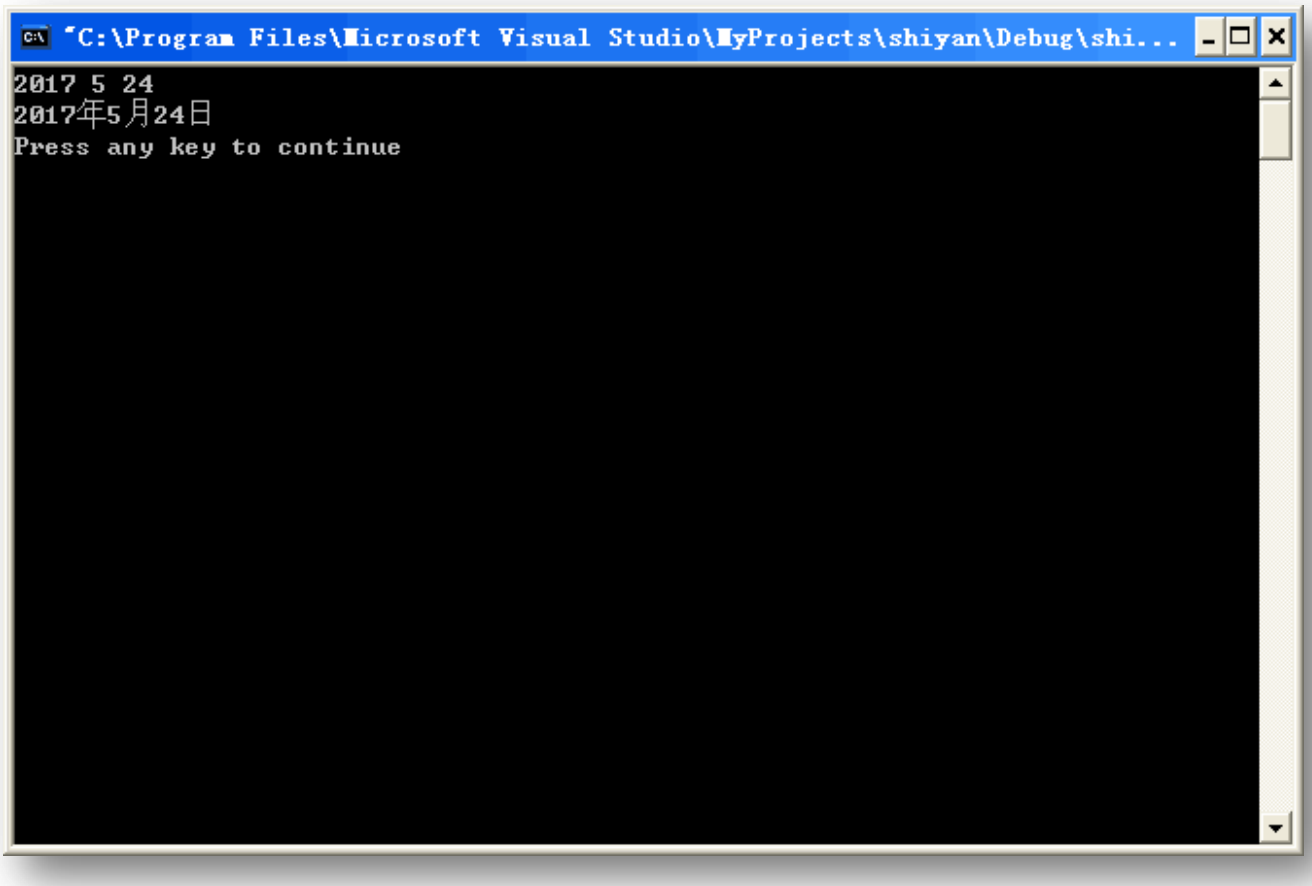
```
1  #include <iostream>
2  using namespace std;
3  class complex
4  {
5  public:
6      complex(double real = 0.0, double imag = 0.0): m_real(real), m_imag(imag){ };
7  public:
8      friend complex operator+(const complex & A, const complex & B);
9      friend complex operator-(const complex & A, const complex & B);
10     friend complex operator*(const complex & A, const complex & B);
11     friend complex operator/(const complex & A, const complex & B);
12     friend istream & operator>>(istream & in, complex & A);
13     friend ostream & operator<<(ostream & out, complex & A);
14 private:
15     double m_real;  //实部
16     double m_imag;  //虚部
```

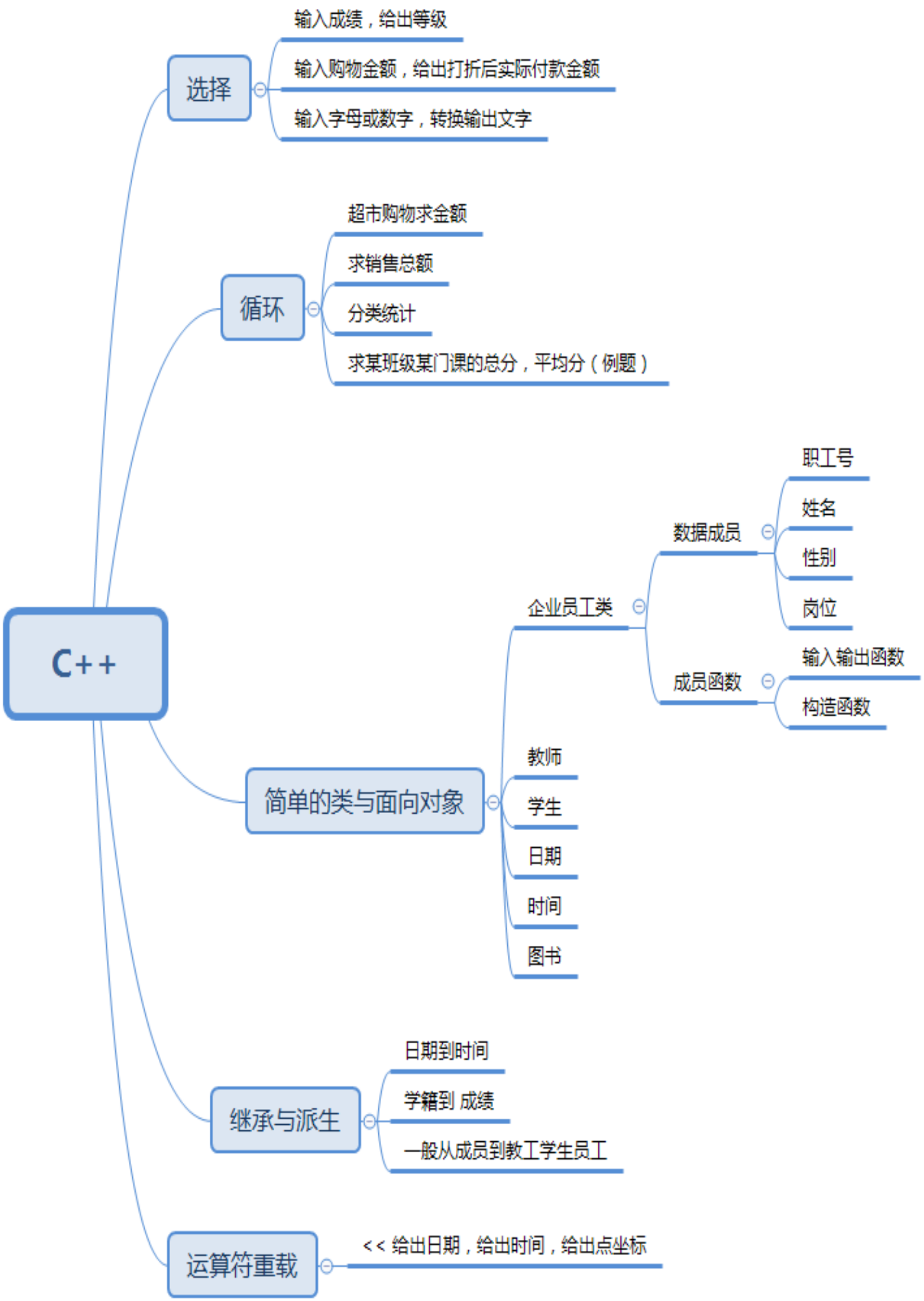
```
编译结果...
-------
错误: 0
警告: 0
输出文件名: C:\Users\Administrator\Desktop\运算符输出.exe
输出大小: 1.83400630950928 MiB
编译时间: 0.94s
```

```
//DEV-C++ 版本，如果使用 VC，请改头文件和去掉 using namespace std;
//输出 3 维点坐标，在这个例子中为什么不需要修改头文件呢？因为友元函数的声明和定义都在类中
//在 VC6 中经过了验证，保险起见，建议使用修改头文件的方法。
#include <iostream>
using namespace std;

class P3
{
private:
        double x, y, z;
public:
        P3(){}
        P3(double a, double b, double c):x(a), y(b), z(c){}

        void set_data();
        void display(P3 &);  //对象引用
        friend ostream & operator<<(ostream & output, P3 & c)
        {
                output << c.x << " " << c.y << " " << c.z << endl;
                return output;
        }
};
void P3::set_data()
{
        cin >> x >> y >> z;
}
//这里是强行添加 显示函数
void P3::display(P3 &c)
{
        cout << c;
}

int main()
{
        P3 c1(1, 2, 3), c2;
        c2.set_data();
        //c2.display(c2);
        c1.display(c1); //默认输出 1 2 3
        cout << c2;  //不添加显示函数，直接输出对象也行
    return 0;
}
```

```
6 6 6
1 2 3
6 6 6

--------------------------------
Process exited after 4.5 seconds with return value 0
请按任意键继续. . .
```

```
//课题：运算符重载，输出日期
//在这个例子中为什么不需要修改头文件呢？因为友元函数的声明和定义都在类中
//在 VC6 中经过了验证。
#include<iostream>
using namespace std;

class Date
{
//public:
private:
    int year, month, day;
public:
    Date(){}
    Date(int y, int m, int d)
    {
            year = y;
            month = m;
            day = d;
    }
    void set_data()
    {
        cin >> year >> month >> day;
    }

    //重载<<，使之能输出：xxxx 年 x 月 x 日
//    friend ostream& operator<<(ostream&, Date&)
    friend ostream & operator<<(ostream& output, Date& d)
    {
        output << d.year << "年" << d.month << "月" << d.day << "日" << endl;
        return output;
    }

};

void main()
{
    Date d1(2000, 1, 1), d2;
    d2.set_data();
    cout << d2 ;
}
```


"C:\Program Files\Microsoft Visual Studio\MyProjects\shiyan\Debug\shi...
2017 5 24
2017年5月24日
Press any key to continue


C++
选择
输入成绩，给出等级
输入购物金额，给出打折后实际付款金额
输入字母或数字，转换输出文字
循环
超市购物求金额
求销售总额
分类统计
求某班级某门课的总分，平均分（例题）
简单的类与面向对象
企业员工类
数据成员
职工号
姓名
性别
岗位
成员函数
输入输出函数
构造函数
教师
学生
日期
时间
图书
继承与派生
日期到时间
学籍到 成绩
一般从成员到教工学生员工
运算符重载
<< 给出日期，给出时间，给出点坐标