请将所有编程题目答案保存到同一个word文件内，以“选课序号-姓名.doc”命名，上传到答案处。答案应包含代码和运行截图。

可根据题意需要自行设计其他成员函数。

（1）定义一个时间类Time,它能提供由时分秒组成的时间。要求能够设置时间和打印出时间。

数据成员：时、分、秒

成员函数：

1. 设置时间set函数，可以设置时间的时、分、秒，可以分别设置也可以整体设置时间; 可以用函数重载和函数默认参数实现。

2. tick函数，负责给当前时间加1秒钟，要保证测试下列情况：

a)递增到下一分钟。  
    b)递增到下一小时。  
    c)递增到下一天(即11：59：59PM到00：00：00AM)

                    3. 输出时间的函数show

class Time

{

private:

int hour = 0;

int minute = 0;

int second = 0;

public:

void Set(int hour = 0, int minute = 0, int second = 0); // 默认参数设置时间

void Set(int, int); // 重载函数设置时间，和默认参数二选一即可

void Set(int); // 重载函数设置时间，和默认参数二选一即可

void Tick(); // +1s

void Show(); // 输出时间，采用24小时制，参考H:M:S

};

int main()

{

Time time;

time.Set(13, 1, 59); // 采用24小时制

time.Show(); // 输出13:1:59

time.Tick();

time.Show(); // 输出13:2:0

time.Set(13, 59, 59);

time.Tick();

time.Show(); // 输出14:0:0

time.Set(23, 59, 59);

time.Tick();

time.Show(); // 输出0:0:0

return 0;

}

（2）编写一个‘User’类，其有2个数据成员：用户id号和密码（密码必须是6位数字）； 编写构造函数，可根据id号构造一个新用户，其密码初始化为‘000000’； 编写一个输出id号的函数；写一个修改密码的函数，需要旧密码和新密码。 只有输入的旧密码正确，此功能才能修改密码并提示操作是否成功。在主函数中测试该User类。

class User

{

private:

unsigned int id = 0; // id 非负

char password[6]; // 6为密码，默认000000

public:

User(unsigned int id); // 根据id构造User对象

void PrinId(); // 输出id

void SetPassword(char \*oldPassword, char \*newPassword); // 设置密码，比较新旧密码，提示操作是否成功

};

int main()

{

User user(123456);

user.PrinId(); // 输出123456

user.SetPassword("000000", "11111A"); // 输出密码设置失败

user.SetPassword("000000", "1111111"); // 输出密码设置失败

user.SetPassword("0", "111111"); // 输出密码设置失败

user.SetPassword("100000", "111111"); // 输出密码设置失败

user.SetPassword("000000", "111111"); // 输出密码设置成功

return 0;

}

(3)定义一个分数类如下，要求实现各个成员函数，并在主函数中测试两个分数的加减乘除等运算，可以根据需要设计其他成员函数。

class Rational{

  public:

    Rational(int nn=1,int mm=1); //构造函数

     Rational  add(const Rational & A);        //加

    Rational  sub(const Rational & A);        //减

    Rational  mul(const Rational & A);        //乘

    Rational  div(const Rational & A);        //除

   void print();             //以分数形式显示，注意约分

  private:

void simple( );        //约分

      int m;   // 分母

      int n;   // 分子

};

int main()

{

Rational a(1, 2), b(2, 3);

Rational result = a.add(b);

result.print(); // 7/6

result = a.sub(b);

result.print(); // -1/6

result = a.mul(b);

result.print(); // 1/3

result = a.div(b);

result.print(); // 3/4

Rational c(0, 1);

result = a.mul(c); // 乘0

result.print();

result = a.div(c); // 不能除以0

Rational test\_zero(1, 0); // 分母不能为0

return 0;

}

**（4）设计日期类 Date**

•          类包括私有数据成员 year, month, day;

•          公有成员函数实现以下功能：

1）定义有参数构造与默认构造函数。注意：应该保证每一个新构造出来的日期对象是有效的日期，如果实参表示的日期无效，可以将其设置为一个默认的有效时间

2）设置时间bool set( int y, int m, int d)，需保证设置的日期有效，如果实参表示的日期无效，可以将其设置为一个默认的有效时间。

3）输出时间，按月/ 日 / 年的方式输出日期

4）判断该日期是一年的第几天

5）计算两个日期的差（假设都是2000年以后的日期）  
编写一个驱动程序，对该类进行测试

#include <iostream>

using namespace std;

class Date

{

public:

Date(int y, int m, int d);

bool set(int y, int m, int d);

void print();

int day\_of\_year();

int diff(const Date &A);

private:

bool is\_valid(int y, int m, int d);

int year;

int month;

int day;

};

int main()

{

Date invalid\_date(2020, 2, 30);

invalid\_date.print(); // 输出默认日期

Date default\_date;

default\_date.print(); // 输出默认日期

Date a(2024, 3, 11);

a.print(); // 输出 3/11/2024

a.set(2019, 2, 29);

a.print(); // 输出默认日期

a.set(2024, 2, 29);

a.print(); // 输出 2/29/2024

a.set(2024, 3, 11);

cout << a.day\_of\_year() << endl; // 输出 71

Date b(2024, 5, 1);

cout << a.diff(b) << endl; // 输出 -51

b.set(2023, 12, 25);

cout << a.diff(b) << endl; // 输出 77

cout << a.diff(a) << endl; // 输出 0

}

(5) 完成 Array类

class Array

{

public:

Array(int n); // n个元素. 所有元素初始为 0

Array(int\*p, int n); //用一个整型原生C数组初始化

Array (const Array &other); //拷贝构造函数

~Array(); //析构函数

void print(); //输出所有元素

void input(); //键盘输入所有元素

void set(int index, int value); //将index号元素的值设置为value

private:

int \* data;

int size; // number of elements

};

int main()

{

Array arr1(4);

arr1.input(); // 输入1 2 3 4

arr1.print(); // 输出1 2 3 4

arr1.set(5,6); // 输出数组越界！

arr1.print(); // 输出1 2 3 4

arr1.set(2,6);

arr1.print(); // 输出1 2 6 4

Array arr2 = arr1; // 调用拷贝构造函数对arr2初始化

arr1.print(); // 输出1 2 6 4

int arr[6] = {1,2,3,4,5,6};

Array arr3(arr,6);

arr3.print(); // 输出1 2 3 4 5 6

Array arr4(arr+2,4);

arr4.print(); // 输出3,4,5,6

return 0;

}