一共 4 道题+2 选做题；

上机课现场任意检查 2 道，另外在超星提交所有 4 道题，选做题不做要求。

可以修改类的定义，使类的设计更加合理，没有错误。

(1)**定义如下字符串类**, 实现字符串的存储和操作。

```
class myString
{
   public :
        myString (const char *pn = NULL) ;
        ~ myString ( ) ;
        void set(const char *pn) ;
        void print();
        int getLen();
        void toUpper( );     //转化为大写字符串
        void toLower( );     //转化为小写字符串
    private:
         char *pStr ;       //  指向存储字符串的空间
         int size ;          //包含字符的数目
} ;
int main()
{    //测试类
    myString s0, s1("hello");
     s0.print();     //输出“空字符串”
     s0.set("world");
     s0.print();    //输出“world”
     s1.toUpper();
     cout<<s1.getLen()<<endl;   //输出 5
     myString s2 = s1;
     s2.print();    //输出“HELLO”
     s2.toLower();
     s1.print();    //输出?
     return 0;
}
```

(2) 利用上次上机作业中的日期类 Date, 设计学生类 Student，私有数据成员包括学号和姓名 int id; char name[20]; 和表示入学日期内嵌时间对象 Date roll;以及统计学生对象数目的静态成员 number。公有成员函数实现以下功能：

① 定义构造函数，实现对成员的初始化，默认学生对象的名字是“ssdut”，默认入学日期为 2023 年 8 月 28 日，学号为当前学生数目 number 的值如 2023001，每创建一个对象则 number 加 1。若有 Date d(2022,9, 1), 则可定义学生数组

Stu s[4]={ Student (”S1”, 1994,8,3), Student (”S2”, d), Student(“S3”)};

三个学生的学号自动设置为分别为 2023001, 2023002 和 2023003；

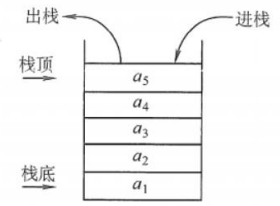
② 显示学生相关信息的 show(), 按清晰格式输出学生的所有信息

③ 获取 number 的静态成员函数 geNum()

请给出完整的类定义，并在 main()中对类的功能进行测试，注意测试用例的设计。

(3)设计存放字符数据的堆栈类，实现栈的基本操作，并设计主函数对该类的功能进行测试，注意对特殊情况/边界条件的测试。

栈是一种由若干个线性次序排列的元素构成的复合数据类型。对栈中数据进行操作的重要原则是先出后入（LIFO, Last In First Out）。




```
class Stack
{
public:
   Stack( int n);   // 初始化, 栈的容量为 n
   ~Stack() ;   // 析构函数
   bool push( char   x );   // 将 x 入栈存在栈顶位置，返回知否入栈成功（如果栈已满则无法入栈）
   char pop();   //将栈顶元素出栈，返回栈顶元素，并修改栈顶位置
   bool empty();   //判断栈是否为空
   bool full();   //判断栈是否已满
private:
    char* buffer;
    int top;         //栈顶位置
    int size;        // 栈的容量(最多能存放的元素个数)
};
```

(4)定义并实现一个 Point 类，一个“PointSet”类（点的集合）。 PointSet 类可以将不超过 5 个点放入集合中或将它们从集合中取出， 也可以对所有点成员进行统一处理（将所有点同时设置）。 集合超过 5 个元素后，不能再添加元素。 按照如下设计完成类实现

```
class PointSet{
private :
    int num ;  // 点的实际数量
    Point point[5] ;  // Array of points
public :
    PointSet(int n= 0 ) ;  //初始没有点
    bool add(Point & p) ; // 加入一个点.
                          //如果超过 5 个点则加入失败.
    Point get(int n) ; // 获得第 n 号点, 点的排序从 0 号开始
                       //检查 n 的合理性. 如果该点不存在，则显示提示信息.
    void setAll(double x, double y) ;// 将所有的点设置为同一个坐标 x, y
    int getNum() ;// 返回集合中点的个数
    bool resetPoint(int n, double x, double y) ; // 将第 n 号点的坐标设置为 x，y
                                                 // 如果该点不存在 return false
    void show() ;// 输出所有的点
};
```

实现这两个类并编写函数进行测试.

（附加题，不计分）

1. **此题想想就行，看是否合理**（**类结构设计**）商店经销一种货物，货物成箱购进，成箱卖出，购进和卖出时以箱（序号）为单位，各箱的重量不一样，单价不一样，因此商店需要记录下目前库存的货物的总重量和总价值。编写一个程序，通过定义类 Cargo 来模拟商店货物购进和卖出的情况。

   （本题目主要练习静态数据成员的使用，定义私有变量存每件货物的价格和重量，用静态数据成员存货物的总重量和总价钱，定义构造函数和析构函数，当定义新的对象完成初始化的功能和删除对象时，从总重量和总价钱中减去对象的重量和价格）

no	weight	price	next

   序号　　重量　　单价

```
Class Node{
    public:
        int weight;     int price;      Node *next;
};
```

   *注：重量单价 建议用链表（带头节点的单链表） ；链表完成不了用数组。

   （1）链表方式用 Node 做结点

   （2）数组方式 Node a[100]; //Node 中去掉 next

2. **（链表思维）（彻底理解单链表：1. 链表操作期望在头；2. 带头节点的单链表无需考虑插入、删除结点是否在头和尾的情形）**

   设计一个整数链表类，满足栈操作。即，总在链表首插入结点，总在链表首取出（删除）结点。类中需有记录结点个数的数据成员。如果链表为空，而要做取出结点操作，则类必须给出错误信息。

   编写应用程序，生成 100 次随机数（范围 10－200），放入链表中。然后逐个由链表中取出，求其和。

   求和工作不要在链表类中完成，以使该链表类具有通用性；提示：定义一个 sum 函数。

```
class Node{
    public:
      int data;
      Node *next;
};
```

```
class Stack{
   Node *head;
   public:
   void push( int d); //入栈
   int pop(); //弹栈,返回弹出元素的值。
   int empty(); //判断栈是否为空
};
int sum(Stack &s) //计算完，参数栈就清空了；思考一下，若不想改变参数栈应该怎么设计。
```