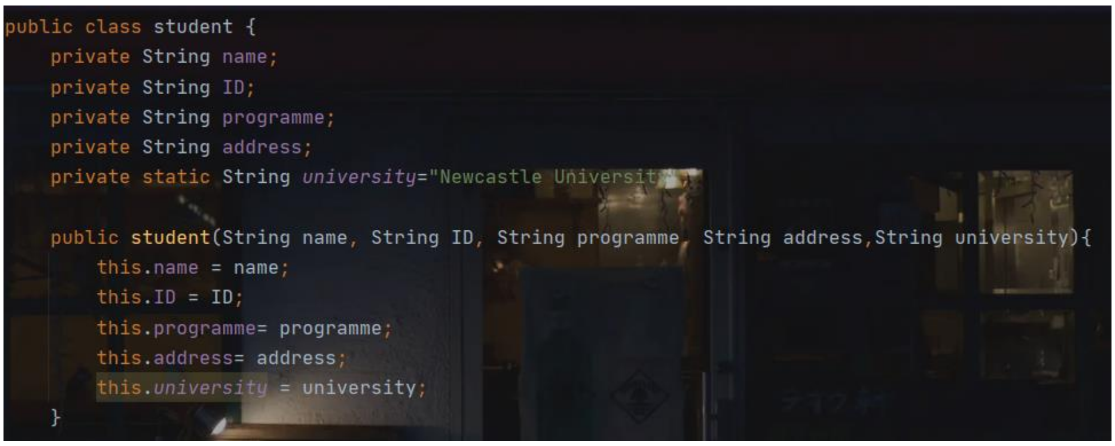
CSC8014 Coursework Report1
Ruipeng Jiao 200952811

Q1: A student class should be implemented with the following details:

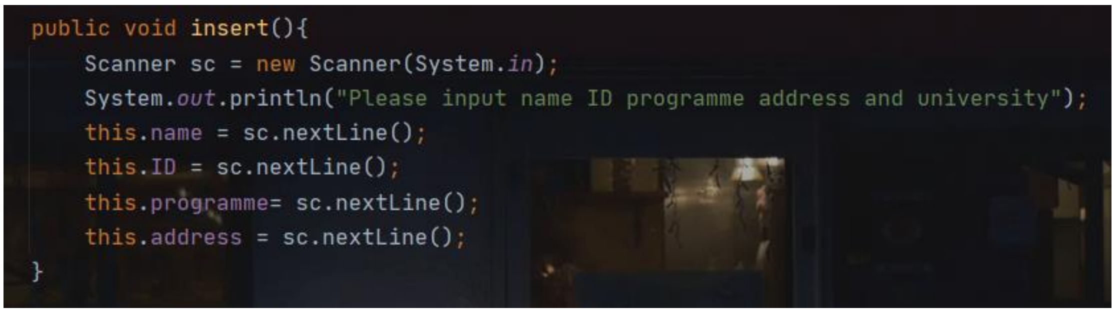
Each object involves a set of fields/variables: (i) name, (ii) ID, (iii) programme of study, (iv) address, and (v) university name;

(Some values are defined. Considering that the ID may contain letters, it is set to string type.)

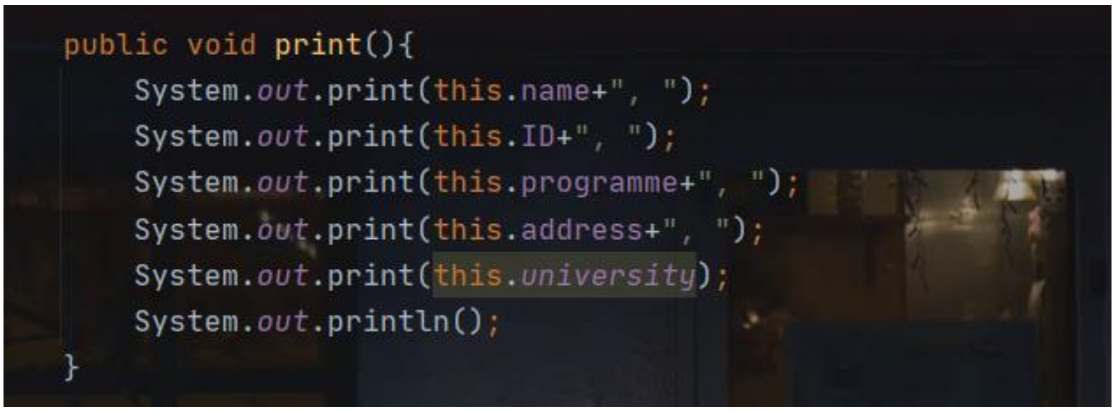


```
public class student {
    private String name;
    private String ID;
    private String programme;
    private String address;
    private static String university="Newcastle University";

    public student(String name, String ID, String programme, String address,String university){
        this.name = name;
        this.ID = ID;
        this.programme= programme;
        this.address= address;
        this.university = university;
    }
```

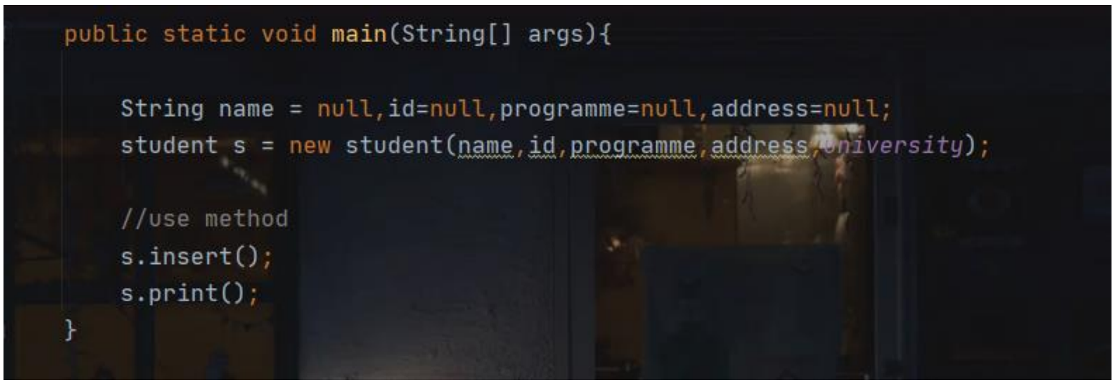
Each object can execute two functions: (i) *insert()* that sets the objects' values to the fields, and (ii) *print()* which prints the fields with their values.



```
public void insert(){
    Scanner sc = new Scanner(System.in);
    System.out.println("Please input name ID programme address and university");
    this.name = sc.nextLine();
    this.ID = sc.nextLine();
    this.programme= sc.nextLine();
    this.address = sc.nextLine();
}
```

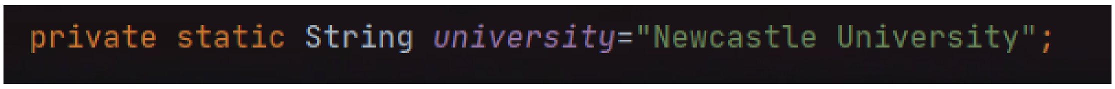


```java
public void print(){
    System.out.print(this.name+", ");
    System.out.print(this.ID+", ");
    System.out.print(this.programme+", ");
    System.out.print(this.address+", ");
    System.out.print(this.university);
    System.out.println();
}
```

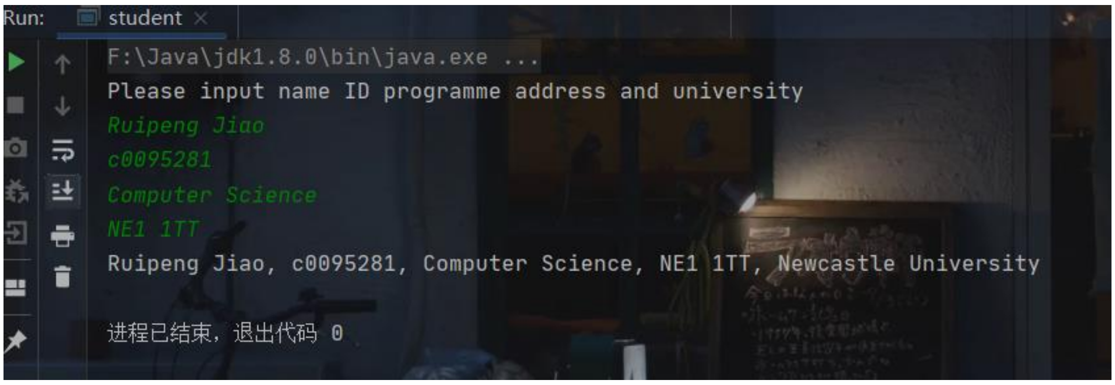


```java
public static void main(String[] args){

    String name = null,id=null,programme=null,address=null;
    student s = new student(name,id,programme,address,university);

    //use method
    s.insert();
    s.print();
}
```

The assumption is that all the objects have the same university name's value.



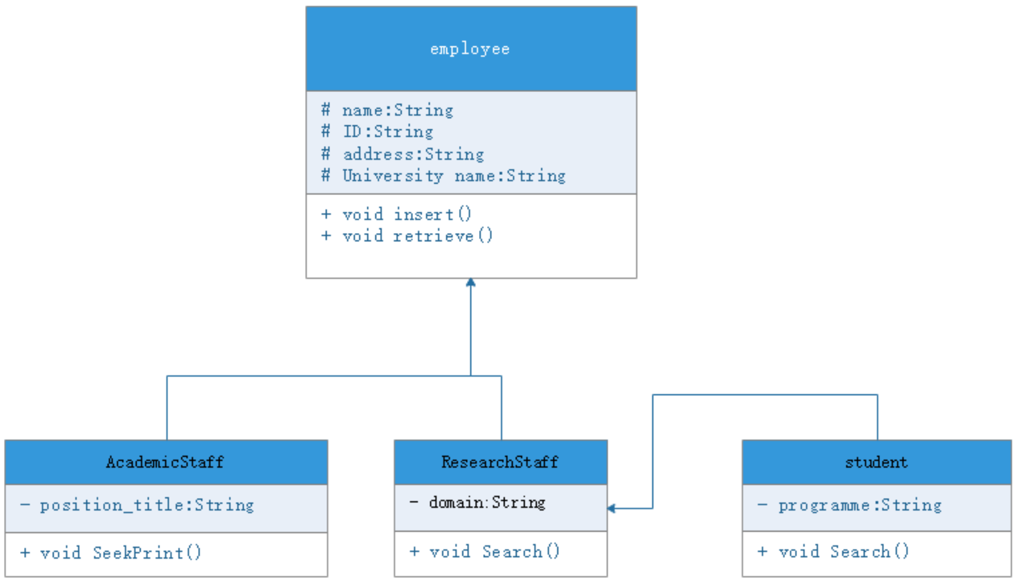
```java
private static String university="Newcastle University";
```



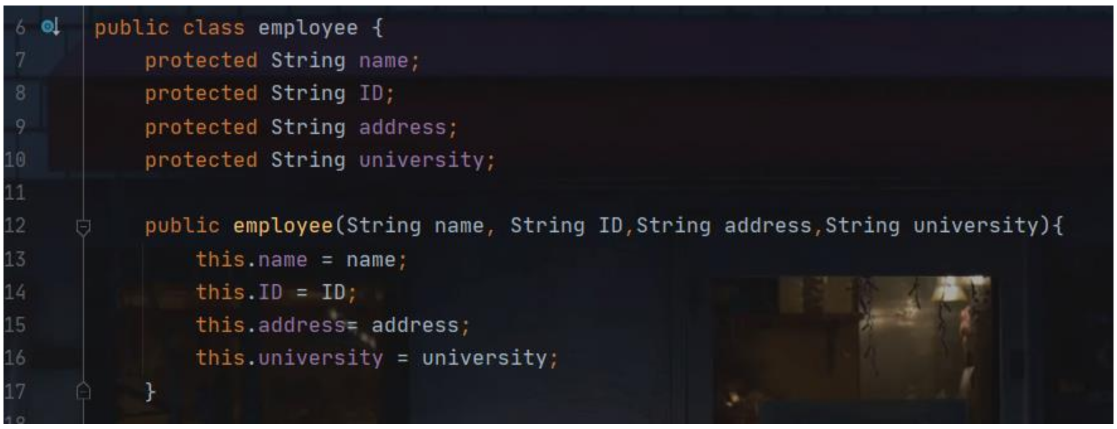
```
F:\Java\jdk1.8.0\bin\java.exe ...
Please input name ID programme address and university
Ruipeng Jiao
c0095281
Computer Science
NE1 1TT
Ruipeng Jiao, c0095281, Computer Science, NE1 1TT, Newcastle University

进程已结束，退出代码 0
```

Q2: A class is implemented for the employees working at a university. The class has two sub classes: *academic staff* and *research staff*. It is also supposed that the *student class* in Q1 is a sub class of the *research staff*. Given these assumptions, implement the classes in accordance with the following information:

UML:

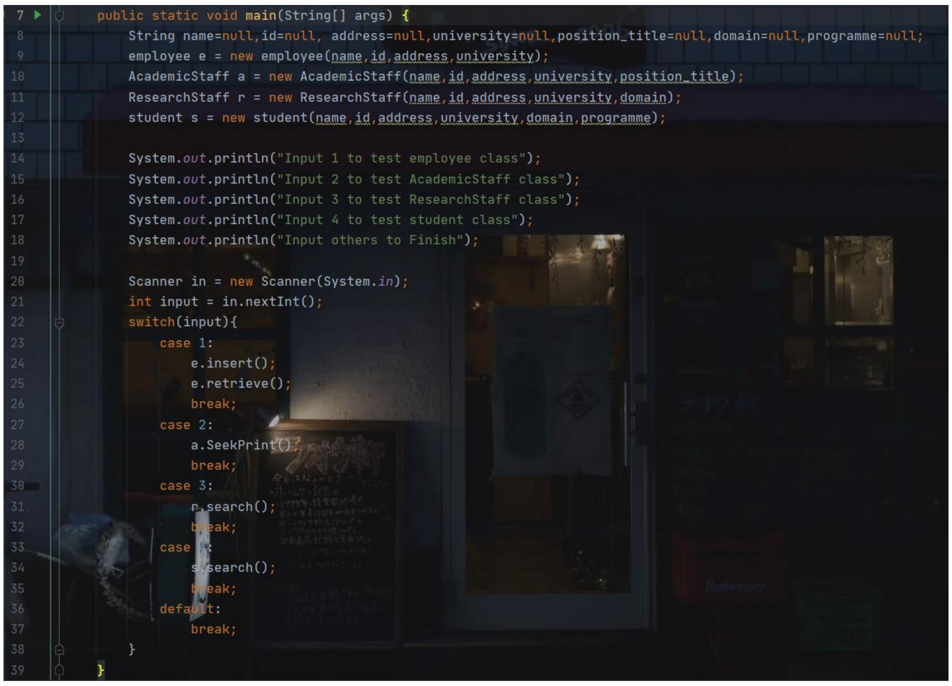


1. Employee class includes: Name, ID, address and University name fields. It has a function for inserting the fields' values and a function for retrieving them.



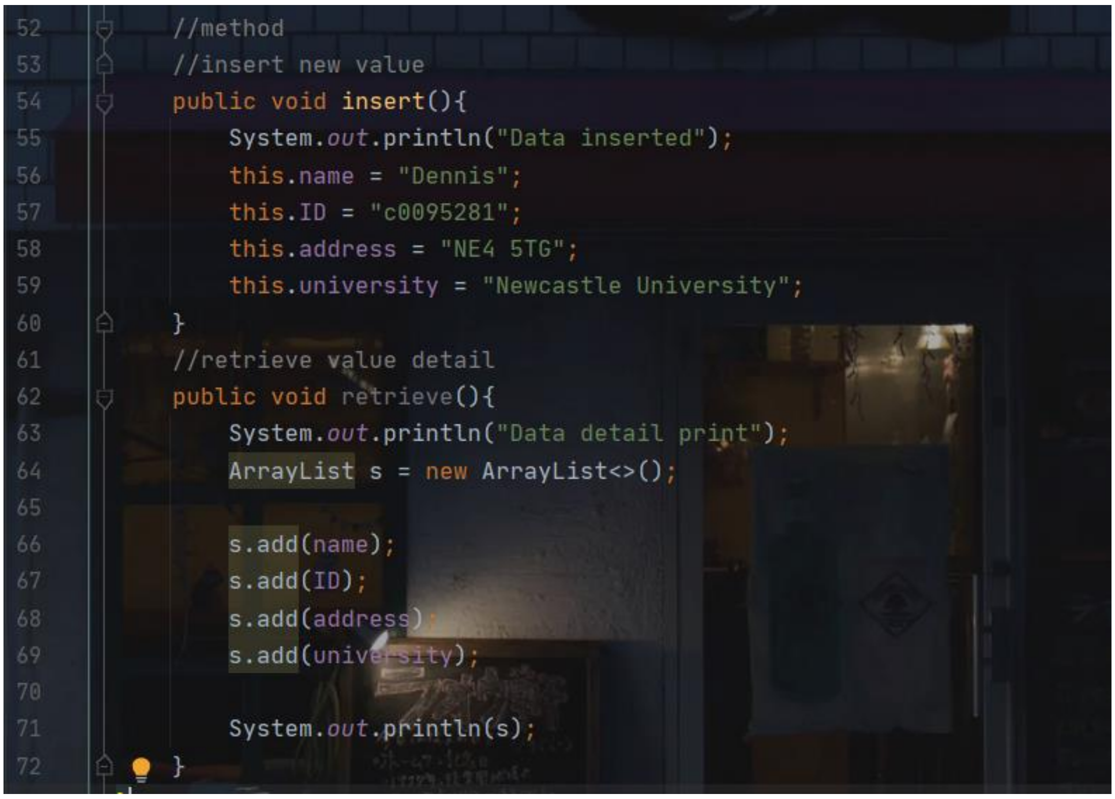
```java
public class employee {
    protected String name;
    protected String ID;
    protected String address;
    protected String university;

    public employee(String name, String ID,String address,String university){
        this.name = name;
        this.ID = ID;
        this.address= address;
        this.university = university;
    }
```

Main class:



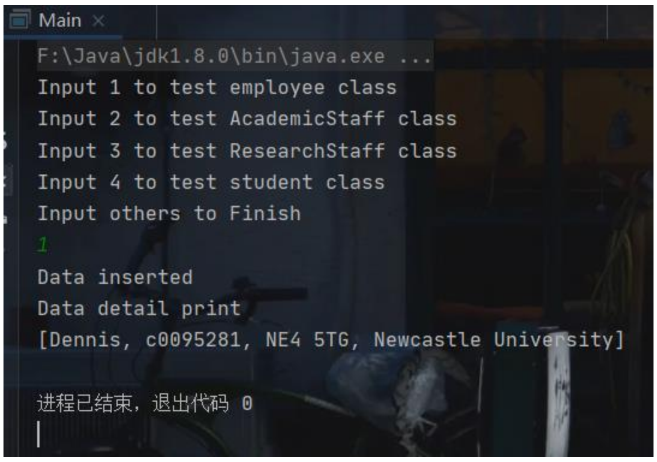
```java
    public static void main(String[] args) {
        String name=null,id=null, address=null,university=null,position_title=null,domain=null,programme=null;
        employee e = new employee(name,id,address,university);
        AcademicStaff a = new AcademicStaff(name,id,address,university,position_title);
        ResearchStaff r = new ResearchStaff(name,id,address,university,domain);
        student s = new student(name,id,address,university,domain,programme);

        System.out.println("Input 1 to test employee class");
        System.out.println("Input 2 to test AcademicStaff class");
        System.out.println("Input 3 to test ResearchStaff class");
        System.out.println("Input 4 to test student class");
        System.out.println("Input others to Finish");

        Scanner in = new Scanner(System.in);
        int input = in.nextInt();
        switch(input){
            case 1:
                e.insert();
                e.retrieve();
                break;
            case 2:
                a.SeekPrint();
                break;
            case 3:
                r.search();
                break;
            case 4:
                s.search();
                break;
            default:
                break;
        }
    }
```

(Store the inserted value in ArrayList and print it)



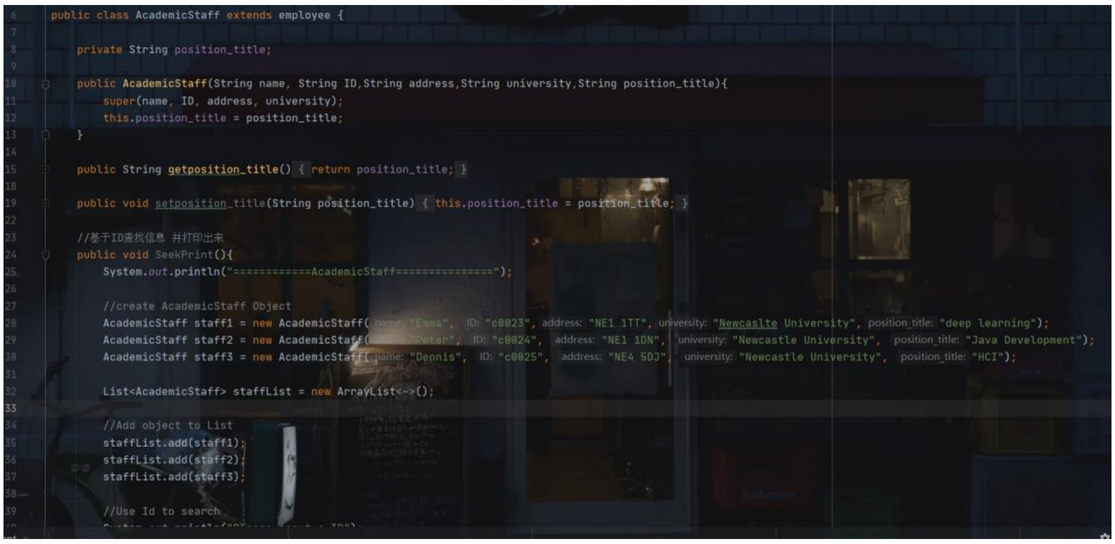
```java
    //method
    //insert new value
    public void insert(){
        System.out.println("Data inserted");
        this.name = "Dennis";
        this.ID = "c0095281";
        this.address = "NE4 5TG";
        this.university = "Newcastle University";
    }
    //retrieve value detail
    public void retrieve(){
        System.out.println("Data detail print");
        ArrayList s = new ArrayList<>();

        s.add(name);
        s.add(ID);
        s.add(address);
        s.add(university);

        System.out.println(s);
    }
```

Running result:

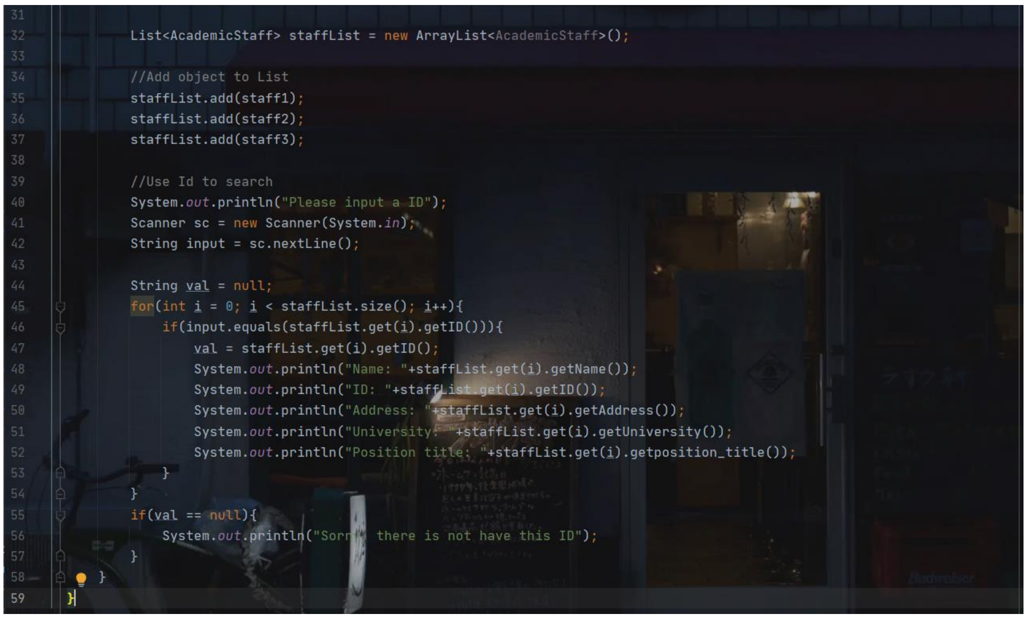


```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to test employee class
Input 2 to test AcademicStaff class
Input 3 to test ResearchStaff class
Input 4 to test student class
Input others to Finish
1
Data inserted
Data detail print
[Dennis, c0095281, NE4 5TG, Newcastle University]

进程已结束，退出代码 0
```

2. *Academic staff class* inherits all the fields and functions declared/ defined in the employee class, and also has a specific filed called as *position title*. Furthermore, it contains a function for seeking a staff based on the ID and printing his/her information.

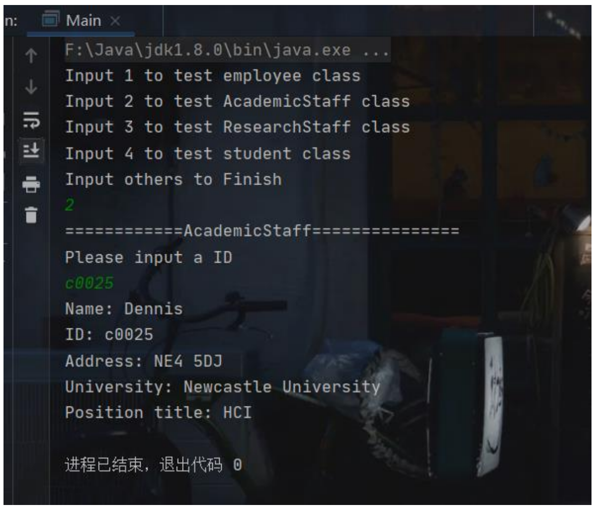


```java
public class AcademicStaff extends employee {

    private String position_title;

    public AcademicStaff(String name, String ID,String address,String university,String position_title){
        super(name, ID, address, university);
        this.position_title = position_title;
    }

    public String getposition_title() { return position_title; }

    public void setposition_title(String position_title) { this.position_title = position_title; }

    //基于ID查找信息 并打印出来
    public void SeekPrint(){
        System.out.println("============AcademicStaff===============");

        //create AcademicStaff Object
        AcademicStaff staff1 = new AcademicStaff( name: "Emma",  ID: "c0023", address: "NE1 1TT", university: "Newcaslte University", position_title: "deep learning");
        AcademicStaff staff2 = new AcademicStaff( name: "Peter",  ID: "c0024",  address: "NE1 1DN",  university: "Newcastle University",  position_title: "Java Development");
        AcademicStaff staff3 = new AcademicStaff( name: "Dennis",  ID: "c0025",  address: "NE4 5DJ",  university: "Newcastle University",  position_title: "HCI");

        List<AcademicStaff> staffList = new ArrayList<~>();

        //Add object to List
        staffList.add(staff1);
        staffList.add(staff2);
        staffList.add(staff3);

        //Use Id to search
```



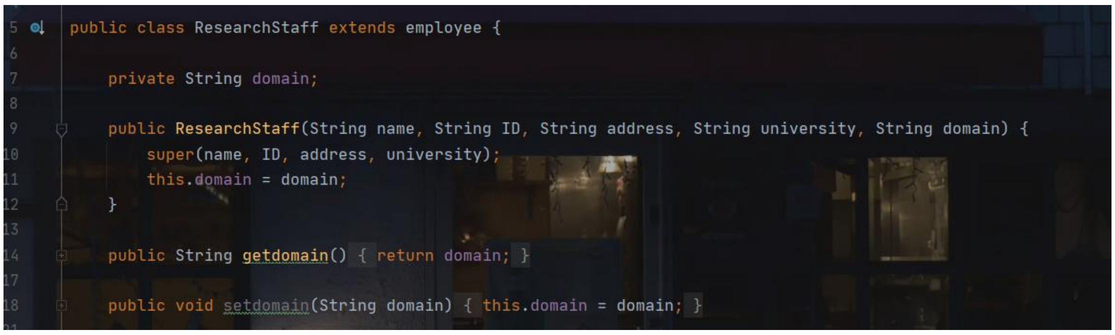
```
        List<AcademicStaff> staffList = new ArrayList<AcademicStaff>();

        //Add object to List
        staffList.add(staff1);
        staffList.add(staff2);
        staffList.add(staff3);

        //Use Id to search
        System.out.println("Please input a ID");
        Scanner sc = new Scanner(System.in);
        String input = sc.nextLine();

        String val = null;
        for(int i = 0; i < staffList.size(); i++){
            if(input.equals(staffList.get(i).getID())){
                val = staffList.get(i).getID();
                System.out.println("Name: "+staffList.get(i).getName());
                System.out.println("ID: "+staffList.get(i).getID());
                System.out.println("Address: "+staffList.get(i).getAddress());
                System.out.println("University: "+staffList.get(i).getUniversity());
                System.out.println("Position title: "+staffList.get(i).getposition_title());
            }
        }
        if(val == null){
            System.out.println("Sorry, there is not have this ID");
        }
    }
}
```

Running resilt:

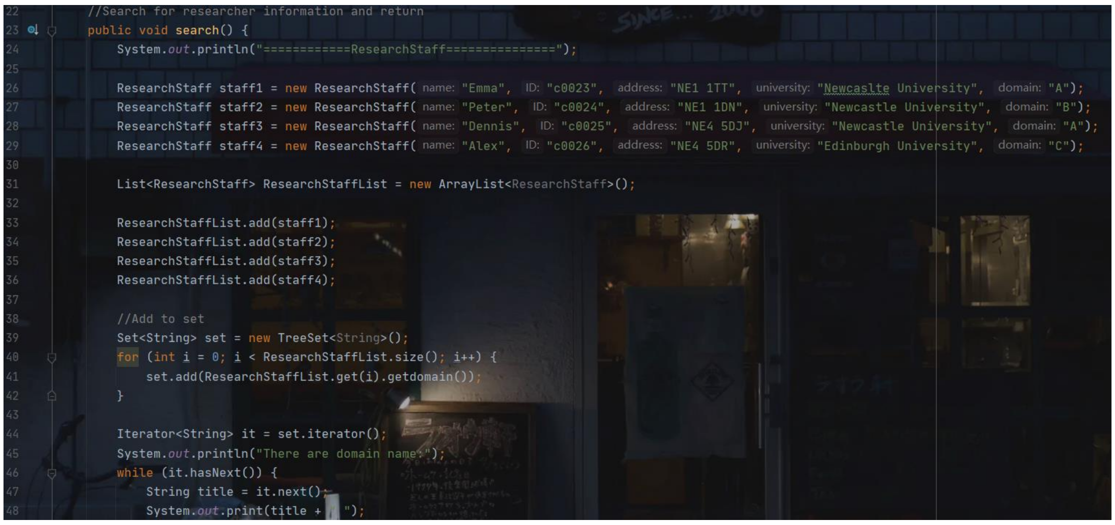


```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to test employee class
Input 2 to test AcademicStaff class
Input 3 to test ResearchStaff class
Input 4 to test student class
Input others to Finish
2
===========AcademicStaff==============
Please input a ID
c0025
Name: Dennis
ID: c0025
Address: NE4 5DJ
University: Newcastle University
Position title: HCI

进程已结束，退出代码 0
```

3. Research staff class inherits all the fields and functions declared/ defined in the employee class. It has a private variable for keeping research domains and a list for maintaining the list of researchers who works in a specific domain determined by an object. The class should implement a function for returning the staffs whose research are on the identified domain by an object. (The function name is *search()*).

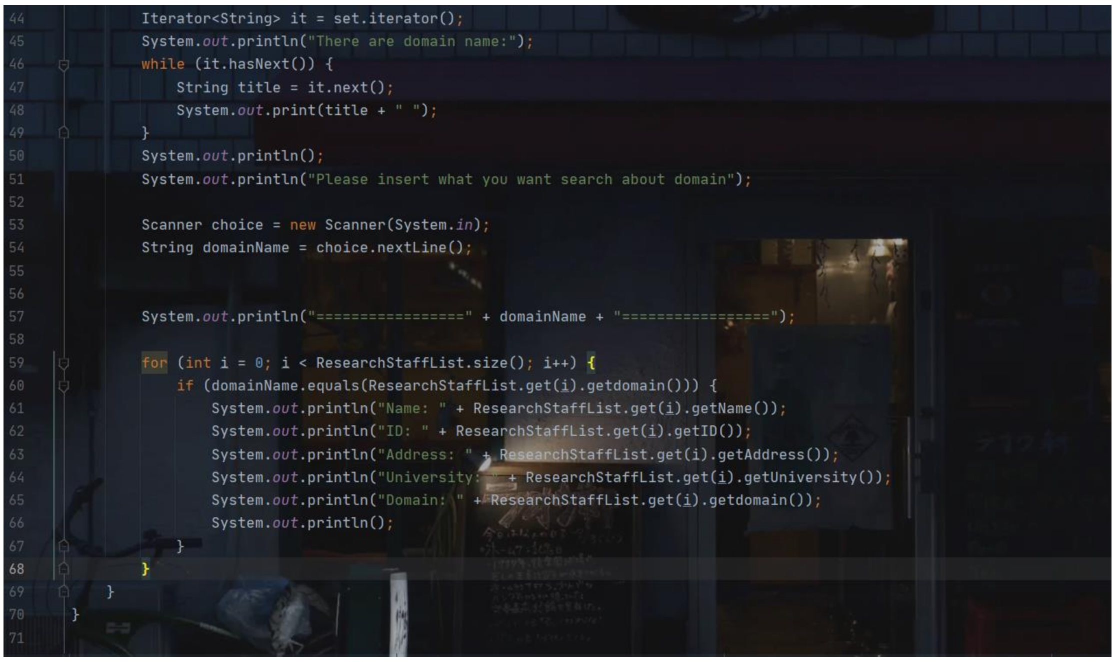


```java
public class ResearchStaff extends employee {

    private String domain;

    public ResearchStaff(String name, String ID, String address, String university, String domain) {
        super(name, ID, address, university);
        this.domain = domain;
    }

    public String getdomain() { return domain; }

    public void setdomain(String domain) { this.domain = domain; }
```



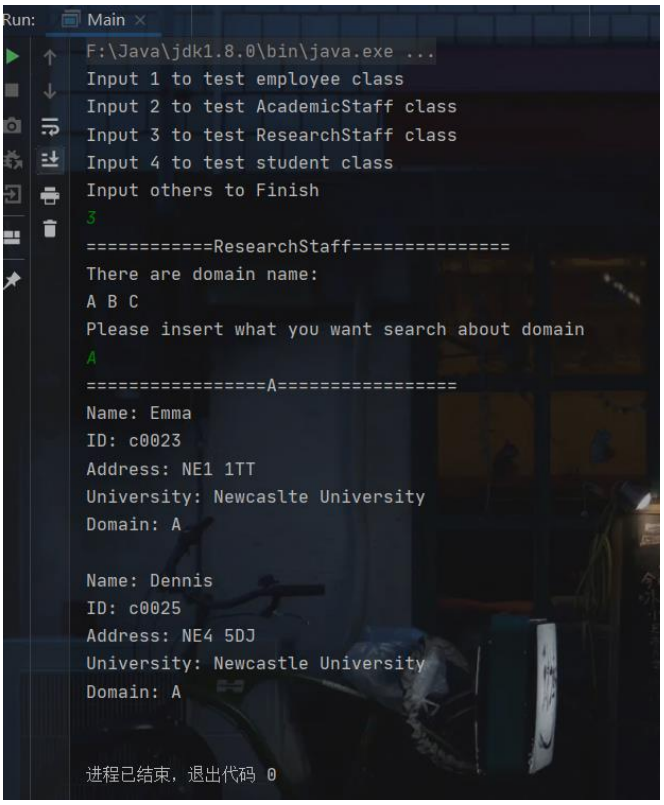
```java
    //Search for researcher information and return
    public void search() {
        System.out.println("===========ResearchStaff==============");

        ResearchStaff staff1 = new ResearchStaff( name: "Emma",  ID: "c0023",  address: "NE1 1TT",  university: "Newcaslte University",  domain: "A");
        ResearchStaff staff2 = new ResearchStaff( name: "Peter",  ID: "c0024",  address: "NE1 1DN",  university: "Newcastle University",  domain: "B");
        ResearchStaff staff3 = new ResearchStaff( name: "Dennis",  ID: "c0025",  address: "NE4 5DJ",  university: "Newcastle University",  domain: "A");
        ResearchStaff staff4 = new ResearchStaff( name: "Alex",  ID: "c0026",  address: "NE4 5DR",  university: "Edinburgh University",  domain: "C");

        List<ResearchStaff> ResearchStaffList = new ArrayList<ResearchStaff>();

        ResearchStaffList.add(staff1);
        ResearchStaffList.add(staff2);
        ResearchStaffList.add(staff3);
        ResearchStaffList.add(staff4);

        //Add to set
        Set<String> set = new TreeSet<String>();
        for (int i = 0; i < ResearchStaffList.size(); i++) {
            set.add(ResearchStaffList.get(i).getdomain());
        }

        Iterator<String> it = set.iterator();
        System.out.println("There are domain name:");
        while (it.hasNext()) {
            String title = it.next();
            System.out.print(title + " ");
```

In the set set, the same value will only be stored in one. When the domain is repeated, only one value will be saved as a tag.(The domain values of staff1 and staff2 are both A). According to the value in set, receiving the input value can output the corresponding staff information. ‘it’ is used to output the retrieved domain value.



```java
        Iterator<String> it = set.iterator();
        System.out.println("There are domain name:");
        while (it.hasNext()) {
            String title = it.next();
            System.out.print(title + " ");
        }
        System.out.println();
        System.out.println("Please insert what you want search about domain");

        Scanner choice = new Scanner(System.in);
        String domainName = choice.nextLine();


        System.out.println("================" + domainName + "================");

        for (int i = 0; i < ResearchStaffList.size(); i++) {
            if (domainName.equals(ResearchStaffList.get(i).getdomain())) {
                System.out.println("Name: " + ResearchStaffList.get(i).getName());
                System.out.println("ID: " + ResearchStaffList.get(i).getID());
                System.out.println("Address: " + ResearchStaffList.get(i).getAddress());
                System.out.println("University: " + ResearchStaffList.get(i).getUniversity());
                System.out.println("Domain: " + ResearchStaffList.get(i).getdomain());
                System.out.println();
            }
        }
    }
}
```

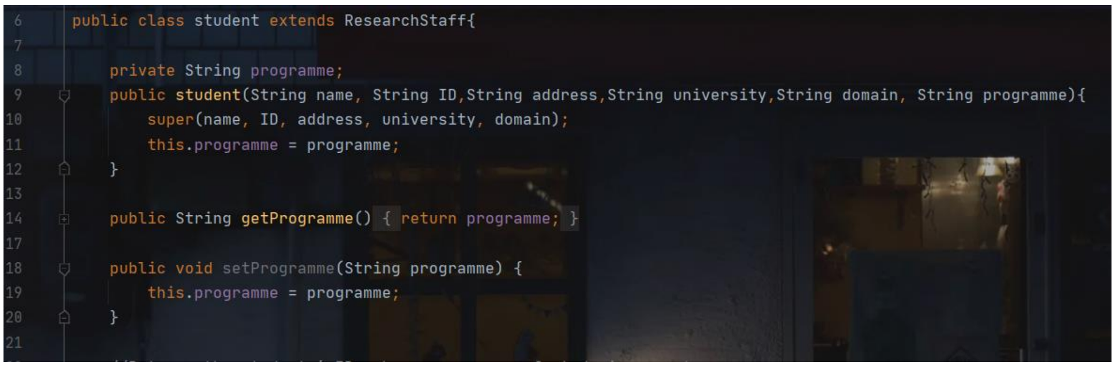
Running result:



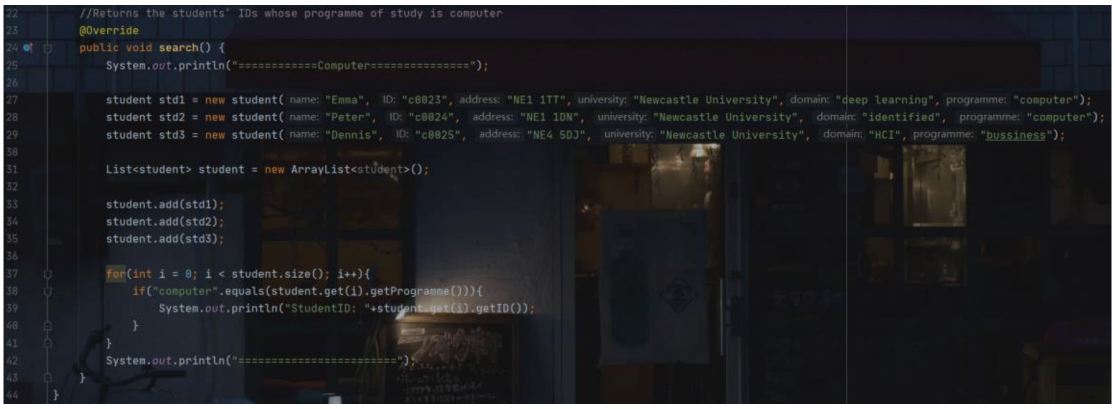
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to test employee class
Input 2 to test AcademicStaff class
Input 3 to test ResearchStaff class
Input 4 to test student class
Input others to Finish
3
============ResearchStaff==============
There are domain name:
A B C
Please insert what you want search about domain
A
================A================
Name: Emma
ID: c0023
Address: NE1 1TT
University: Newcaslte University
Domain: A

Name: Dennis
ID: c0025
Address: NE4 5DJ
University: Newcastle University
Domain: A


进程已结束，退出代码 0
```

4. *Student class* has a programme of study field and also contains those declared in the *Research staff* class. It has a function that returns the

students' IDs whose programme of study is *computer*. (The function name is *search()*).

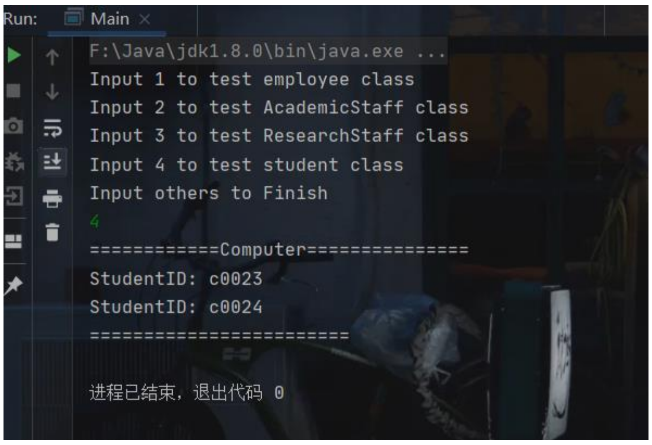


```java
public class student extends ResearchStaff{

    private String programme;
    public student(String name, String ID,String address,String university,String domain, String programme){
        super(name, ID, address, university, domain);
        this.programme = programme;
    }

    public String getProgramme() { return programme; }

    public void setProgramme(String programme) {
        this.programme = programme;
    }

```



```java
    //Returns the students' IDs whose programme of study is computer
    @Override
    public void search() {
        System.out.println("===========Computer==============");

        student std1 = new student( name: "Emma",  ID: "c0023", address: "NE1 1TT", university: "Newcastle University", domain: "deep learning", programme: "computer");
        student std2 = new student( name: "Peter",  ID: "c0024",  address: "NE1 1DN",  university: "Newcastle University",  domain: "identified",  programme: "computer");
        student std3 = new student( name: "Dennis",  ID: "c0025",  address: "NE4 5DJ",  university: "Newcastle University",  domain: "HCI", programme: "bussiness");

        List<student> student = new ArrayList<student>();

        student.add(std1);
        student.add(std2);
        student.add(std3);

        for(int i = 0; i < student.size(); i++){
            if("computer".equals(student.get(i).getProgramme())){
                System.out.println("StudentID: "+student.get(i).getID());
            }
        }
        System.out.println("======================");
    }
}
```

Running result



```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to test employee class
Input 2 to test AcademicStaff class
Input 3 to test ResearchStaff class
Input 4 to test student class
Input others to Finish
4
===========Computer==============
StudentID: c0023
StudentID: c0024
======================

进程已结束，退出代码 0
```

Q3: Design a class called *Meeting* to represent meetings in a diary. The Meeting class has the following fields:

time of the meeting represented as string in hours and minutes,

location of the meeting (such as “room 205”),

subject that represents the meeting’s subject (such as “Examiner’s meeting”).

Time, location and subject are stored as strings. The class should include a constructor and the following methods:

setTime: to set the time.

setLocation: to set the location.

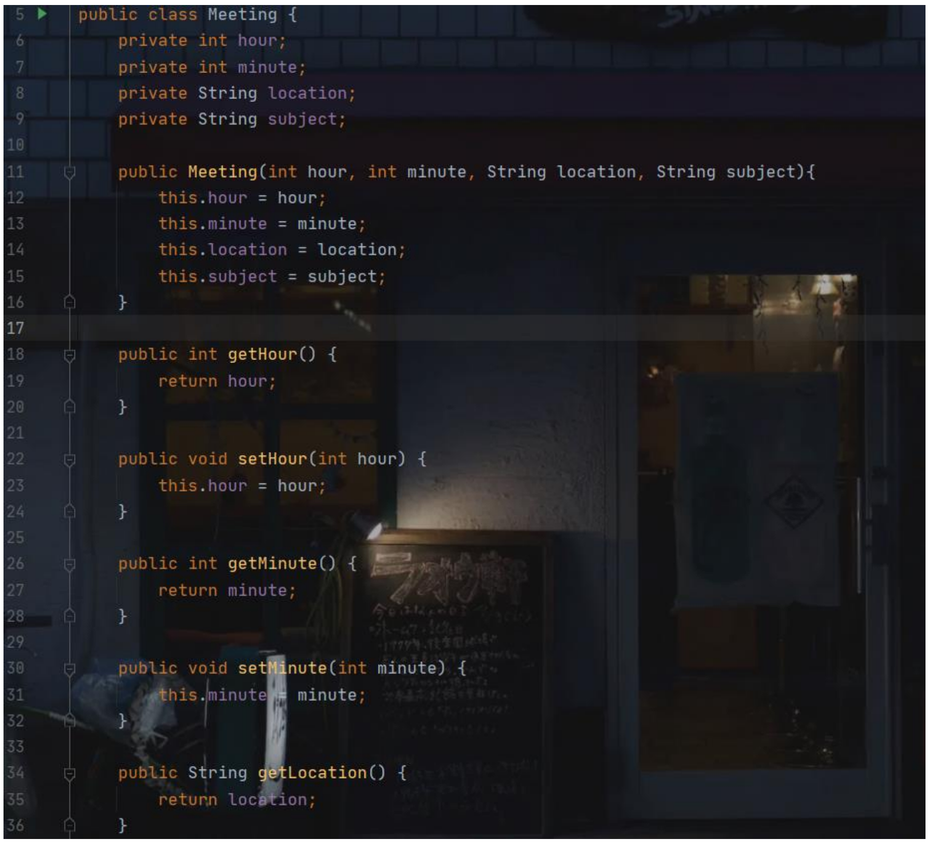
setSubject: to set the subject.

getSubject: to return the subject of the meeting.

printDetails: to print all information of a meeting in the following form:

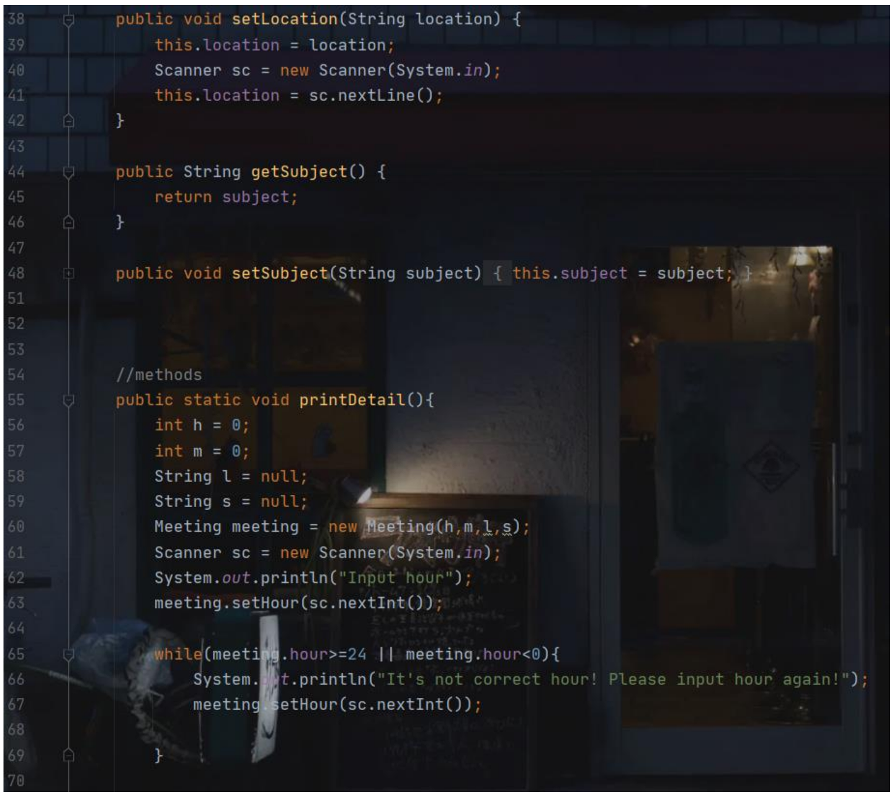
Meeting in room 205 at 12:30; Subject: Examiner’s meeting.

Meeting class(1):



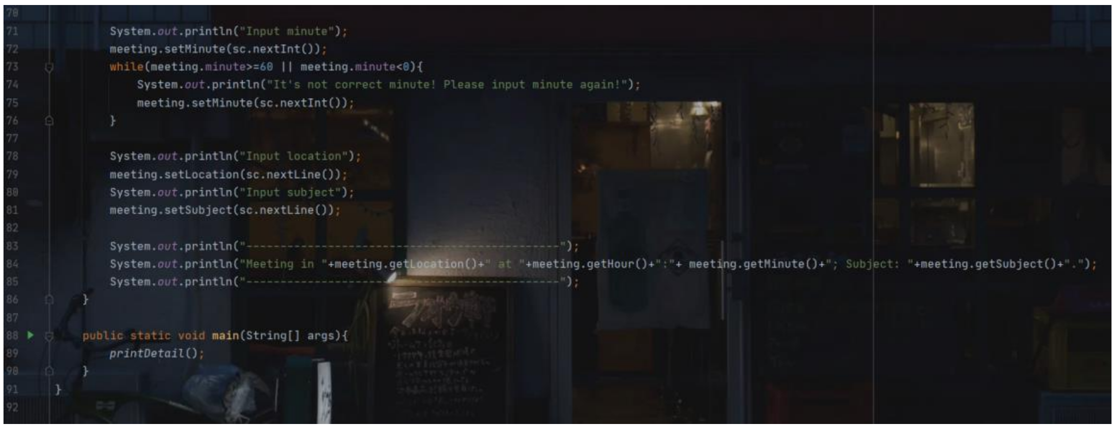
```java
public class Meeting {
    private int hour;
    private int minute;
    private String location;
    private String subject;

    public Meeting(int hour, int minute, String location, String subject){
        this.hour = hour;
        this.minute = minute;
        this.location = location;
        this.subject = subject;
    }

    public int getHour() {
        return hour;
    }

    public void setHour(int hour) {
        this.hour = hour;
    }

    public int getMinute() {
        return minute;
    }

    public void setMinute(int minute) {
        this.minute = minute;
    }

    public String getLocation() {
        return location;
    }
```

Meeting class (2):



```java
    public void setLocation(String location) {
        this.location = location;
        Scanner sc = new Scanner(System.in);
        this.location = sc.nextLine();
    }

    public String getSubject() {
        return subject;
    }

    public void setSubject(String subject) { this.subject = subject; }



    //methods
    public static void printDetail(){
        int h = 0;
        int m = 0;
        String l = null;
        String s = null;
        Meeting meeting = new Meeting(h,m,l,s);
        Scanner sc = new Scanner(System.in);
        System.out.println("Input hour");
        meeting.setHour(sc.nextInt());

        while(meeting.hour>=24 || meeting.hour<0){
            System.out.println("It's not correct hour! Please input hour again!");
            meeting.setHour(sc.nextInt());

        }

```

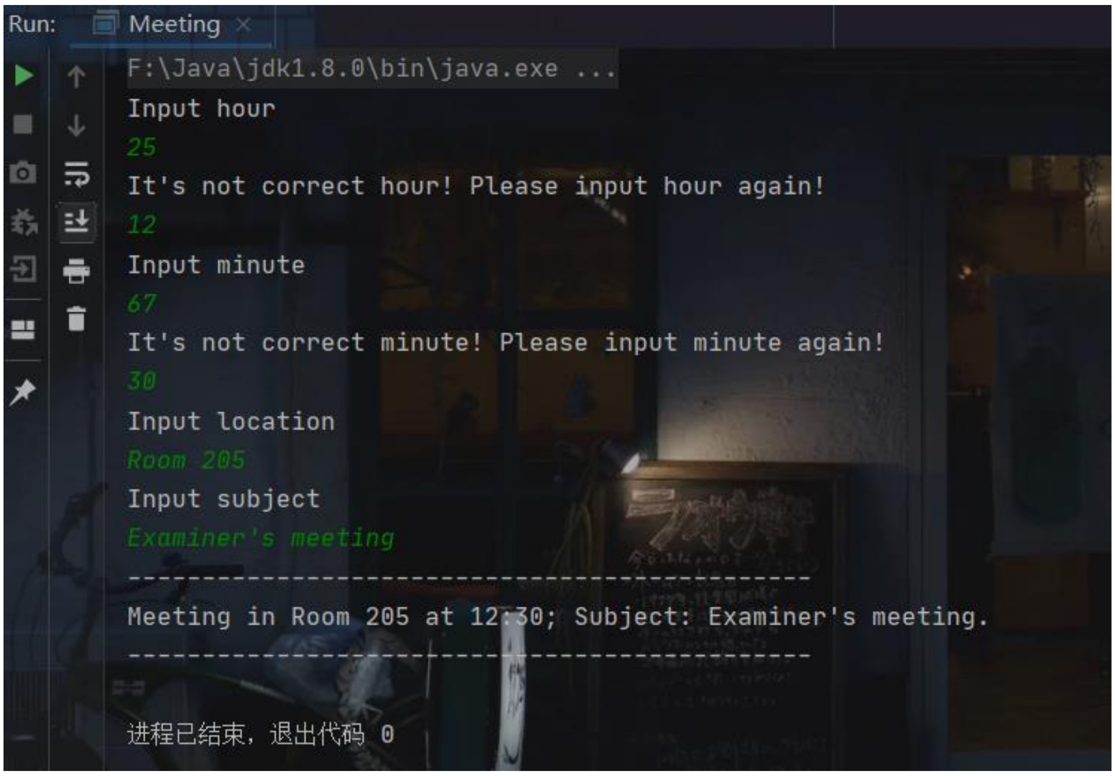
Meeting class (3):



```java
        System.out.println("Input minute");
        meeting.setMinute(sc.nextInt());
        while(meeting.minute>=60 || meeting.minute<0){
            System.out.println("It's not correct minute! Please input minute again!");
            meeting.setMinute(sc.nextInt());
        }

        System.out.println("Input location");
        meeting.setLocation(sc.nextLine());
        System.out.println("Input subject");
        meeting.setSubject(sc.nextLine());

        System.out.println("---------------------------------------------");
        System.out.println("Meeting in "+meeting.getLocation()+" at "+meeting.getHour()+":"+ meeting.getMinute()+"; Subject: "+meeting.getSubject()+".");
        System.out.println("---------------------------------------------");
    }

    public static void main(String[] args){
        printDetail();
    }
}
```

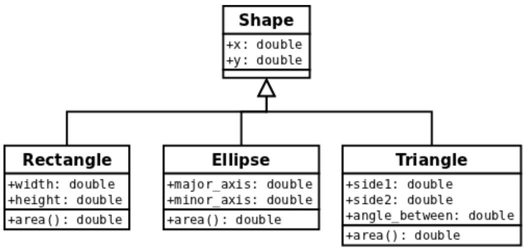
(Add method for judging the wrong input hour and minute).

Running result:



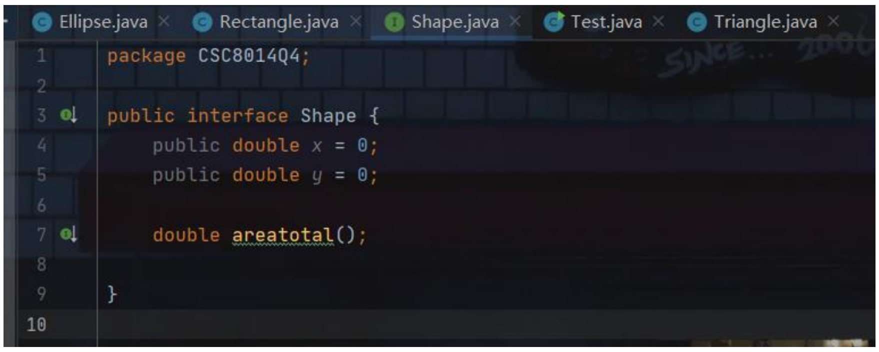
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input hour
25
It's not correct hour! Please input hour again!
12
Input minute
67
It's not correct minute! Please input minute again!
30
Input location
Room 205
Input subject
Examiner's meeting
---------------------------------------------
Meeting in Room 205 at 12:30; Subject: Examiner's meeting.
---------------------------------------------

进程已结束，退出代码 0
```

Q4: Figure 1 illustrates a class for a *shape* that involves three sub-classes, namely rectangle, ellipse and triangle.



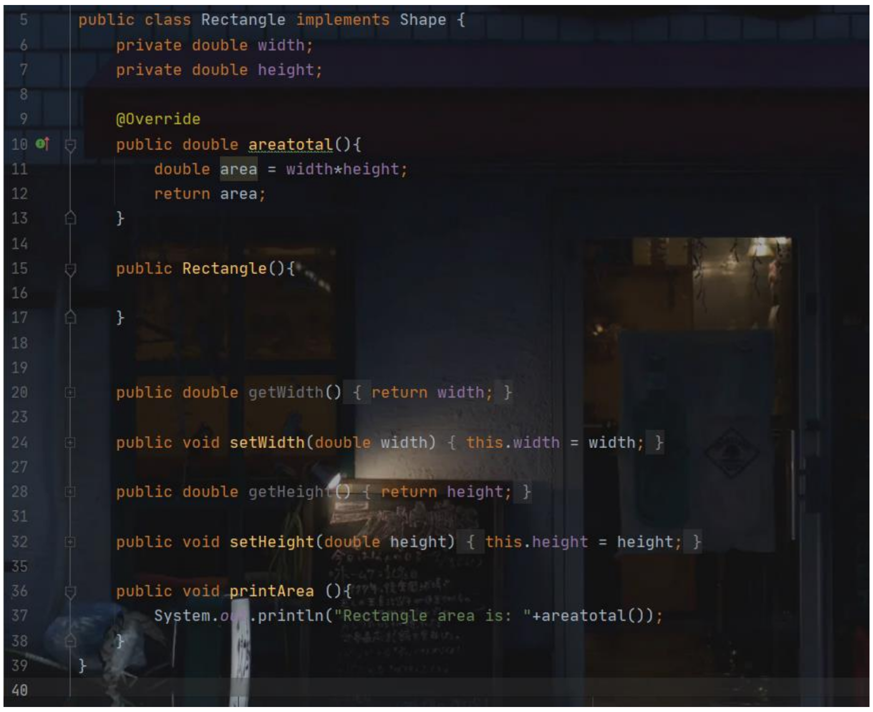
Given the figure and the details of classes. Implement an interface for the classes.

Shape class: (As interface)



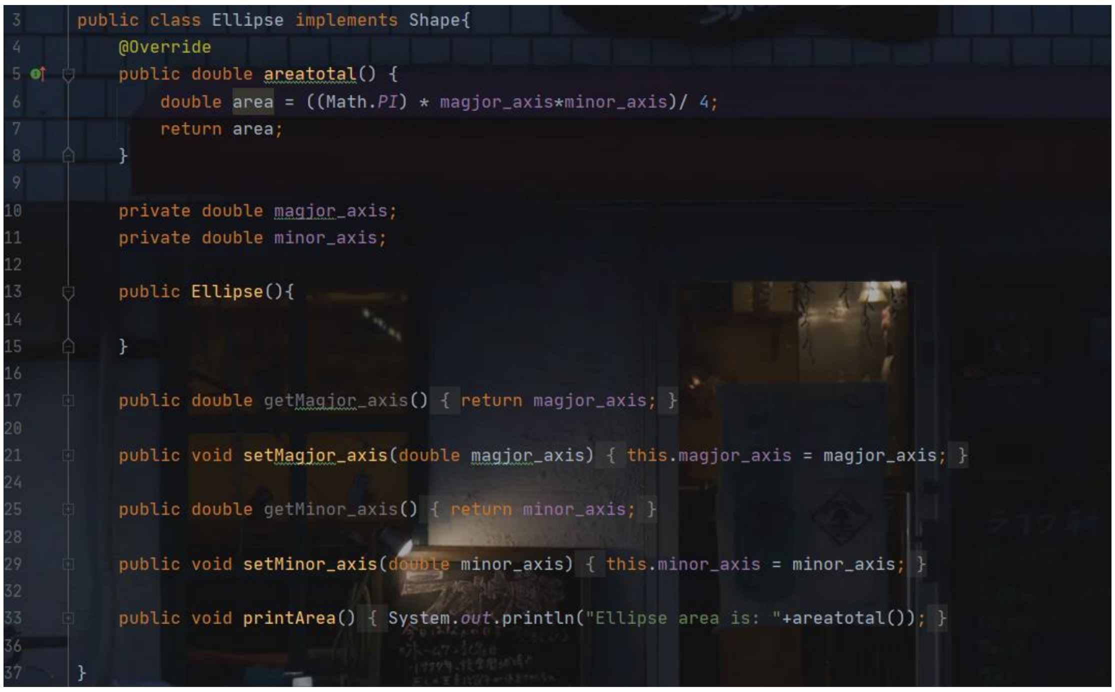
```java
package CSC8014Q4;

public interface Shape {
    public double x = 0;
    public double y = 0;

    double areatotal();

}
```

Rectangle class:



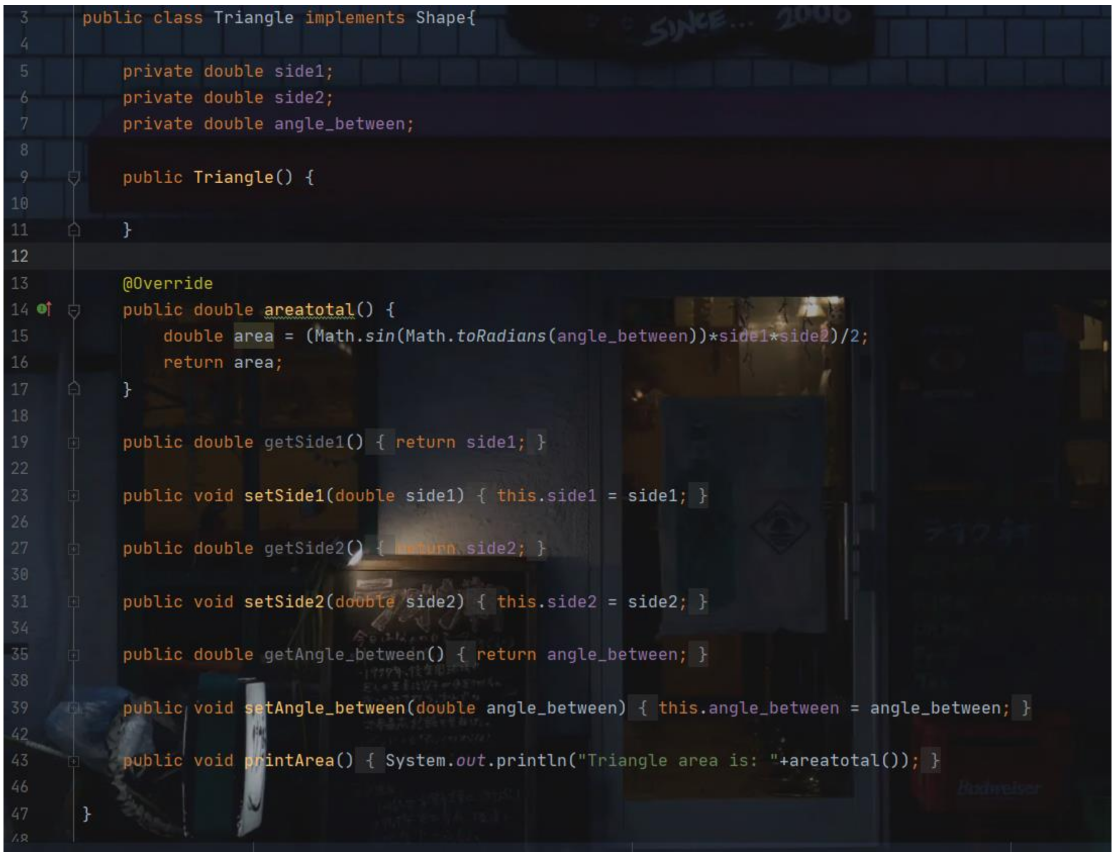
```java
public class Rectangle implements Shape {
    private double width;
    private double height;

    @Override
    public double areatotal(){
        double area = width*height;
        return area;
    }

    public Rectangle(){

    }


    public double getWidth() { return width; }

    public void setWidth(double width) { this.width = width; }

    public double getHeight() { return height; }

    public void setHeight(double height) { this.height = height; }

    public void printArea (){
        System.out.println("Rectangle area is: "+areatotal());
    }
}
```

Ellipse class:



```java
public class Ellipse implements Shape{
    @Override
    public double areatotal() {
        double area = ((Math.PI) * magjor_axis*minor_axis)/ 4;
        return area;
    }

    private double magjor_axis;
    private double minor_axis;

    public Ellipse(){

    }

    public double getMagjor_axis() { return magjor_axis; }

    public void setMagjor_axis(double magjor_axis) { this.magjor_axis = magjor_axis; }

    public double getMinor_axis() { return minor_axis; }

    public void setMinor_axis(double minor_axis) { this.minor_axis = minor_axis; }

    public void printArea() { System.out.println("Ellipse area is: "+areatotal()); }

}
```

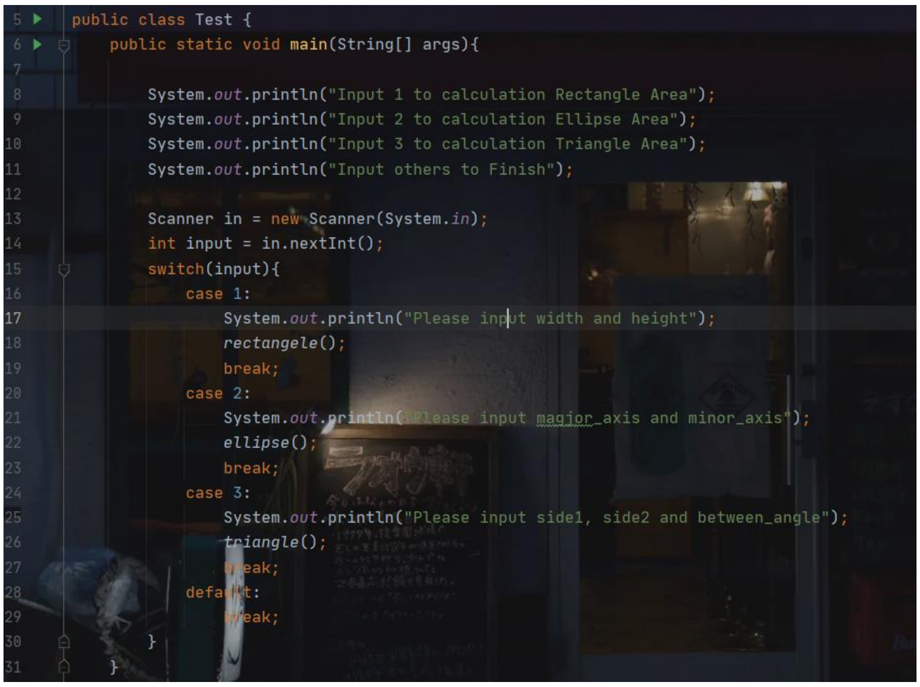
Triangle class:



```java
public class Triangle implements Shape{

    private double side1;
    private double side2;
    private double angle_between;

    public Triangle() {

    }

    @Override
    public double areatotal() {
        double area = (Math.sin(Math.toRadians(angle_between))*side1*side2)/2;
        return area;
    }

    public double getSide1() { return side1; }

    public void setSide1(double side1) { this.side1 = side1; }

    public double getSide2() { return side2; }

    public void setSide2(double side2) { this.side2 = side2; }

    public double getAngle_between() { return angle_between; }

    public void setAngle_between(double angle_between) { this.angle_between = angle_between; }

    public void printArea() { System.out.println("Triangle area is: "+areatotal()); }

}
```

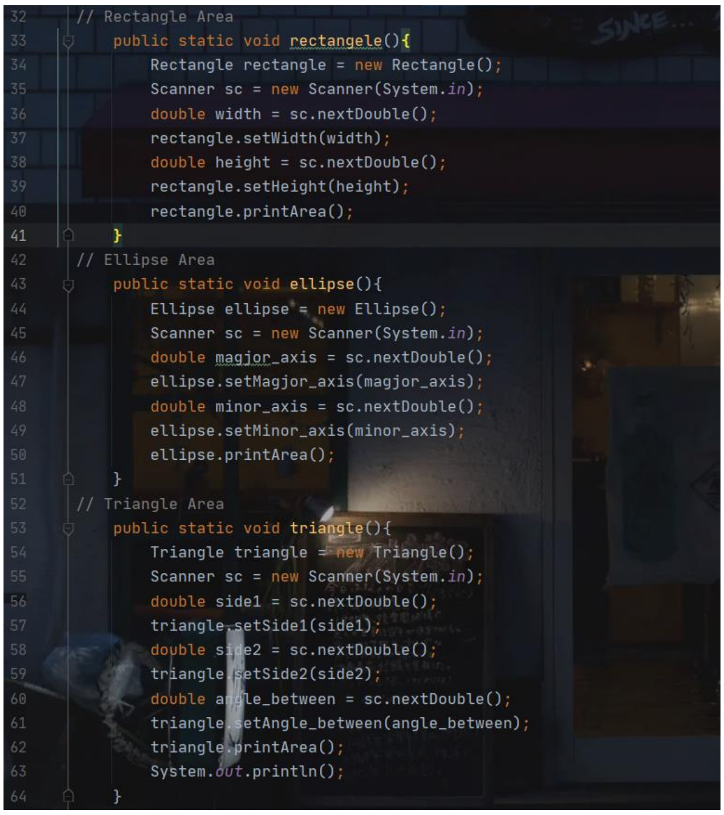
Write the code for the *area()* functions and implement the main function, which tests the object's areas regarding arbitrary arguments.

Test class (1):



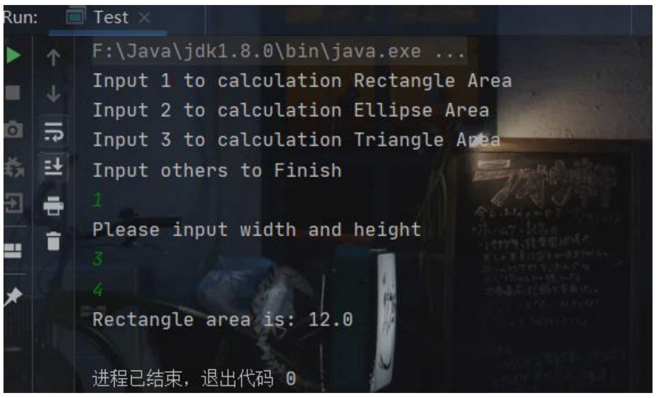
```java
public class Test {
    public static void main(String[] args){

        System.out.println("Input 1 to calculation Rectangle Area");
        System.out.println("Input 2 to calculation Ellipse Area");
        System.out.println("Input 3 to calculation Triangle Area");
        System.out.println("Input others to Finish");

        Scanner in = new Scanner(System.in);
        int input = in.nextInt();
        switch(input){
            case 1:
                System.out.println("Please input width and height");
                rectangele();
                break;
            case 2:
                System.out.println("Please input magjor_axis and minor_axis");
                ellipse();
                break;
            case 3:
                System.out.println("Please input side1, side2 and between_angle");
                triangle();
                break;
            default:
                break;
        }
    }
```

Test class (2):



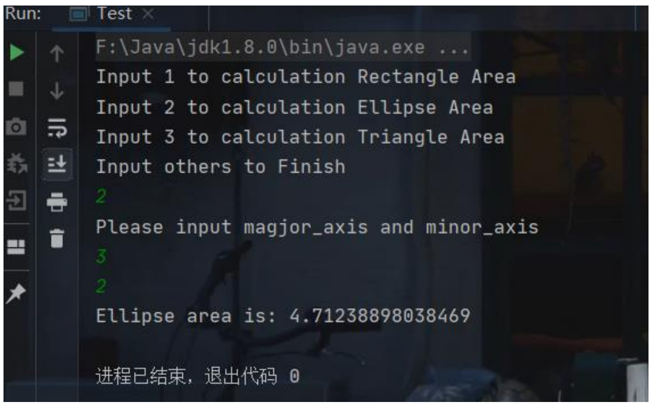
```java
// Rectangle Area
    public static void rectangele(){
        Rectangle rectangle = new Rectangle();
        Scanner sc = new Scanner(System.in);
        double width = sc.nextDouble();
        rectangle.setWidth(width);
        double height = sc.nextDouble();
        rectangle.setHeight(height);
        rectangle.printArea();
    }
// Ellipse Area
    public static void ellipse(){
        Ellipse ellipse = new Ellipse();
        Scanner sc = new Scanner(System.in);
        double magjor_axis = sc.nextDouble();
        ellipse.setMagjor_axis(magjor_axis);
        double minor_axis = sc.nextDouble();
        ellipse.setMinor_axis(minor_axis);
        ellipse.printArea();
    }
// Triangle Area
    public static void triangle(){
        Triangle triangle = new Triangle();
        Scanner sc = new Scanner(System.in);
        double side1 = sc.nextDouble();
        triangle.setSide1(side1);
        double side2 = sc.nextDouble();
        triangle.setSide2(side2);
        double angle_between = sc.nextDouble();
        triangle.setAngle_between(angle_between);
        triangle.printArea();
        System.out.println();
    }
```

Rectangle Area result:



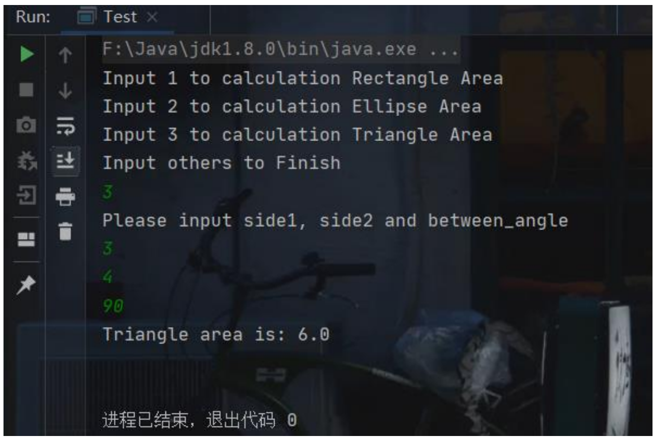
```
Run:    Test
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to calculation Rectangle Area
Input 2 to calculation Ellipse Area
Input 3 to calculation Triangle Area
Input others to Finish
1
Please input width and height
3
4
Rectangle area is: 12.0

进程已结束，退出代码 0
```

Ellipse Area result:



```
Run:    Test
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to calculation Rectangle Area
Input 2 to calculation Ellipse Area
Input 3 to calculation Triangle Area
Input others to Finish
2
Please input magjor_axis and minor_axis
3
2
Ellipse area is: 4.71238898038469

进程已结束，退出代码 0
```

Triangle Area result:

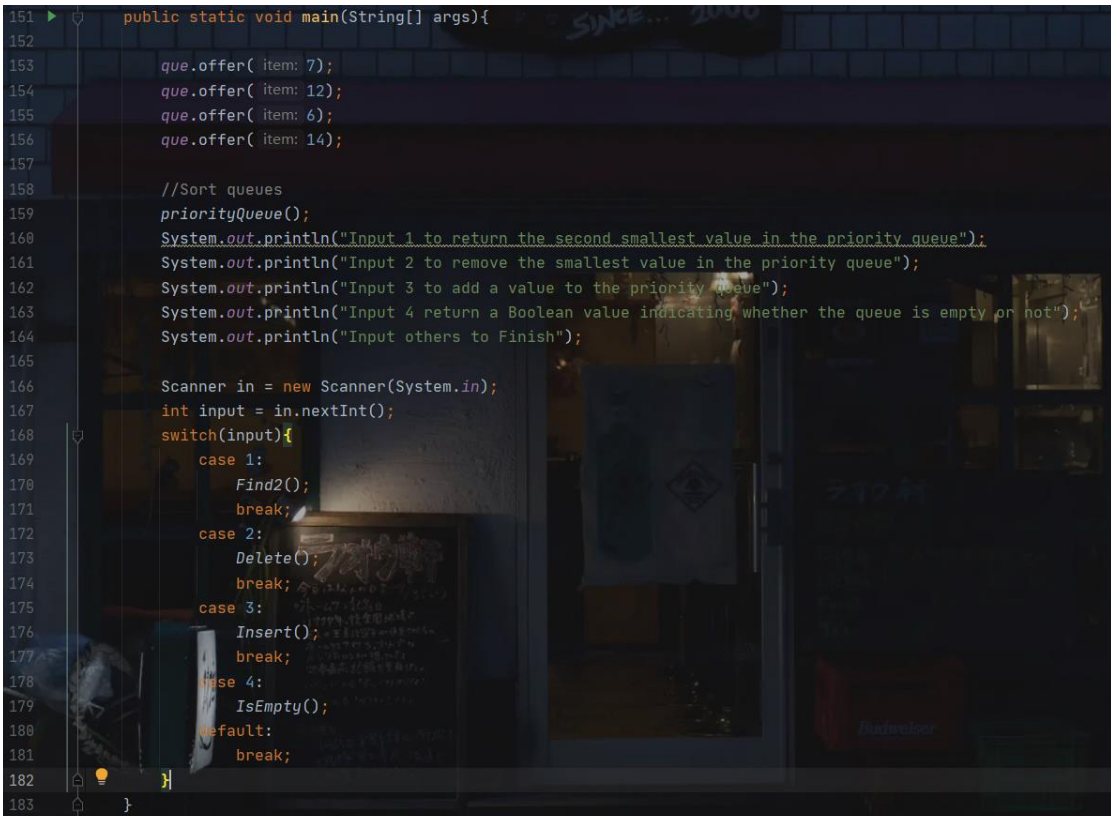


```
Run:    Test
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to calculation Rectangle Area
Input 2 to calculation Ellipse Area
Input 3 to calculation Triangle Area
Input others to Finish
3
Please input side1, side2 and between_angle
3
4
90
Triangle area is: 6.0

进程已结束，退出代码 0
```

Q5: Create a priority queue class with the following methods:

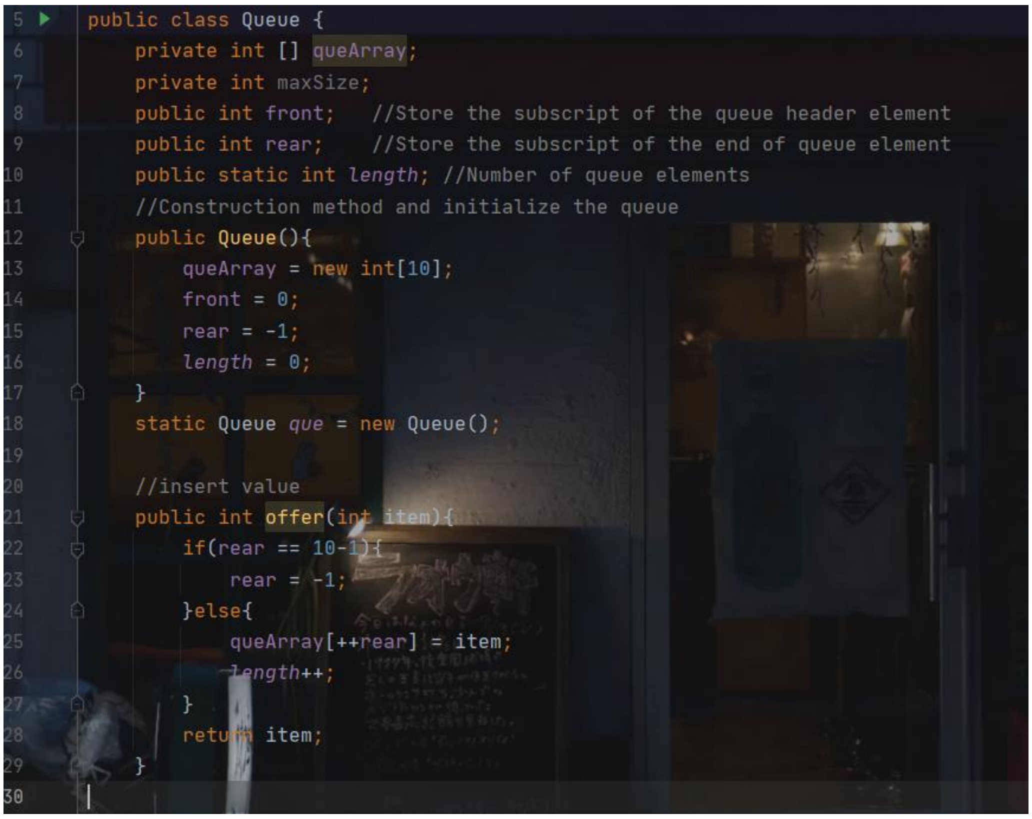
Test queue is: 7 12 6 14

Main class:



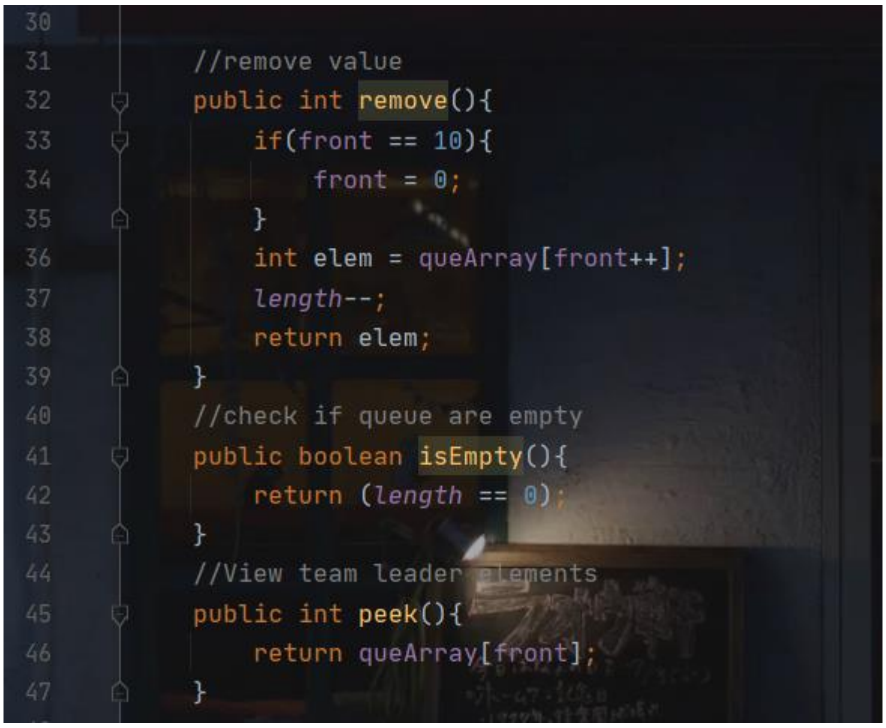
```java
    public static void main(String[] args){

        que.offer( item: 7);
        que.offer( item: 12);
        que.offer( item: 6);
        que.offer( item: 14);

        //Sort queues
        priorityQueue();
        System.out.println("Input 1 to return the second smallest value in the priority queue");
        System.out.println("Input 2 to remove the smallest value in the priority queue");
        System.out.println("Input 3 to add a value to the priority queue");
        System.out.println("Input 4 return a Boolean value indicating whether the queue is empty or not");
        System.out.println("Input others to Finish");

        Scanner in = new Scanner(System.in);
        int input = in.nextInt();
        switch(input){
            case 1:
                Find2();
                break;
            case 2:
                Delete();
                break;
            case 3:
                Insert();
                break;
            case 4:
                IsEmpty();
            default:
                break;
        }
    }
```

Construction method and initialize the queue. Override the offer method to insert values



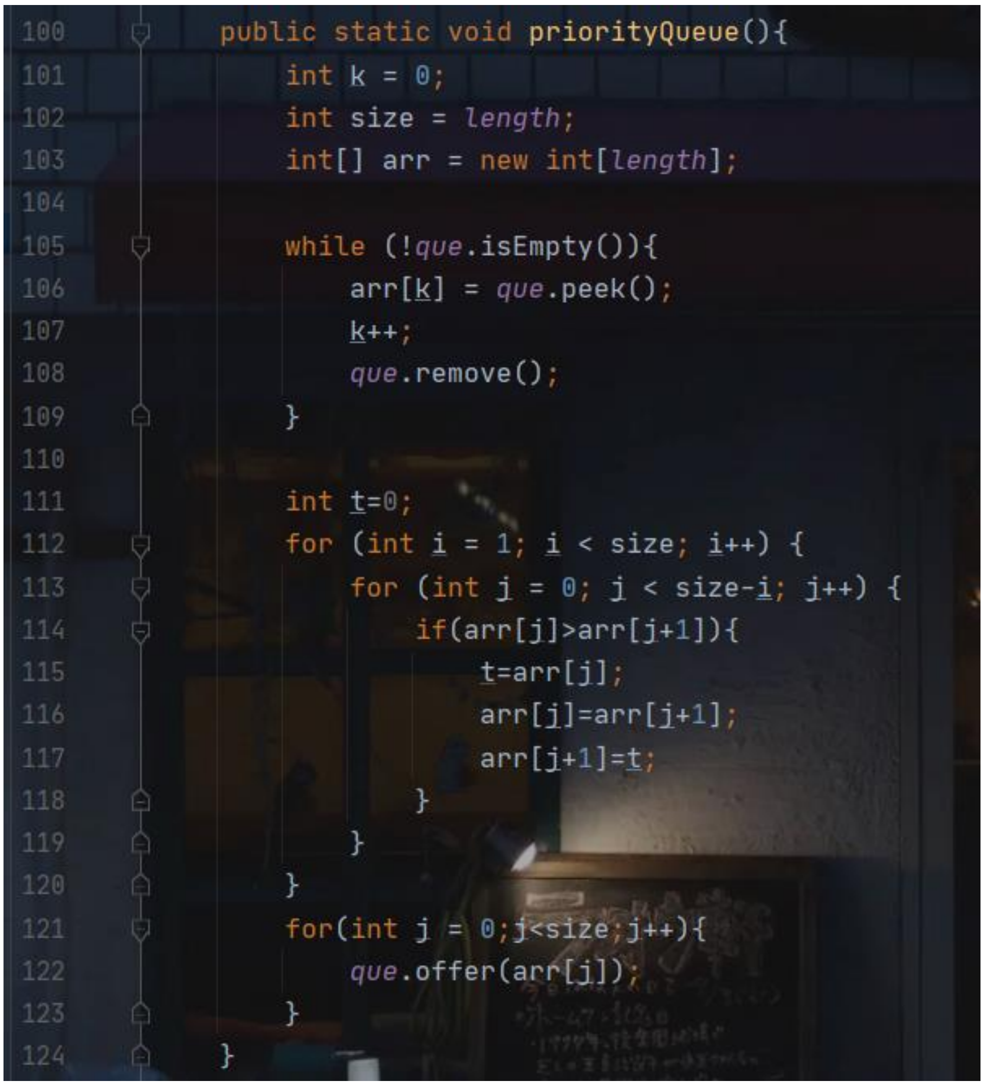
```java
public class Queue {
    private int [] queArray;
    private int maxSize;
    public int front;   //Store the subscript of the queue header element
    public int rear;    //Store the subscript of the end of queue element
    public static int length; //Number of queue elements
    //Construction method and initialize the queue
    public Queue(){
        queArray = new int[10];
        front = 0;
        rear = -1;
        length = 0;
    }
    static Queue que = new Queue();

    //insert value
    public int offer(int item){
        if(rear == 10-1){
            rear = -1;
        }else{
            queArray[++rear] = item;
            length++;
        }
        return item;
    }

```

Override the method of remove(), peek()



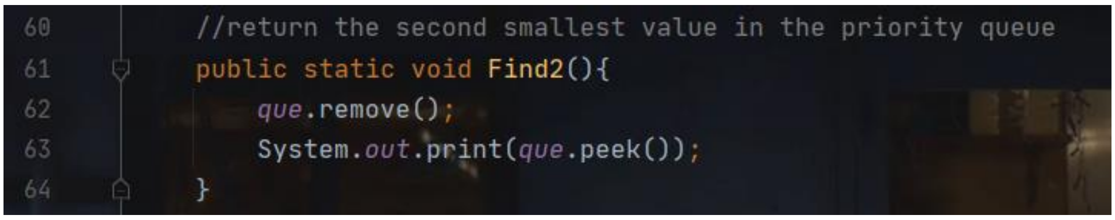
```java

    //remove value
    public int remove(){
        if(front == 10){
            front = 0;
        }
        int elem = queArray[front++];
        length--;
        return elem;
    }
    //check if queue are empty
    public boolean isEmpty(){
        return (length == 0);
    }
    //View team leader elements
    public int peek(){
        return queArray[front];
    }
```

Assign the original queue to the arr[] array. And sort in the array. Finally, put the ordered array back to the queue.



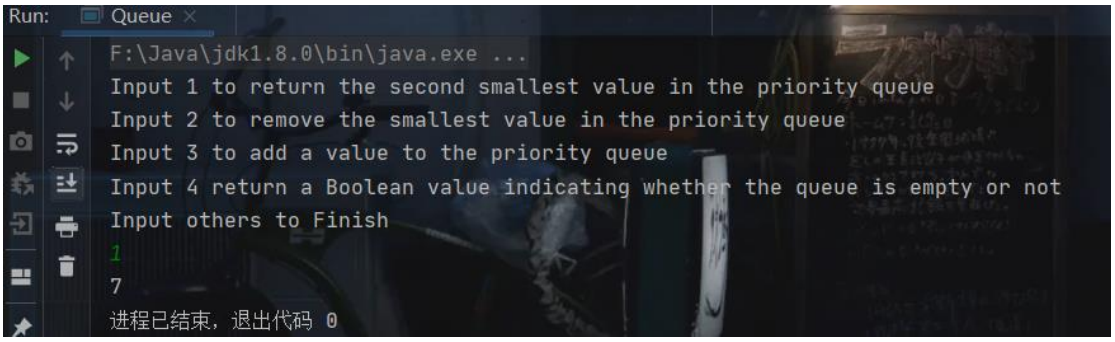
```java
    public static void priorityQueue(){
        int k = 0;
        int size = length;
        int[] arr = new int[length];

        while (!que.isEmpty()){
            arr[k] = que.peek();
            k++;
            que.remove();
        }

        int t=0;
        for (int i = 1; i < size; i++) {
            for (int j = 0; j < size-i; j++) {
                if(arr[j]>arr[j+1]){
                    t=arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=t;
                }
            }
        }
        for(int j = 0;j<size;j++){
            que.offer(arr[j]);
        }
    }
```

Find2(): return the second smallest value in the priority queue;



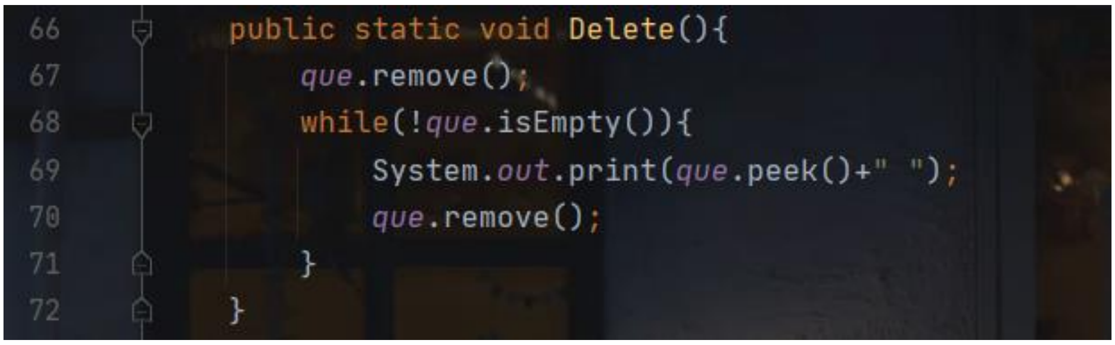
```java
    //return the second smallest value in the priority queue
    public static void Find2(){
        que.remove();
        System.out.print(que.peek());
    }
```

Running result:



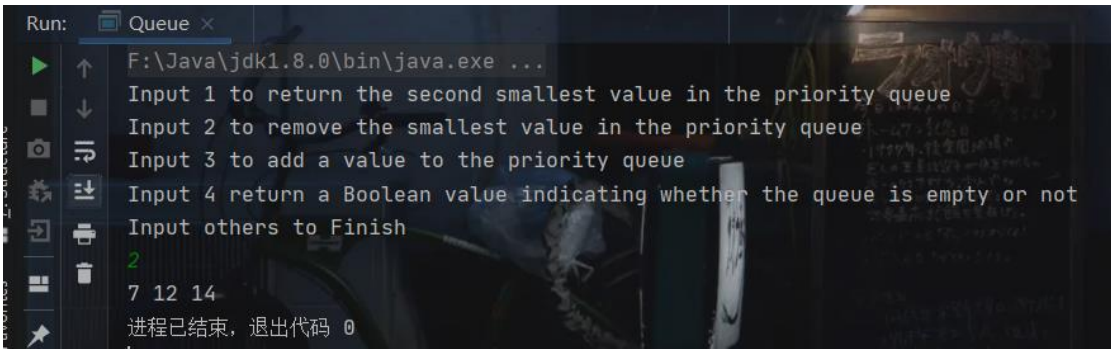
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to return the second smallest value in the priority queue
Input 2 to remove the smallest value in the priority queue
Input 3 to add a value to the priority queue
Input 4 return a Boolean value indicating whether the queue is empty or not
Input others to Finish
1
7
进程已结束，退出代码 0
```

Delete(): remove the smallest value in the priority queue;



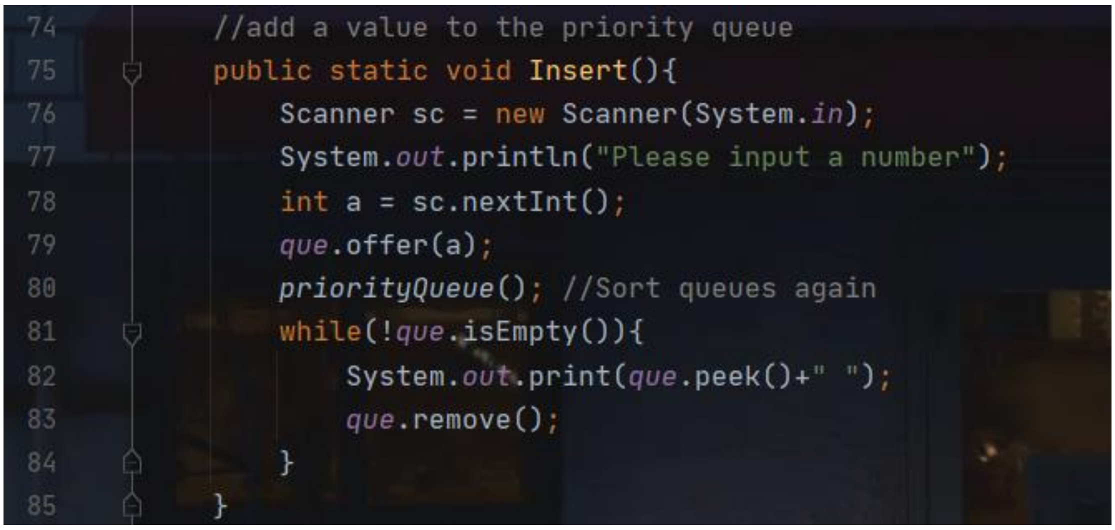
```java
    public static void Delete(){
        que.remove();
        while(!que.isEmpty()){
            System.out.print(que.peek()+" ");
            que.remove();
        }
    }
```

Running result:



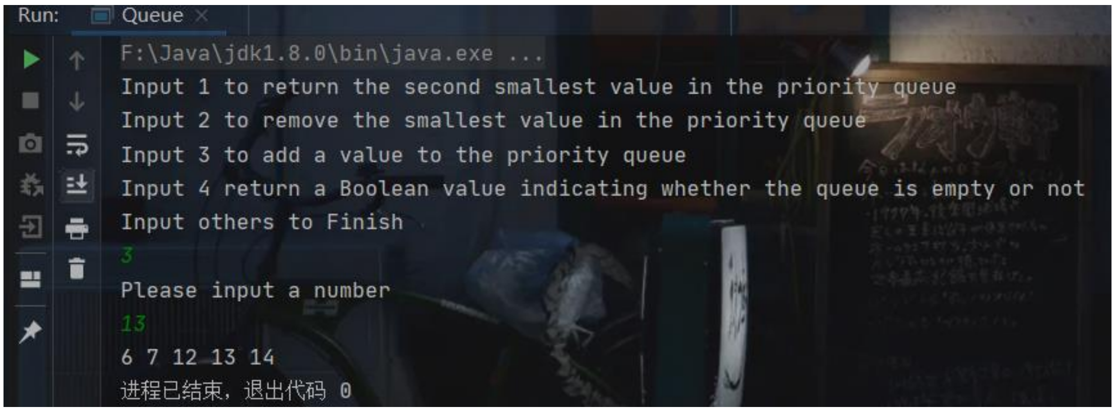
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to return the second smallest value in the priority queue
Input 2 to remove the smallest value in the priority queue
Input 3 to add a value to the priority queue
Input 4 return a Boolean value indicating whether the queue is empty or not
Input others to Finish
2
7 12 14
进程已结束，退出代码 0
```

Insert(): add a value to the queue;



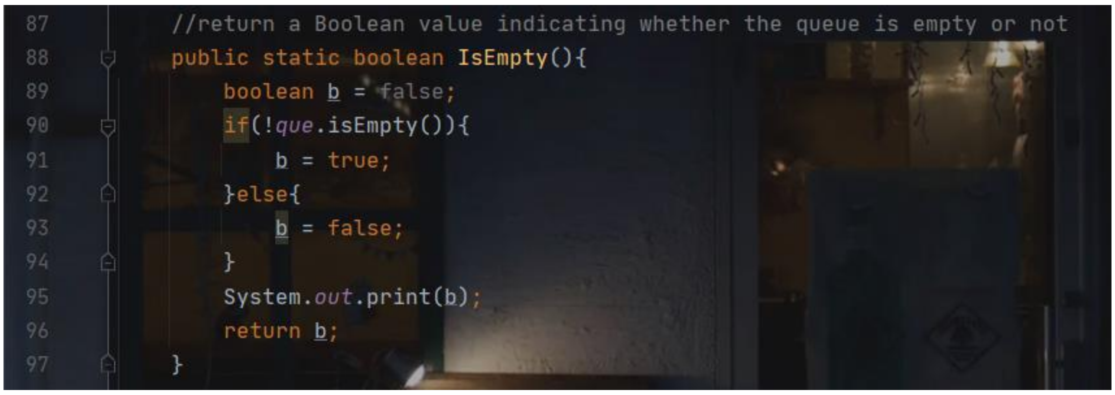
```
    //add a value to the priority queue
    public static void Insert(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Please input a number");
        int a = sc.nextInt();
        que.offer(a);
        priorityQueue(); //Sort queues again
        while(!que.isEmpty()){
            System.out.print(que.peek()+" ");
            que.remove();
        }
    }
```

Running result:



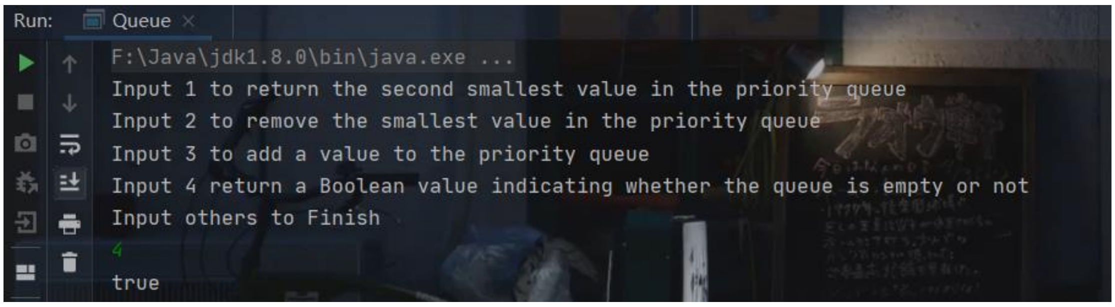
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to return the second smallest value in the priority queue
Input 2 to remove the smallest value in the priority queue
Input 3 to add a value to the priority queue
Input 4 return a Boolean value indicating whether the queue is empty or not
Input others to Finish
3
Please input a number
13
6 7 12 13 14
进程已结束，退出代码 0
```

IsEmpty(): return a Boolean value indicating whether the queue is empty or not.



```
    //return a Boolean value indicating whether the queue is empty or not
    public static boolean IsEmpty(){
        boolean b = false;
        if(!que.isEmpty()){
            b = true;
        }else{
            b = false;
        }
        System.out.print(b);
        return b;
    }
```

Running result:

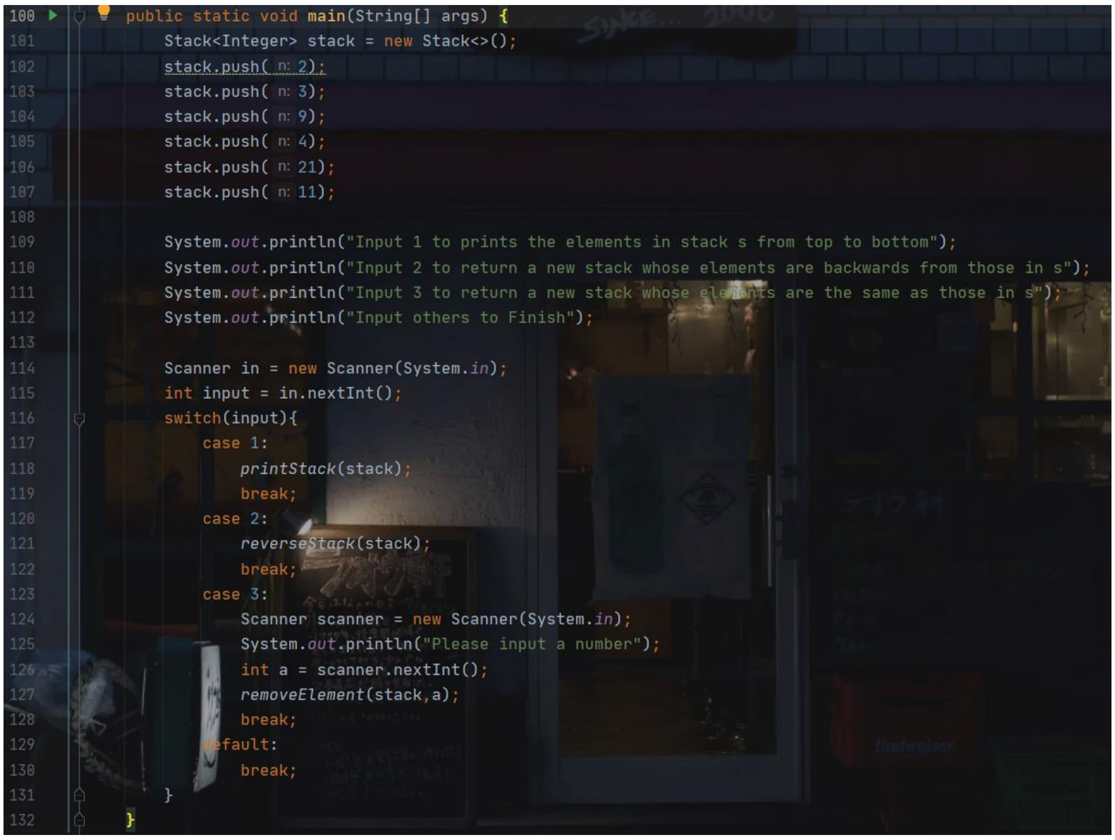


```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to return the second smallest value in the priority queue
Input 2 to remove the smallest value in the priority queue
Input 3 to add a value to the priority queue
Input 4 return a Boolean value indicating whether the queue is empty or not
Input others to Finish
4
true
```

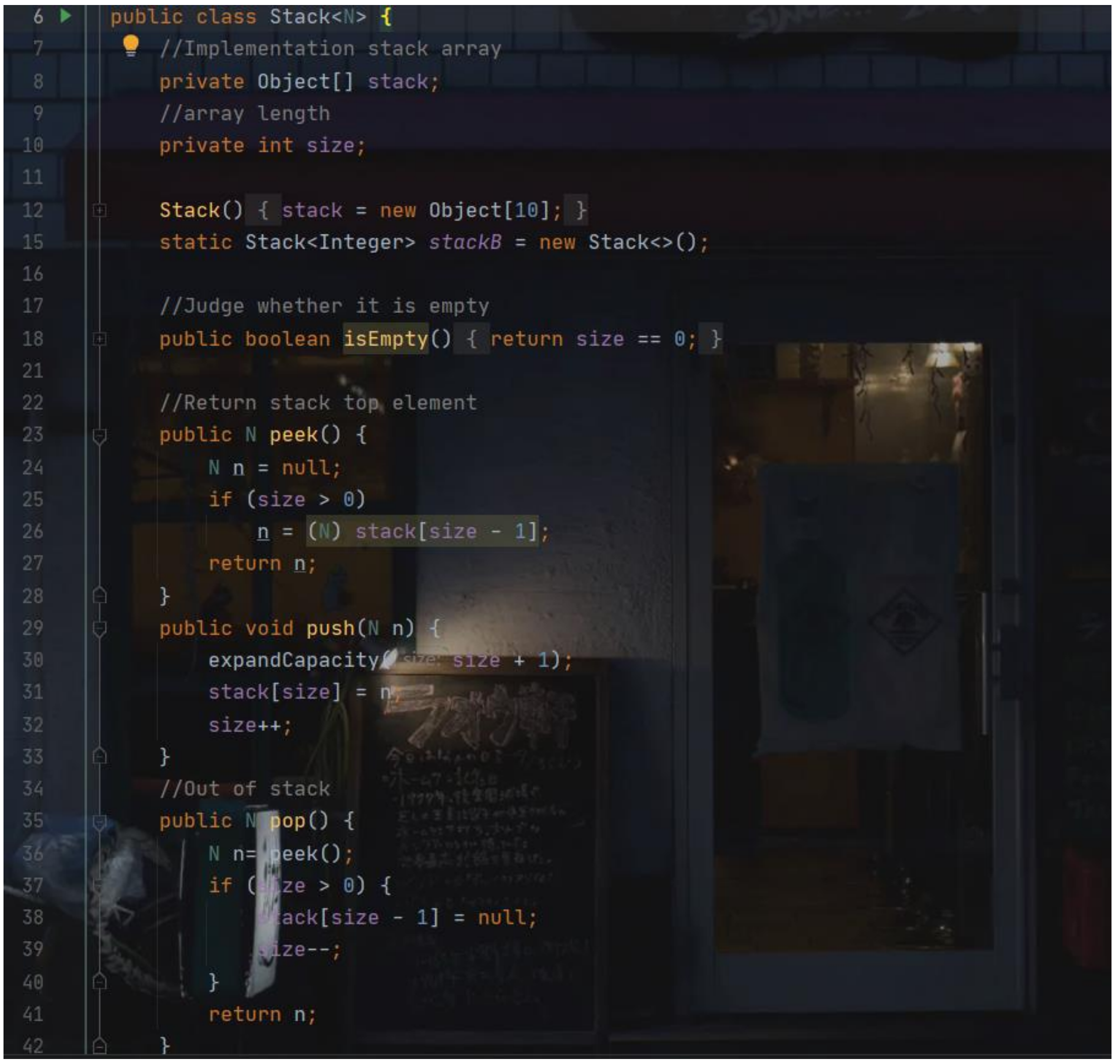
Q6: Create a stack class and add the functions below to it:

Test stack is:2 3 9 4 21 11

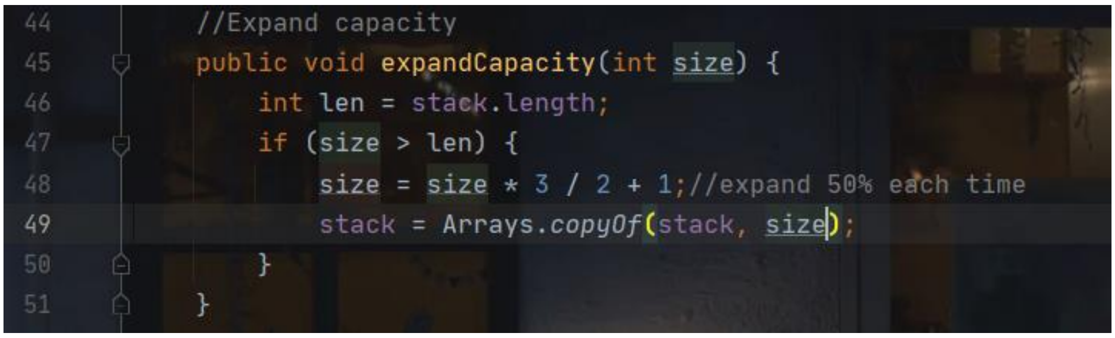
Main class:



```java
    public static void main(String[] args) {
        Stack<Integer> stack = new Stack<>();
        stack.push( n: 2);
        stack.push( n: 3);
        stack.push( n: 9);
        stack.push( n: 4);
        stack.push( n: 21);
        stack.push( n: 11);

        System.out.println("Input 1 to prints the elements in stack s from top to bottom");
        System.out.println("Input 2 to return a new stack whose elements are backwards from those in s");
        System.out.println("Input 3 to return a new stack whose elements are the same as those in s");
        System.out.println("Input others to Finish");

        Scanner in = new Scanner(System.in);
        int input = in.nextInt();
        switch(input){
            case 1:
                printStack(stack);
                break;
            case 2:
                reverseStack(stack);
                break;
            case 3:
                Scanner scanner = new Scanner(System.in);
                System.out.println("Please input a number");
                int a = scanner.nextInt();
                removeElement(stack,a);
                break;
            default:
                break;
        }
    }
```



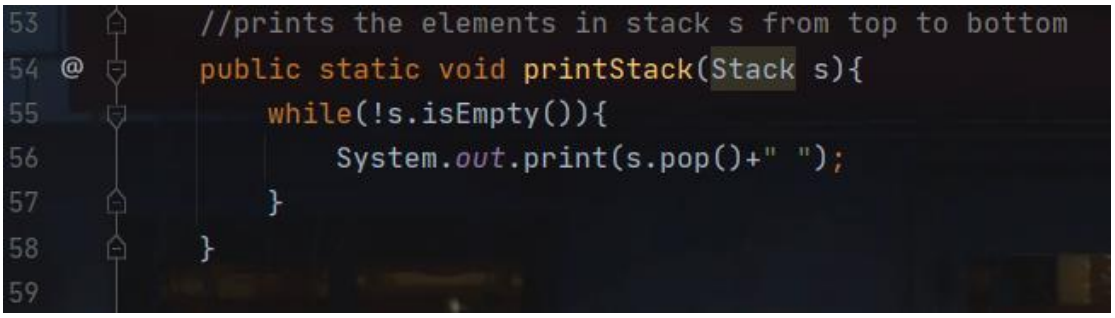
```java
public class Stack<N> {
    //Implementation stack array
    private Object[] stack;
    //array length
    private int size;

    Stack() { stack = new Object[10]; }
    static Stack<Integer> stackB = new Stack<>();

    //Judge whether it is empty
    public boolean isEmpty() { return size == 0; }

    //Return stack top element
    public N peek() {
        N n = null;
        if (size > 0)
            n = (N) stack[size - 1];
        return n;
    }
    public void push(N n) {
        expandCapacity( size: size + 1);
        stack[size] = n;
        size++;
    }
    //Out of stack
    public N pop() {
        N n= peek();
        if (size > 0) {
            stack[size - 1] = null;
            size--;
        }
        return n;
    }
```



```java
    //Expand capacity
    public void expandCapacity(int size) {
        int len = stack.length;
        if (size > len) {
            size = size * 3 / 2 + 1;//expand 50% each time
            stack = Arrays.copyOf(stack, size);
        }
    }
```

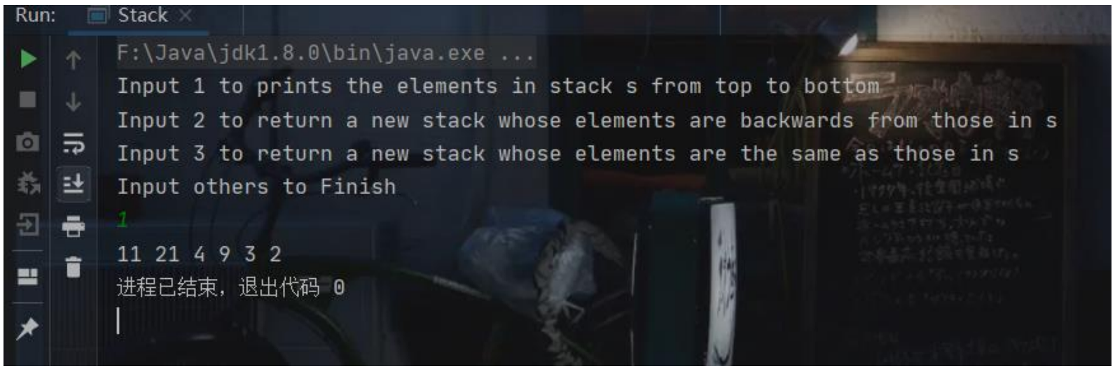
void *printStack*(Stack *s*)—prints the elements in stack *s* from top to bottom.

When printStack returns, *s* should be unchanged.



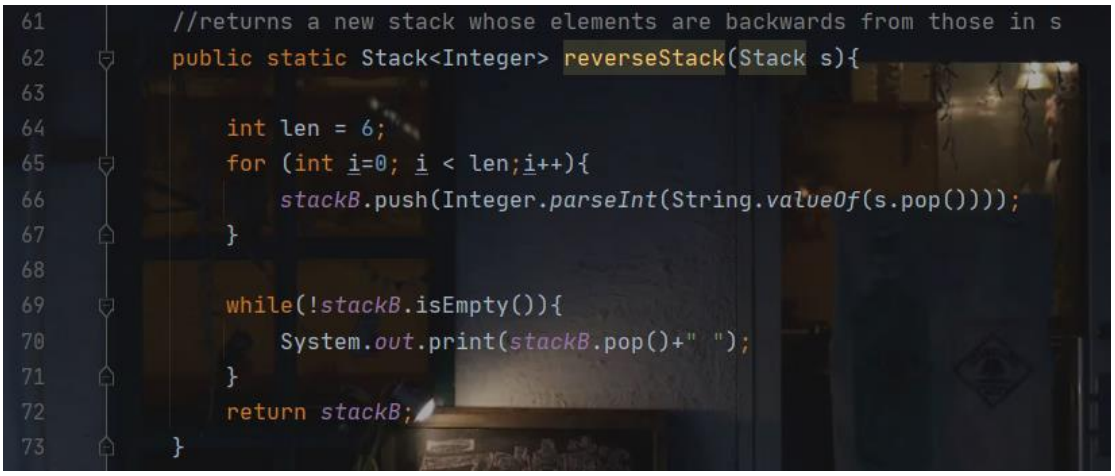
```java
    //prints the elements in stack s from top to bottom
    public static void printStack(Stack s){
        while(!s.isEmpty()){
            System.out.print(s.pop()+" ");
        }
    }

```

Running result:



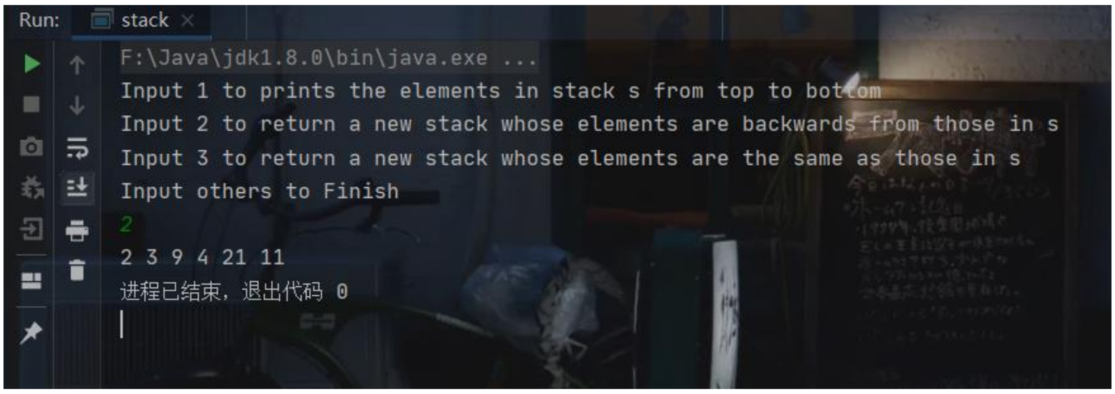
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to prints the elements in stack s from top to bottom
Input 2 to return a new stack whose elements are backwards from those in s
Input 3 to return a new stack whose elements are the same as those in s
Input others to Finish
1
11 21 4 9 3 2
进程已结束，退出代码 0
```

Stack *reverseStack*(Stack *s*)—returns a new stack whose elements are backwards from those in *s*. Again, *s* is unchanged.



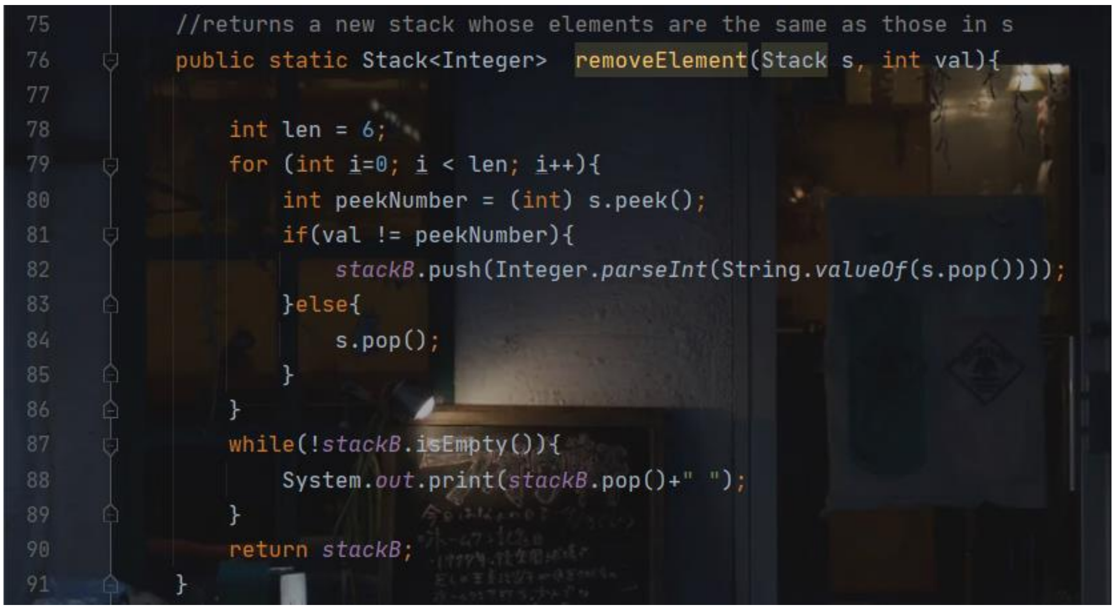
```java
    //returns a new stack whose elements are backwards from those in s
    public static Stack<Integer> reverseStack(Stack s){

        int len = 6;
        for (int i=0; i < len;i++){
            stackB.push(Integer.parseInt(String.valueOf(s.pop())));
        }

        while(!stackB.isEmpty()){
            System.out.print(stackB.pop()+" ");
        }
        return stackB;
    }
```

Running result:



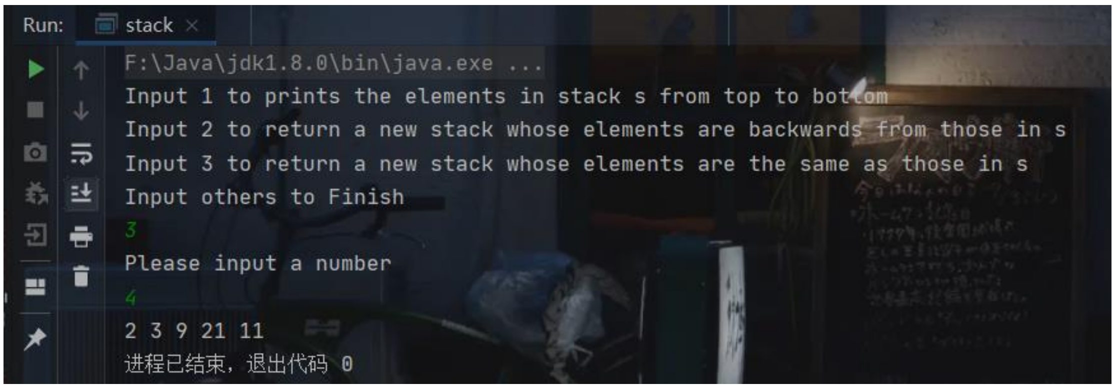
```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to prints the elements in stack s from top to bottom
Input 2 to return a new stack whose elements are backwards from those in s
Input 3 to return a new stack whose elements are the same as those in s
Input others to Finish
2
2 3 9 4 21 11
进程已结束，退出代码 0
```

Stack *removeElement*(Stack *s*, int *val*)—returns a new stack whose elements are the same as those in *s* (and in the same order) except that all occurrences of *val* have been removed. Again, *s* is unchanged.



```
    //returns a new stack whose elements are the same as those in s
    public static Stack<Integer>  removeElement(Stack s, int val){

        int len = 6;
        for (int i=0; i < len; i++){
            int peekNumber = (int) s.peek();
            if(val != peekNumber){
                stackB.push(Integer.parseInt(String.valueOf(s.pop())));
            }else{
                s.pop();
            }
        }
        while(!stackB.isEmpty()){
            System.out.print(stackB.pop()+" ");
        }
        return stackB;
    }
```

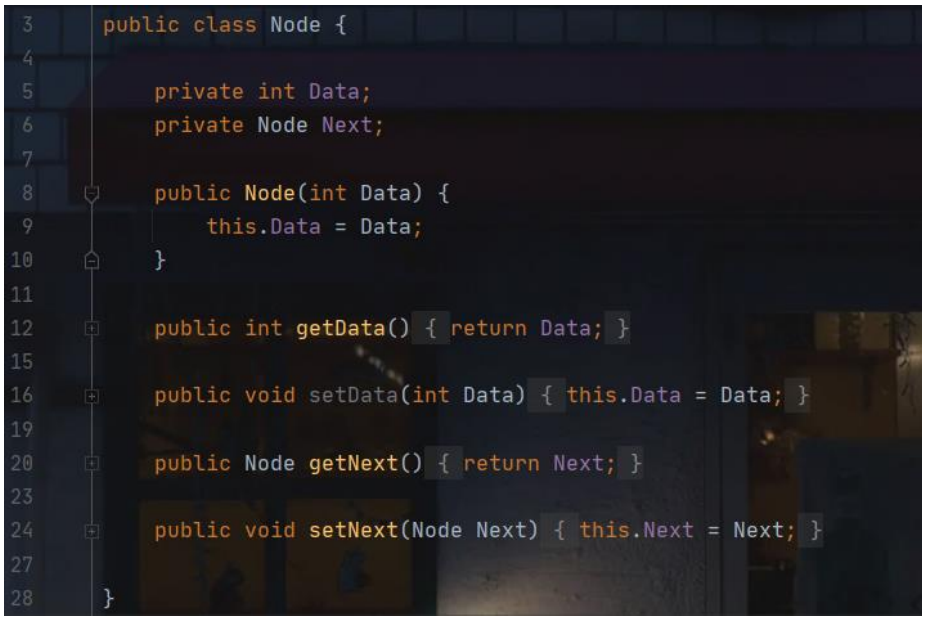
Running result:



```
F:\Java\jdk1.8.0\bin\java.exe ...
Input 1 to prints the elements in stack s from top to bottom
Input 2 to return a new stack whose elements are backwards from those in s
Input 3 to return a new stack whose elements are the same as those in s
Input others to Finish
3
Please input a number
4
2 3 9 21 11
进程已结束，退出代码 0
```

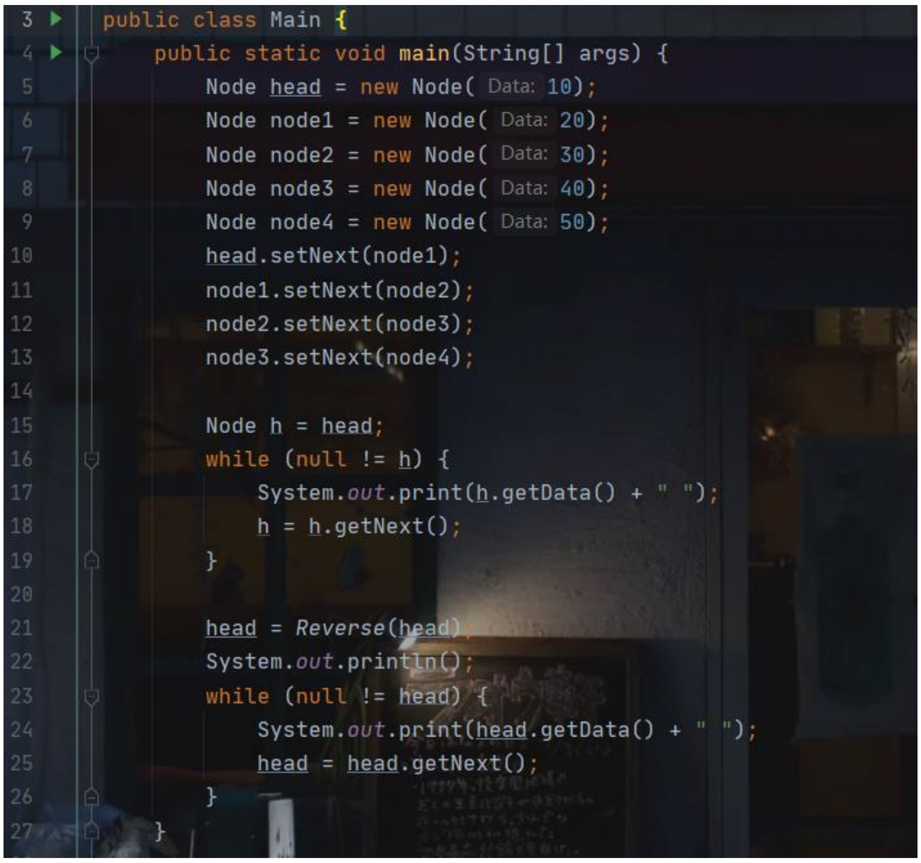
Q7: Print a linked list in reverse order. You are given a Singly linked list, print the linked list in reverse way, from end to start.

Node class:



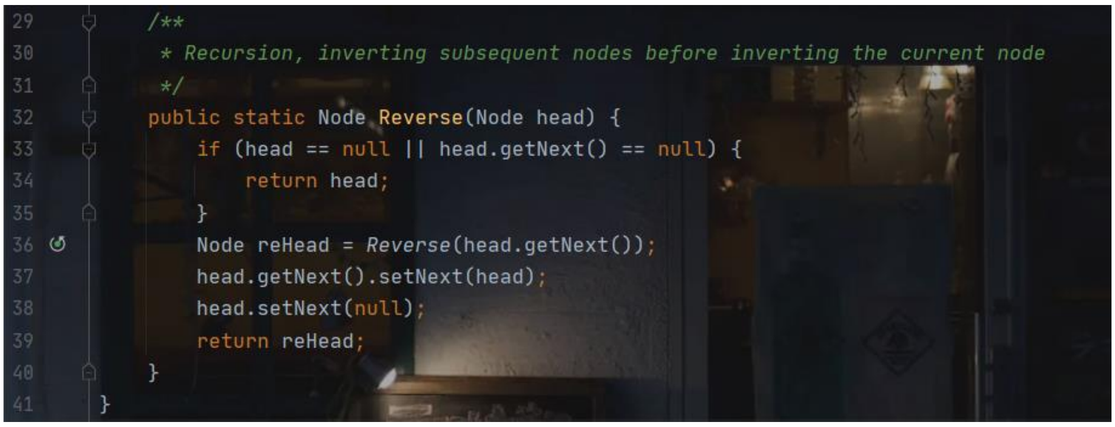
```java
public class Node {

    private int Data;
    private Node Next;

    public Node(int Data) {
        this.Data = Data;
    }

    public int getData() { return Data; }

    public void setData(int Data) { this.Data = Data; }

    public Node getNext() { return Next; }

    public void setNext(Node Next) { this.Next = Next; }

}
```

Main class:



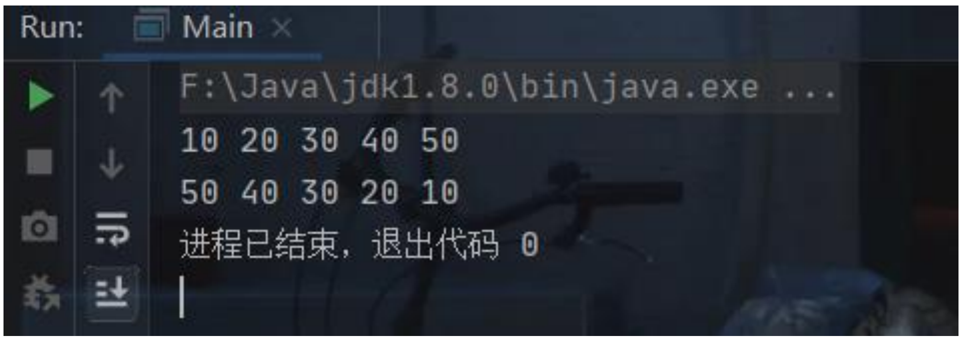
```java
public class Main {
    public static void main(String[] args) {
        Node head = new Node( Data: 10);
        Node node1 = new Node( Data: 20);
        Node node2 = new Node( Data: 30);
        Node node3 = new Node( Data: 40);
        Node node4 = new Node( Data: 50);
        head.setNext(node1);
        node1.setNext(node2);
        node2.setNext(node3);
        node3.setNext(node4);

        Node h = head;
        while (null != h) {
            System.out.print(h.getData() + " ");
            h = h.getNext();
        }

        head = Reverse(head);
        System.out.println();
        while (null != head) {
            System.out.print(head.getData() + " ");
            head = head.getNext();
        }
    }
```

Reverse node method



```java
    /**
     * Recursion, inverting subsequent nodes before inverting the current node
     */
    public static Node Reverse(Node head) {
        if (head == null || head.getNext() == null) {
            return head;
        }
        Node reHead = Reverse(head.getNext());
        head.getNext().setNext(head);
        head.setNext(null);
        return reHead;
    }
}
```

Running result:



```
F:\Java\jdk1.8.0\bin\java.exe ...
10 20 30 40 50
50 40 30 20 10
进程已结束，退出代码 0
```

Q8: Write a set of Java classes that can simulate an Internet application, where one party, Alice, is periodically creating a set of packets that she wants to send to Bob. An Internet process is continuously checking if Alice has any packets to send, and if so, it delivers them to Bob's computer, and Bob is periodically checking if his computer has a packet from Alice, and, if so, he reads and deletes it.

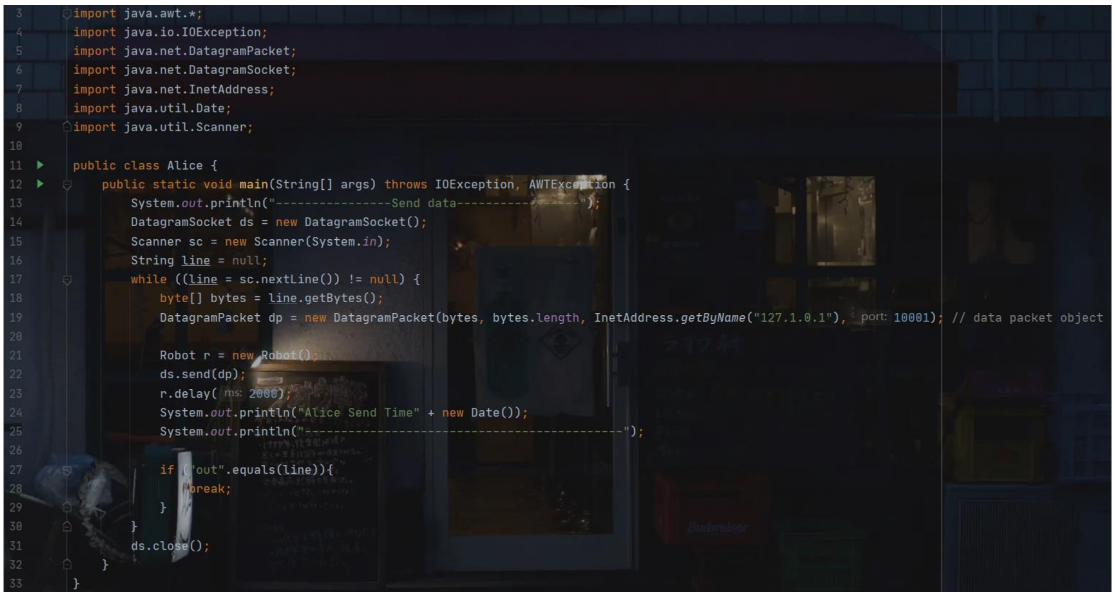
Alice class:

Use Robot object to add delay time(2000ms).

The sending port is 10001.

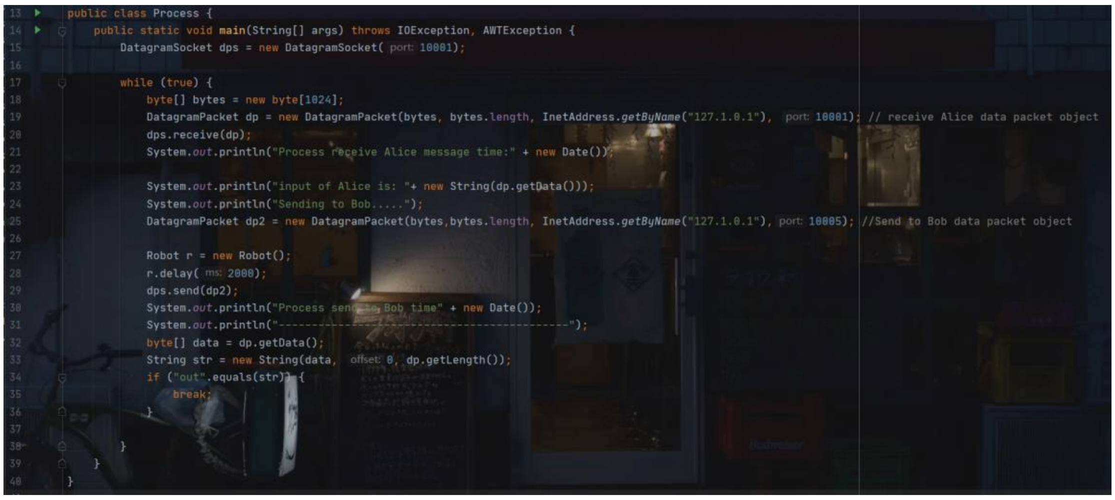
The receiving port is 10005.

Enter the out character to end.



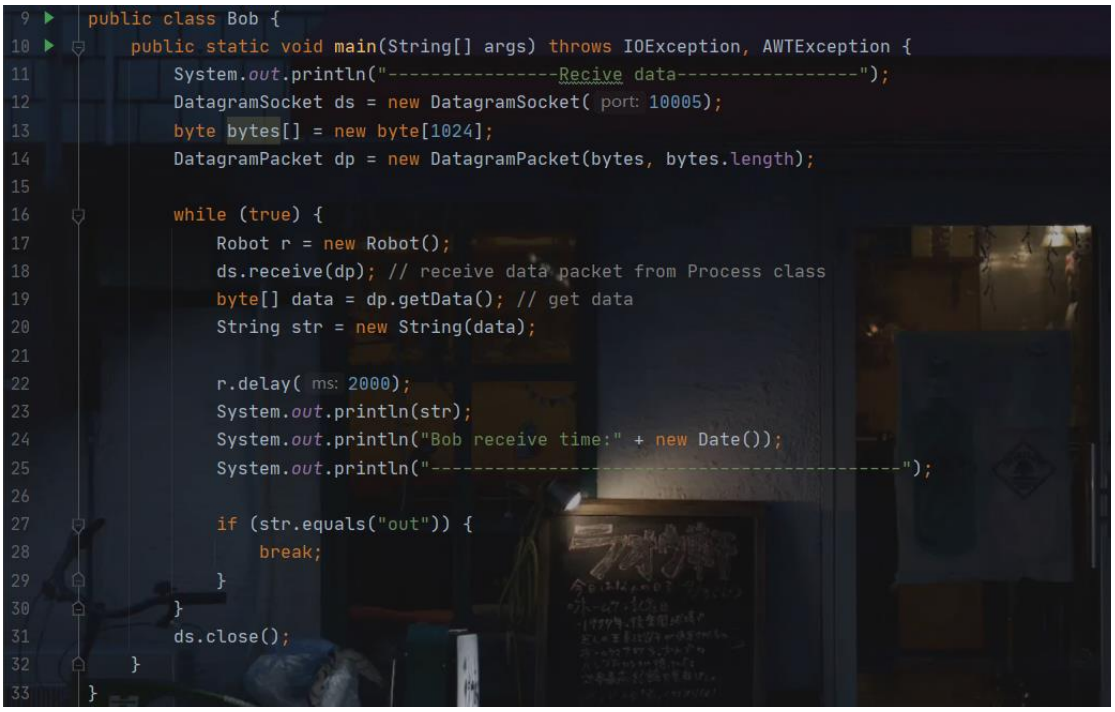
```java
import java.awt.*;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Date;
import java.util.Scanner;

public class Alice {
    public static void main(String[] args) throws IOException, AWTException {
        System.out.println("----------------Send data----------------");
        DatagramSocket ds = new DatagramSocket();
        Scanner sc = new Scanner(System.in);
        String line = null;
        while ((line = sc.nextLine()) != null) {
            byte[] bytes = line.getBytes();
            DatagramPacket dp = new DatagramPacket(bytes, bytes.length, InetAddress.getByName("127.1.0.1"),  port: 10001); // data packet object

            Robot r = new Robot();
            ds.send(dp);
            r.delay( ms: 2000);
            System.out.println("Alice Send Time" + new Date());
            System.out.println("---------------------------------------------");

            if ("out".equals(line)){
                break;
            }
        }
        ds.close();
    }
}
```

Process class:



```java
public class Process {
    public static void main(String[] args) throws IOException, AWTException {
        DatagramSocket dps = new DatagramSocket( port: 10001);

        while (true) {
            byte[] bytes = new byte[1024];
            DatagramPacket dp = new DatagramPacket(bytes, bytes.length, InetAddress.getByName("127.1.0.1"),  port: 10001); // receive Alice data packet object
            dps.receive(dp);
            System.out.println("Process receive Alice message time:" + new Date());

            System.out.println("input of Alice is: "+ new String(dp.getData()));
            System.out.println("Sending to Bob.....");
            DatagramPacket dp2 = new DatagramPacket(bytes,bytes.length, InetAddress.getByName("127.1.0.1"), port: 10005); //Send to Bob data packet object

            Robot r = new Robot();
            r.delay( ms: 2000);
            dps.send(dp2);
            System.out.println("Process send to Bob time" + new Date());
            System.out.println("--------------------------------------------");
            byte[] data = dp.getData();
            String str = new String(data,  offset: 0, dp.getLength());
            if ("out".equals(str)) {
                break;
            }

        }
    }
}
```

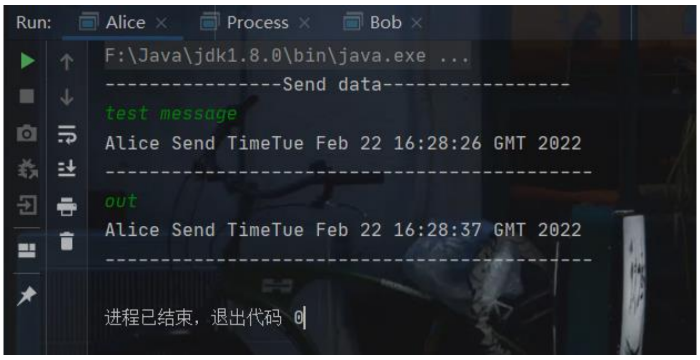
Bob class:



```java
public class Bob {
    public static void main(String[] args) throws IOException, AWTException {
        System.out.println("----------------Recive data-----------------");
        DatagramSocket ds = new DatagramSocket( port: 10005);
        byte bytes[] = new byte[1024];
        DatagramPacket dp = new DatagramPacket(bytes, bytes.length);

        while (true) {
            Robot r = new Robot();
            ds.receive(dp); // receive data packet from Process class
            byte[] data = dp.getData(); // get data
            String str = new String(data);

            r.delay( ms: 2000);
            System.out.println(str);
            System.out.println("Bob receive time:" + new Date());
            System.out.println("--------------------------------------------");

            if (str.equals("out")) {
                break;
            }
        }
        ds.close();
    }
}
```

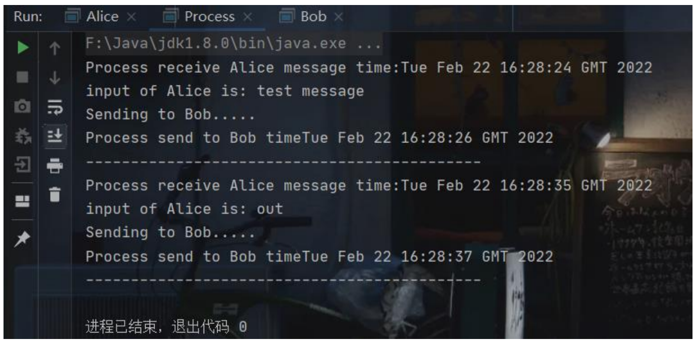
Running result:

Alice send message



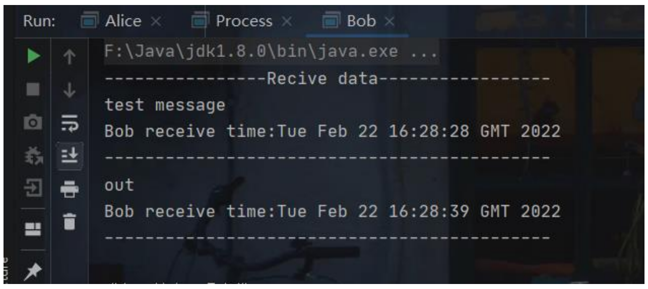
```
F:\Java\jdk1.8.0\bin\java.exe ...
----------------Send data-----------------
test message
Alice Send TimeTue Feb 22 16:28:26 GMT 2022
--------------------------------------------
out
Alice Send TimeTue Feb 22 16:28:37 GMT 2022
--------------------------------------------

进程已结束，退出代码 0
```

Process receive message and send to Bob



```
F:\Java\jdk1.8.0\bin\java.exe ...
Process receive Alice message time:Tue Feb 22 16:28:24 GMT 2022
input of Alice is: test message
Sending to Bob.....
Process send to Bob timeTue Feb 22 16:28:26 GMT 2022
--------------------------------------------
Process receive Alice message time:Tue Feb 22 16:28:35 GMT 2022
input of Alice is: out
Sending to Bob.....
Process send to Bob timeTue Feb 22 16:28:37 GMT 2022
--------------------------------------------

进程已结束，退出代码 0
```

Bob receive message



```
F:\Java\jdk1.8.0\bin\java.exe ...
----------------Recive data-----------------
test message
Bob receive time:Tue Feb 22 16:28:28 GMT 2022
--------------------------------------------
out
Bob receive time:Tue Feb 22 16:28:39 GMT 2022
--------------------------------------------
```