D:

一、

A,C,A,D,D

二、

continue

interface

单

三:

1.什么是对象？什么是类？类和对象有什么关系？

- **类**是对一组具有相同属性和方法的对象的抽象描述。它是一种模板或蓝图，定义了对象的特征（属性）和行为（方法）。
- **对象**是类的实例，是根据类的定义创建的具体实体。每个对象都有自己的状态和行为。
- **关系**：类是对象的模板，而对象是类的具体实例。一个类可以创建多个对象，而每个对象都是类的实例，具有类定义的属性和方法。

2. 请简述this关键字的应用。

this关键字用于引用当前对象的实例。在以下情况下常用：

- **区分成员变量和参数**：在构造函数或方法中，如果参数名与成员变量相同，可以使用`this`关键字来区分。
- **调用其他构造函数**：在构造函数内部，可以使用`this()`调用同一个类的其他构造函数。
- **传递当前对象**：可以将当前对象作为参数传递给其他方法或构造函数。

3. 请简述String类和StringBuffer类的区别。

- **不可变性**：
  - `String`是不可变的，创建后无法更改内容，每次操作都会返回一个新的字符串对象。
  - `StringBuffer`是可变的，可以在原有基础上修改内容而不生成新对象。
- **性能**：
  - 因为`String`每次修改都创建新对象，所以在频繁修改字符串时性能较差。
  - `StringBuffer`在需要频繁修改字符串的情况下性能更优，因为它直接修改现有对象。
- **线程安全**：
  - `StringBuffer`是线程安全的，适合在多线程环境中使用。
  - `String`则是线程安全的，但其不可变性使得它本身无需考虑并发问题。

四:

1.

```
public class PerfectSquare {
    public static void main(String[] args) {
        for (int num = 1; num < 100000; num++) {
            if (isPerfectSquare(num + 100) && isPerfectSquare(num + 268)) {
                System.out.println("该正整数是: " + num);
                break;
            }
```

```
        }
    }

    private static boolean isPerfectSquare(int n) {
        int sqrt = (int) Math.sqrt(n);
        return n == sqrt * sqrt;
    }
}
```

2.

```
public class GCDAndLCM {
    public static void main(String[] args) {
        int a = 56;
        int b = 98;

        int gcd = findGCD(a, b);
        int lcm = findLCM(a, b, gcd);

        System.out.println("最大公因数: " + gcd);
        System.out.println("最小公倍数: " + lcm);
    }

    private static int findGCD(int a, int b) {
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }

    private static int findLCM(int a, int b, int gcd) {
        return (a * b) / gcd;
    }
}
```

3.

```
import java.util.ArrayList;
import java.util.Scanner;

class Student {
    private String name;
    private String studentId;
    private ArrayList<Course> courses;

    public Student(String name, String studentId) {
        this.name = name;
        this.studentId = studentId;
        this.courses = new ArrayList<>();
    }

    public String getName() {
```

```
        return name;
    }

    public String getStudentId() {
        return studentId;
    }

    public void addCourse(Course course) {
        courses.add(course);
    }

    public ArrayList<Course> getCourses() {
        return courses;
    }
}

class Course {
    private String courseName;
    private String teacherName;

    public Course(String courseName, String teacherName) {
        this.courseName = courseName;
        this.teacherName = teacherName;
    }

    public String getCourseName() {
        return courseName;
    }

    public String getTeacherName() {
        return teacherName;
    }
}

public class CourseSelectionSystem {
    private static ArrayList<Student> students = new ArrayList<>();

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int choice;

        do {
            System.out.println("选课系统:");
            System.out.println("1. 添加学生");
            System.out.println("2. 添加课程");
            System.out.println("3. 查询学生信息");
            System.out.println("4. 退出");
            System.out.print("请输入选择: ");
            choice = scanner.nextInt();
            scanner.nextLine(); // 清除缓冲区

            switch (choice) {
                case 1:
                    addStudent(scanner);
                    break;
                case 2:
                    addCourse(scanner);
                    break;
```

```
                case 3:
                    queryStudentInfo(scanner);
                    break;
                case 4:
                    System.out.println("退出系统");
                    break;
                default:
                    System.out.println("无效选择，请重试。");
            }
        } while (choice != 4);
    }

    private static void addStudent(Scanner scanner) {
        System.out.print("请输入学生姓名: ");
        String name = scanner.nextLine();
        System.out.print("请输入学号: ");
        String studentId = scanner.nextLine();
        students.add(new Student(name, studentId));
        System.out.println("学生添加成功!");
    }

    private static void addCourse(Scanner scanner) {
        System.out.print("请输入学生姓名: ");
        String name = scanner.nextLine();
        Student student = findStudentByName(name);
        if (student != null) {
            System.out.print("请输入课程名称: ");
            String courseName = scanner.nextLine();
            System.out.print("请输入任课老师: ");
            String teacherName = scanner.nextLine();
            student.addCourse(new Course(courseName, teacherName));
            System.out.println("课程添加成功!");
        } else {
            System.out.println("未找到学生！");
        }
    }

    private static void queryStudentInfo(Scanner scanner) {
        System.out.print("请输入学生姓名: ");
        String name = scanner.nextLine();
        Student student = findStudentByName(name);
        if (student != null) {
            System.out.println("学号: " + student.getStudentId());
            System.out.println("选课信息:");
            for (Course course : student.getCourses()) {
                System.out.println("课程: " + course.getCourseName() + ", 任课老
师: " + course.getTeacherName());
            }
        } else {
            System.out.println("未找到学生！");
        }
    }

    private static Student findStudentByName(String name) {
        for (Student student : students) {
            if (student.getName().equalsIgnoreCase(name)) {
                return student;
            }
```

```
        }
        return null;
    }
}
```