B：

一、

B,C,A,D,A

二、

//，

MyClass.java,

Try

三：简答题

1. **简述break、continue和return语句的区别。**
   - **break**: 用于结束当前循环或switch语句，跳出循环体。例如，在一个for循环中使用break将导致循环立即终止，程序控制流转向循环之后的代码。
   - **continue**: 用于跳过当前循环中的剩余代码，并立即开始下一次循环迭代。在for或while循环中使用continue时，控制流会跳到循环的条件判断部分。
   - **return**: 用于结束函数的执行并返回一个值（如果有）。当函数执行到return语句时，函数退出，返回到调用该函数的位置。
2. **简述IO流的分类有哪些?**
   - **按数据流动方向分类**:
     - 输入流: 用于读取数据的流，例如`FileInputStream`。
     - 输出流: 用于写入数据的流，例如`FileOutputStream`。
   - **按数据处理方式分类**:
     - 字节流: 以字节为单位读取和写入数据，如`InputStream`和`OutputStream`。
     - 字符流: 以字符为单位读取和写入数据，如`Reader`和`Writer`。
   - **按功能分类**:
     - 缓冲流: 提供缓冲功能的流，以提高效率，如`BufferedInputStream`和`BufferedOutputStream`。
     - 数据流: 处理基本数据类型的流，如`DataInputStream`和`DataOutputStream`。
     - 对象流: 用于对象的序列化和反序列化，如`ObjectInputStream`和`ObjectOutputStream`。

四：

1.

```
public class NarcissisticNumbers {
    public static void main(String[] args) {
        for (int number = 0; number < 1000; number++) {
            int sum = 0;
            int temp = number;
            int digits = String.valueOf(number).length();

            while (temp > 0) {
                int digit = temp % 10;
                sum += Math.pow(digit, digits);
                temp /= 10;
            }
```

```
            if (sum == number) {
                System.out.println(number);
            }
        }
    }
}
```

2.

```
public class ChickenRabbit {
    public static void main(String[] args) {
        int x = 35; // 鸡兔总数
        int y = 94; // 脚总数

        int chickens = (4 * x - y) / 2; // 鸡的数量
        int rabbits = x - chickens;      // 兔的数量

        System.out.println("鸡的数量: " + chickens);
        System.out.println("兔的数量: " + rabbits);
    }
}
```

3.

```
public class DiamondShape {
    public static void main(String[] args) {
        int n = 5; // 行数可以根据需要进行设置

        // 打印上半部分
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                System.out.print(" ");
            }
            for (int j = 0; j < 2 * i + 1; j++) {
                System.out.print("*");
            }
            System.out.println();
        }

        // 打印下半部分
        for (int i = n - 2; i >= 0; i--) {
            for (int j = 0; j < n - i - 1; j++) {
                System.out.print(" ");
            }
            for (int j = 0; j < 2 * i + 1; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

4,

```
import java.util.ArrayList;

class Student {
    public String studentId; // 公有属性
    public String name;       // 公有属性
    private int javaScore;    // 私有属性

    public Student(String studentId, String name, int javaScore) {
        this.studentId = studentId;
        this.name = name;
        this.javaScore = javaScore;
    }

    public int getJavaScore() {
        return javaScore;
    }
}

public class StudentInfo {
    public static void main(String[] args) {
        ArrayList<Student> students = new ArrayList<>();
        students.add(new Student("001", "Alice", 85));
        students.add(new Student("002", "Bob", 90));
        students.add(new Student("003", "Charlie", 75));
        students.add(new Student("004", "David", 95));
        students.add(new Student("005", "Eva", 80));

        int totalScore = 0;
        int maxScore = Integer.MIN_VALUE;
        int minScore = Integer.MAX_VALUE;

        for (Student student : students) {
            System.out.printf("学号: %s, 姓名: %s, Java成绩: %d%n",
                    student.studentId, student.name, student.getJavaScore());
            int score = student.getJavaScore();
            totalScore += score;
            maxScore = Math.max(maxScore, score);
            minScore = Math.min(minScore, score);
        }

        double averageScore = (double) totalScore / students.size();
        System.out.printf("Java语言成绩的平均值: %.2f%n", averageScore);
        System.out.printf("Java语言成绩的最大值: %d%n", maxScore);
        System.out.printf("Java语言成绩的最小值: %d%n", minScore);
    }
}
```