중간고사

1. 아래는 Problem1.java 파일이다.

Problem1.java

```
 1 import java.util.ArrayList;
 2 import java.util.List;
 3 import java.util.Scanner;
 4
 5 public class Problem1 {
 6     public static void main(String[] args) {
 7         Scanner scanner = new Scanner(System.in);
 8         int n = scanner.nextInt();
 9         scanner.close();
10
11         Pattern pattern = new Pattern(              );
12         pattern.print();
13     }
14 }
15
16 class Pattern {
17     List<Line> lines = new ArrayList<>();
18
19     public Pattern(LineFactory factory) {
20         lines.addAll(factory.createLines());
21     }
22
23     public void print() {
24         for (Line line : lines) {
25             line.print();
26         }
27     }
28 }
29
30 interface Line {
31     void print();
32 }
33
34 interface LineFactory {
35     List<Line> createLines();
36 }
```

(1) 중간고사 1-1번 문제를 위한 Problem1.java에는 11행의 적색 빈칸에 new Line11(n)이 입력되어 있다. Problem1.java의 main 메소드를 실행하여 아래의 Input에 대해 Output과 같은 패턴을 출력하는 **Line11.java** 파일을 작성하여 제출하라. (5점)

Input	**Output**
1	x
2	-x xx
5	----x ---xx --xxx -xxxx xxxxx

(2) 중간고사 1-2번 문제를 위한 Problem1.java에는 11행의 적색 빈칸에 new Line12(n)이 입력되어 있다. Problem1.java의 main 메소드를 실행하여 아래의 Input에 대해 Output과 같은 패턴을 출력하는 **Line12.java** 파일을 작성하여 제출하라. (15점)

Input	Output
1	x
2	x-x- -x-x
3	x-x-x- -x-x-x x-x-x-
5	x-x-x-x-x- -x-x-x-x-x x-x-x-x-x- -x-x-x-x-x x-x-x-x-x-

2. 최장 경로 하노이탑 (30점)

세 개의 장대가 있고 첫 번째 장대에는 반경이 서로 다른 n개의 원판이 쌓여 있다. 각 원판은 반경이 큰 순서대로 쌓여있다. 이제 수도승들이 다음 규칙에 따라 첫 번째 장대에서 세 번째 장대로 옮기려 한다.

- 한 번에 한 개의 원판만을 다른 탑으로 옮길 수 있다.
- 쌓아 놓은 원판은 항상 위의 것이 아래의 것보다 작아야 한다.
- **원판은 인접한 기둥으로만 옮길 수 있다.**

단, 이동 횟수는 최소가 되어야 한다.

예를 들어, 만약 원판이 1개 라면,

- 1번 원판을 A에서 B로 옮긴다.
- 1번 원판을 B에서 C로 옮긴다.

따라서 총 2번을 옮기면 모두 옮길 수 있다.

2.1. 입력

첫째 줄에 첫 번째 장대에 쌓인 원판의 개수 N(1 ≤ N ≤ 10)가 주어진다.

2.2. 출력

총 옮긴 수를 출력한다.

Input	**Output**
1	2
2	8
3	26
5	242
10	59048

2.3. 제약사항

아래 Problem2.java 파일의 main 메소드를 실행하여 예시 Input에 대해 예시 Output을 출력하는 **MultiDiscsTower.java** 파일을 작성하고 제출하라.

Problem2.java

```
import java.util.Scanner;

class Problem2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n= scanner.nextInt();

        Tower tower = new MultiDiscsTower(n);
        System.out.println(tower.move());
    }
}

interface Tower {
    int move();
}
```

3. 별 찍기 (40점)

예제를 보고 규칙을 유추한 뒤에 'x'와 '-'를 찍어 보세요.

- 입력: 첫째 줄에 N(1 ≤ N ≤ 10)이 주어진다.
- 출력: 첫째 줄부터 차례대로 'x'와 '-'을 출력한다.

Input	Output
1	x
2	xxxxx -xxx- --x--
3	------x------ -----x-x----- ----x---x---- ---xxxxxxx--- --x--xxx--x-- -x----x----x- xxxxxxxxxxxxx
4	xxxxxxxxxxxxxxxxxxxxxxxxxxxxx -x------------x------------x- --x----------x-x----------x-- ---x--------x---x--------x--- ----x------xxxxxxx------x---- -----x----x--xxx--x----x----- ------x--x----x----x--x------ -------xxxxxxxxxxxxxxx------- --------x-----------x-------- ---------x---------x--------- ----------x-------x---------- -----------x-----x----------- ------------x---x------------ -------------x-x------------- --------------x--------------

- 제약사항: 아래 Problem3.java 파일의 main 메소드를 실행하여 예시 Input에 대해 예시 Output과 같은 패턴을 출력하는 **Triangle.java** 파일을 작성하고 제출하라.

Problem3.java

```
import java.util.Scanner;

final class Problem3 {
	private Triangle triangle;

	public Problem3(int n) {
		// row 와 col 은 보드의 좌측 상단점이 된다.
		int row = 0, col = 0;
		if (n == 1) triangle = new OneStarTriangle(row, col);
		else triangle = new CompositeTriangle(n, row, col);
	}

	public void solve(Board board) {
		triangle.write(board);
	}

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		int n = sc.nextInt();
		sc.close();

		Board board = new Board(n);
		Problem3 p = new Problem3(n);
		p.solve(board);
		board.print();
	}
}

class Board {
	char[][] board;

	public Board(int n) {
		int row = (int) (Math.pow(2, n) -  1);
		int col = (int) (Math.pow(2, n + 1) - 3);
		board = new char[row][col];

		for (int i = 0; i < row; i++) {
			for (int j = 0; j < col; j++) {
				board[i][j] = '-';
			}
		}
	}

	public void write(int row, int col) {
		board[row][col] = 'x';
	}

	public void print() {
		StringBuilder sb = new StringBuilder();

		for (char[] row : board) {
			for (char c : row)
				sb.append(c);
			sb.append("\n");
		}
		System.out.print(sb.toString());
	}
}
```

4. Report Generators (10점)

Problem4.java 파일의 main 메소드를 실행할 수 있도록 **SimpleReportGenerator.java** 파일을 작성하고 제출하라.

Problem4.java

```
import java.util.ArrayList;
import java.util.List;

public class Problem4 {
	public static void main(String[] args) {
		List<Customer> customers = new ArrayList<>();

		customers.add(new Customer("홍길동", 150));
		customers.add(new Customer("우수한", 350));
		customers.add(new Customer("부족한", 50));
		customers.add(new Customer("훌륭한", 450));
		customers.add(new Customer("최고의", 550));

		AbstractReportGenerator simpleGenerator = new SimpleReportGenerator();
		System.out.println(simpleGenerator.generate(customers));

		AbstractReportGenerator complexGenerator = new ComplexReportGenerator();
		System.out.println(complexGenerator.generate(customers));
	}
}

class Customer {
	private final String name;
	private int point;

	public Customer(String name, int point) {
		this.name = name;
		this.point = point;
	}

	public final int getPoint() { return point;}
	public final String getName() { return name; }
}

abstract class AbstractReportGenerator {
	public final String generate(List<Customer> customers) {
		List<Customer> selectedCustomers = select(customers);
		String report = getReportHeader(selectedCustomers);
		for (final Customer customer : selectedCustomers)
			report += getReportForCustomer(customer);
		report += getReportFooter(selectedCustomers);
		return report;
	}

	protected List<Customer> select(List<Customer> customers) {
		List<Customer> selected = new ArrayList<>();
		for (final Customer customer : customers)
			if (customerReportCondition(customer))
```

```
                selected.add(customer);
            return selected;
        }

        protected abstract boolean customerReportCondition(Customer customer);
        protected abstract String getReportHeader(List<Customer> customers);
        protected abstract String getReportForCustomer(Customer customer);
        protected abstract String getReportFooter(List<Customer> customers);
}
```

- Input과 Output은 아래와 같다.

Input	**Output**
	고객의 수: 5 명 홍길동: 150 우수한: 350 부족한: 50 훌륭한: 450 최고의: 550 고객의 수: 4 명입니다 150: 홍길동 350: 우수한 450: 훌륭한 550: 최고의 점수 합계: 1500