

자바 코딩테스트

JAVA 코딩테스트 답안

류영표 강사

ryp1662@gmail.com

Copyright © "Youngpyo Ryu" All Rights Reserved.

This document was created for the exclusive use of "Youngpyo Ryu".

It must not be passed on to third parties except with the explicit prior consent of "Youngpyo Ryu".

시간복잡도 계산하기

```
public class main {  
    public static void main(String[] args) {  
        for(int i=1;i<5;i++){  
            for(int j=0;j<i;j++){  
                System.out.print("*");  
            }  
            System.out.println("");  
        }  
    }  
}
```

```
public class main {  
    public static void main(String[] args) {  
        for(int i=0;i<4;i++){  
            for(int j=0;j<3-i;j++){  
                System.out.print(" ");  
            }  
            for(int j=0;j<2*i+1;j++){  
                System.out.print("*");  
            }  
            System.out.println("");  
        }  
    }  
}
```

세균 증식

```
class Solution {
    public int solution(int n, int t) {
        // n = 세균의 수, t = 증식하는 과정에서 경과된 시간

        int answer = n;
        // 초기 세균의 수 n을 answer 변수에 할당하는 코드이다.
        // 이를 통해 반복문에서 세균의 수를 계산할 때마다 answer 변수를 사용하여 값을 갱신한다.

        for (int i = 0; i < t; i++) {
            // t보다 작을 때 마다 i를 0부터 1씩 증가시켰다.
            // t 시간까지만 증식이 되기 때문.

            answer *= 2;
            // answer = n이고 1시간에 2배 증식하기 때문에 answer *= 2를 해줌.
        }
        return answer;
    }
}
```

```
public class main {
    public static void main(String[] args) {
        Solution solution = new Solution();
        int initialBacteria = 2; // 초기 세균의 수
        int hoursElapsed = 3;    // 경과된 시간 (시간 단위)

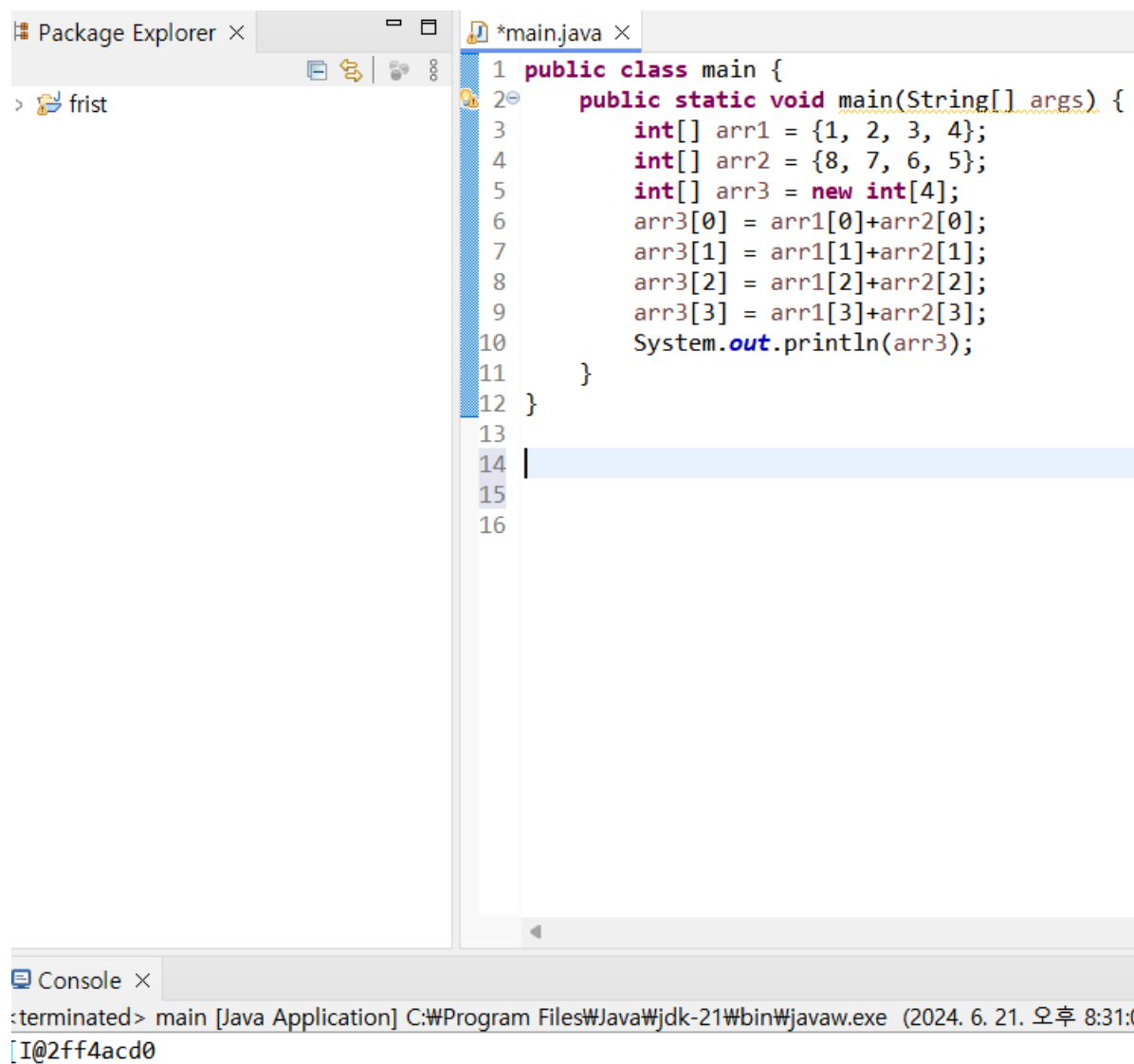
        int result = solution.solution(initialBacteria, hoursElapsed);
        System.out.println("경과된 시간 후 세균의 수: " + result);
    }
}

class Solution {
    public int solution(int n, int t) {
        // n: 초기 세균의 수
        // t: 경과된 시간 (시간 단위)

        int answer = n;
        // 초기 세균의 수 n을 answer 변수에 할당합니다.

        for (int i = 0; i < t; i++) {
            // t 시간 동안 반복합니다.
            answer *= 2;
            // 세균의 수를 2배로 증가시킵니다.
        }

        return answer;
    }
}
```



Package Explorer ×

*main.java ×

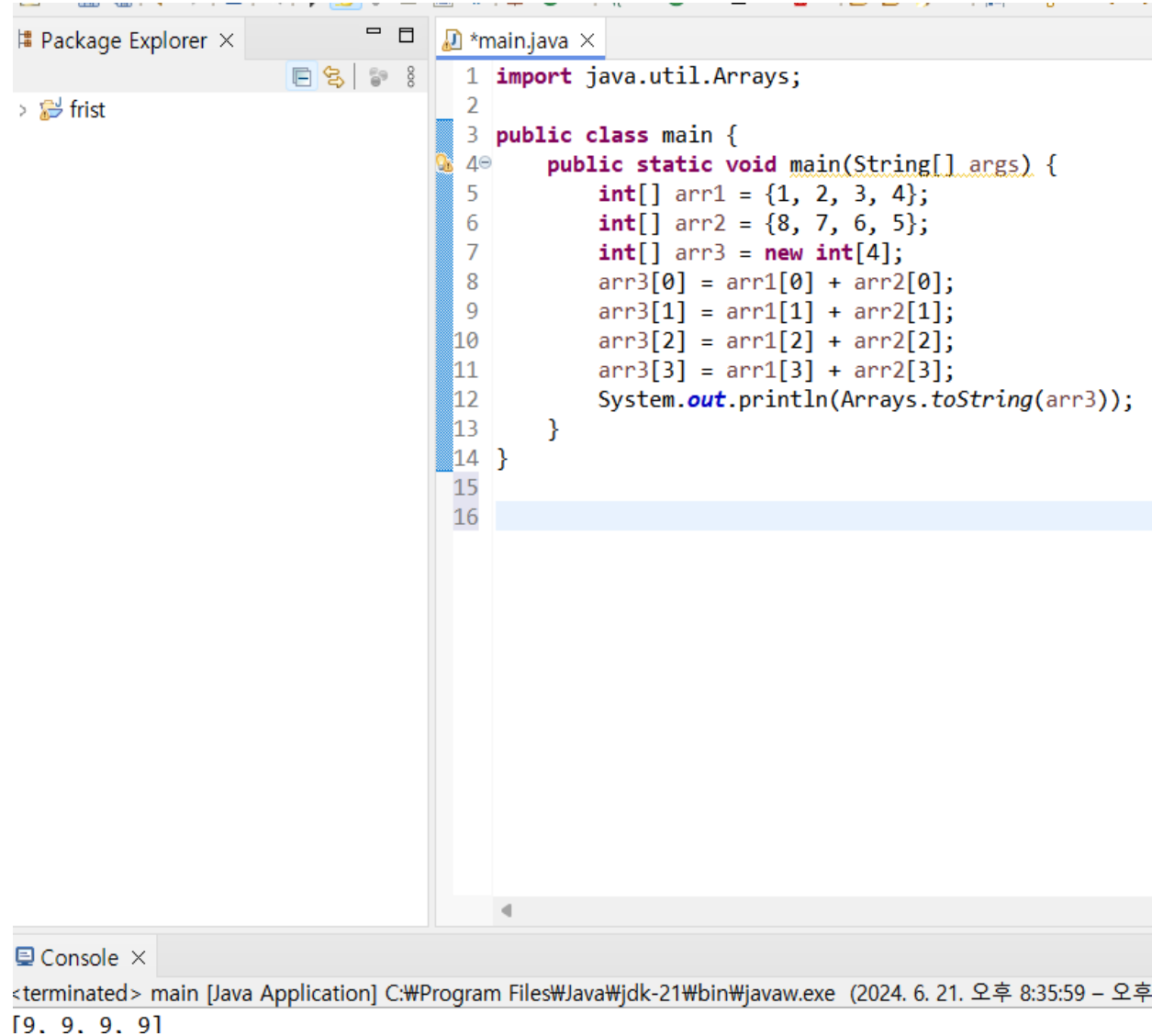
```

1 public class main {
2     public static void main(String[] args) {
3         int[] arr1 = {1, 2, 3, 4};
4         int[] arr2 = {8, 7, 6, 5};
5         int[] arr3 = new int[4];
6         arr3[0] = arr1[0]+arr2[0];
7         arr3[1] = arr1[1]+arr2[1];
8         arr3[2] = arr1[2]+arr2[2];
9         arr3[3] = arr1[3]+arr2[3];
10        System.out.println(arr3);
11    }
12 }
13
14
15
16

```

Console ×

<terminated> main [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (2024. 6. 21. 오후 8:31:00) [I@2ff4acd0]



Package Explorer ×

*main.java ×

```

1 import java.util.Arrays;
2
3 public class main {
4     public static void main(String[] args) {
5         int[] arr1 = {1, 2, 3, 4};
6         int[] arr2 = {8, 7, 6, 5};
7         int[] arr3 = new int[4];
8         arr3[0] = arr1[0] + arr2[0];
9         arr3[1] = arr1[1] + arr2[1];
10        arr3[2] = arr1[2] + arr2[2];
11        arr3[3] = arr1[3] + arr2[3];
12        System.out.println(Arrays.toString(arr3));
13    }
14 }
15
16

```

Console ×

<terminated> main [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (2024. 6. 21. 오후 8:35:59 - 오후 8:36:00) [9. 9. 9. 9]

문제 01) 배열 정리하기

```
import java.util.Arrays;

public class Solution {

    public static void main(String[] args) {
        System.out.println(Arrays.toString(solution(new int[]{1, -5, 2, 4, 3})));
        System.out.println(Arrays.toString(solution(new int[]{2, 1, 1, 3, 2, 5, 4})));
        System.out.println(Arrays.toString(solution(new int[]{6, 1, 7})));
    }

    // 이 부분을 변경해서 실행해보세요.
    private static int[] solution(int[] arr) {
        Arrays.sort(arr);
        return arr;
    }
}
```

```
import java.util.Arrays;
import java.util.Random;

class Solution {
    public static void main(String[] args) {
        // 배열을 랜덤 값으로 초기화
        int[] arr = new int[100000];
        Random random = new Random();
        for (int i = 0; i < arr.length; i++) {
            arr[i] = random.nextInt(100000); // 0부터 99999까지의 랜덤 값
        }

        // 버블 정렬 시간 측정
        long start = System.currentTimeMillis();
        int[] bubble = bubbleSort(arr);
        long end = System.currentTimeMillis();
        System.out.println("Bubble sort time: " + (end - start) / 1000.0 + "초");

        // 배열을 다시 초기화하여 동일한 값으로 정렬 수행
        for (int i = 0; i < arr.length; i++) {
            arr[i] = random.nextInt(100000);
        }

        // 내장 정렬 시간 측정
        start = System.currentTimeMillis();
        int[] sort = doSort(arr);
        end = System.currentTimeMillis();
        System.out.println("API sort time: " + (end - start) / 1000.0 + "초");

        // 두 배열이 같은지 확인
        System.out.println("Arrays are equal: " + Arrays.equals(bubble, sort));
    }
}
```

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int N;
        N = sc.nextInt();

        int[] array = new int[N];

        for(int i=0; i<N; i++) {
            array[i] = sc.nextInt();
        }

        Arrays.sort(array);

        for(int value : array) {
            System.out.println(value);
        }
    }
}
```

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        int N = in.nextInt();
        int[] arr = new int[N];

        for(int i = 0; i < N; i++) {
            arr[i] = in.nextInt();
        }

        // Selection sort
        for(int i = 0; i < N - 1; i++) {
            for(int j = i + 1; j < N; j++) {
                if(arr[i] > arr[j]) {
                    int temp = arr[j];
                    arr[j] = arr[i];
                    arr[i] = temp;
                }
            }
        }

        for(int val : arr) {
            System.out.println(val);
        }
    }
}
```

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.Arrays;
import java.util.StringTokenizer;
import java.io.IOException;

public class Main{
    public static void main(String[] args) throws IOException{
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int N = Integer.parseInt(br.readLine());
        int[][] arr = new int[N][2];

        StringTokenizer st;
        for(int i=0; i<N; i++)
        {
            st = new StringTokenizer(br.readLine());
            arr[i][0] = Integer.parseInt(st.nextToken());
            arr[i][1] = Integer.parseInt(st.nextToken());
        }

        Arrays.sort(arr, (e1,e2) -> {
            if(e1[0] == e2[0])
            {
                return e1[1] - e2[1];
            }
            else
            {
                return e1[0] - e2[0];
            }
        });

        StringBuilder sb = new StringBuilder();
        for(int i=0; i<N; i++)
        {
            sb.append(arr[i][0] + " " + arr[i][1]).append('\n');
        }
        System.out.println(sb);
    }
}
```


문제 01) 배열 정리하기

```
import java.util.Arrays;
import java.util.Random;

class Solution {
    public static void main(String[] args) {
        // 배열을 랜덤 값으로 초기화
        int[] arr = new int[100000];
        Random random = new Random();
        for (int i = 0; i < arr.length; i++) {
            arr[i] = random.nextInt(100000); // 0부터 99999까지의 랜덤 값
        }

        // 버블 정렬 시간 측정
        long start = System.currentTimeMillis();
        int[] bubble = bubbleSort(arr);
        long end = System.currentTimeMillis();
        System.out.println("Bubble sort time: " + (end - start) / 1000.0 + "초");

        // 배열을 다시 초기화하여 동일한 값으로 정렬 수행
        for (int i = 0; i < arr.length; i++) {
            arr[i] = random.nextInt(100000);
        }

        // 내장 정렬 시간 측정
        start = System.currentTimeMillis();
        int[] sort = doSort(arr);
        end = System.currentTimeMillis();
        System.out.println("API sort time: " + (end - start) / 1000.0 + "초");

        // 두 배열이 같은지 확인
        System.out.println("Arrays are equal: " + Arrays.equals(bubble, sort));
    }
}
```

```
private static int[] bubbleSort(int[] org) {
    int[] arr = org.clone();
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int tmp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = tmp;
            }
        }
    }
    return arr;
}

private static int[] doSort(int[] org) {
    int[] arr = org.clone();
    Arrays.sort(arr);
    return arr;
}
}
```


문제 03) 두개 뽑아서 정리하기

```
import java.util.HashSet;

public class Solution {

    public static int[] solution(int[] numbers) {
        HashSet<Integer> set = new HashSet<>(); // ❶ 중복 값 제거를 위한 해쉬셋 생성
        // ❷ 두 수를 선택하는 모든 경우의 수를 반복문으로 구함
        for (int i = 0; i < numbers.length - 1; i++) {
            for (int j = i + 1; j < numbers.length; j++) {
                // ❸ 두 수를 더한 결과를 새로운 배열에 추가
                set.add(numbers[i] + numbers[j]);
            }
        }
        // ❹ 해쉬셋의 값을 오름차순 정렬하고 int[] 형태의 배열로 변환하여 반환
        return set.stream().sorted().mapToInt(Integer::intValue).toArray();
    }
}
```

예제 1)

```
import java.util.*;

public class Main
{
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        Integer numbers[] = new Integer[n];

        for(int i = 0; i < n; i++){
            numbers[i] = scan.nextInt();
        }

        Arrays.sort(numbers, Comparator.reverseOrder());

        for(int number : numbers){
            System.out.print(number + " ");
        }
    }
}
```

예제 2)

```
import java.util.*;

// 학생 클래스
class Student{
    String name;
    int score;

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

    public String getName(){
        return this.name;
    }

    public int getScore(){
        return this.score;
    }
}

public class Main
{
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();

        List<Student> students = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            String name = scan.next();
            int score = scan.nextInt();
            students.add(new Student(name, score));
        }

        // 정렬 후 출력
        students.stream().sorted(Comparator.comparing(Student::getScore))
            .forEach(s -> System.out.print(s.getName() + " "));

    }
}
```

예제 3)

```
import java.util.*;

public class Main
{
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        int k = scan.nextInt();

        Integer arrA[] = new Integer[n];
        Integer arrB[] = new Integer[n];

        for (int i = 0; i < n; i++) {
            arrA[i] = scan.nextInt();
        }

        for (int i = 0; i < n; i++) {
            arrB[i] = scan.nextInt();
        }

        Arrays.sort(arrA); // 오름차순 정렬
        Arrays.sort(arrB, Comparator.reverseOrder()); // 내림차순 정렬

        for(int i = 0; i < k; i++){
            if(arrA[i] >= arrB[i]) break;

            int temp = arrA[i];
            arrA[i] = arrB[i];
            arrB[i] = temp;
        }

        long result = Arrays.stream(arrA).mapToInt(i -> i).sum();

        System.out.println("result = " + result);
    }
}
```

문제05) 행렬의 곱셈

```
public class Solution {  
  
    public int[][] solution(int[][] arr1, int[][] arr2) {  
        // ❶ 행렬 arr1과 arr2의 행과 열의 수  
        int r1 = arr1.length;  
        int c1 = arr1[0].length;  
        int r2 = arr2.length;  
        int c2 = arr2[0].length;  
  
        // ❷ 결과를 저장할 2차원 배열 초기화  
        int[][] answer = new int[r1][c2];  
  
        // ❸ 첫 번째 행렬 arr1의 각 행과 두 번째 행렬 arr2의 각 열에 대해  
        for (int i = 0; i < r1; i++) {  
            for (int j = 0; j < c2; j++) {  
                // ❹ 두 행렬의 데이터를 곱해 결과 리스트에 더해줌  
                for (int k = 0; k < c1; k++) {  
                    answer[i][j] += arr1[i][k] * arr2[k][j];  
                }  
            }  
        }  
  
        return answer;  
    }  
}
```

문제06) 실패율

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

class Solution {
    public int[] solution(int N, int[] stages) {
        // 스테이지별 도달했으나 클리어하지 못한 사람 수
        int[] currentStages = new int[N + 1];
        // 스테이지별 도달한 사람 수
        int[] clearStages = new int[N + 1];

        for (int i = 0; i < stages.length; i++) {
            // 스테이지에 도달한 플레이어 수 증가
            for (int j = 0; j < stages[i]; j++) {
                clearStages[j] += 1;
            }
            // 도달 했으나 클리어하지 못한 플레이어 수 증가
            currentStages[stages[i] - 1] += 1;
        }

        // 스테이지 번호와 실패율을 저장할 map
        Map<Integer, Double> map = new HashMap<>();

        for (int i = 0; i < N; i++) {
            // 스테이지에 도달한 사람이 없거나 클리어한 사람이 없는 경우
            if (currentStages[i] == 0 || clearStages[i] == 0) {
                map.put(i + 1, 0.0);
            } else {
                // 실패율 계산하여 맵에 저장
                map.put(i + 1, (double) currentStages[i] / (double) clearStages[i]);
            }
        }

        List<Integer> list = new ArrayList<>(map.keySet());
        // 실패율을 기준으로 내림차순 정렬
        list.sort((o1, o2) -> Double.compare(map.get(o2), map.get(o1)));

        // 리스트를 배열로 변환하여 반환
        return list.stream().mapToInt(i -> i).toArray();
    }
}
```

문제 8, 올바른 괄호

```
class Solution {  
    boolean solution(String s) {  
        int openCount = 0;  
        int closeCount = 0;  
  
        for (int i = 0; i < s.length(); i++) {  
            if (s.charAt(i) == '(') {  
                openCount++;  
            } else if (s.charAt(i) == ')') {  
                closeCount++;  
            }  
            if (openCount < closeCount) {  
                return false;  
            }  
        }  
        if (openCount == closeCount) {  
            return true;  
        }  
        return false;  
    }  
}
```

스택(stack)을 사용하지 않은 경우

```
import java.util.*;  
  
class Solution {  
    boolean solution(String s) {  
        Stack<Character> stack = new Stack<>();  
        for (int i = 0; i < s.length(); i++) {  
            if (s.charAt(i) == '(') {  
                stack.push('(');  
            } else if (s.charAt(i) == ')') {  
                if (stack.isEmpty()) {  
                    return false;  
                }  
                stack.pop();  
            }  
        }  
        return stack.isEmpty();  
    }  
}
```

스택(stack)을 사용한 것

문제 11, 짝지어 제거하기

```
import java.util.Stack;

class Solution {
    public int solution(String s) {
        // 문자를 저장할 스택
        Stack<Character> stack = new Stack<>();

        // 문자열 순회
        for (int i = 0; i < s.length(); i++) {
            if (stack.isEmpty()) { // 스택이 비어있다면 현재 문자를 스택에 추가
                stack.push(s.charAt(i));
            } else {
                char ch = s.charAt(i); // 현재 문자

                // 스택의 맨 위 문자와 현재 문자가 같다면 스택에서 제거
                if (stack.peek() == ch) {
                    stack.pop();
                } // 같지 않다면 현재 문자를 스택에 추가
                else {
                    stack.push(ch);
                }
            }
        }

        return stack.isEmpty() ? 1 : 0;
    }
}
```

```
import java.util.Stack;

class Solution
{
    public int solution(String s)
    {
        Stack<Character> stack = new Stack<>();
        //for-each문 사용
        for(char a : s.toCharArray()){
            //스택이 비어있지 않고, 맨 윗값이 넣으려는 a값과 동일하다면
            if(!stack.isEmpty() && stack.peek() == a){
                //스택 맨 위의 값 제거
                stack.pop();
            } //그 이외엔 push
            else stack.push(c);
        }
        return stack.isEmpty()? 1 : 0;
    }
}
```

문제 13, 크레인 인형 뽑기 게임

```
import java.util.*;
class Solution {
    public int solution(int[][] board, int[] moves) {
        int answer = 0;
        int len = board[0].length;
        Stack<Integer> st = new Stack<>();

        for(int mv: moves){
            mv -= 1;
            for(int i=0; i<len; i++){
                if(board[i][mv] != 0){ //인형집을 위치에 인형이 있는 경우 board[mv][i]
                    if(st.size() > 0 && st.peek() == board[i][mv]){ //지금 뽑은 인형과 마지막 인형이 같다면
                        st.pop();
                        answer += 2;
                    }else{
                        st.push(board[i][mv]); //인형 넣기
                    }
                    board[i][mv] = 0; //인형뽑았으니 0으로 세팅
                    break;
                }
            }
        }
        return answer;
    }
}
```

Thank you.

자바 기초
ryp1662@gmail.com

Copyright © “Youngpyo Ryu” All Rights Reserved.
This document was created for the exclusive use of “Youngpyo Ryu”.
It must not be passed on to third parties except with the explicit prior consent of “Youngpyo Ryu”.