연습문제

문자열

❖ 문자열 압축

✓ 문제

- 데이터 처리 전문가가 되고 싶은 어피치는 문자열을 압축하는 방법에 대해 공부하고 있습니다.
- 최근에 대량의 데이터 처리를 위한 간단한 비손실 압축 방법에 대해 공부하고 있는데 문자열에서 같은 값이 연속해서 나타나는 것을 그 문자의 개수와 반복되는 값으로 표현하여 더 짧은 문자열로 줄여서 표현하는 알고리즘을 공부하고 있습니다.
- 간단한 예로 aabbaccc 의 경우 2a2ba3c(문자가 반복되지 않아 한 번만 나타난 경우 1은 생략) 와 같이 표현할 수 있는데 이러한 방식은 반복되는 문자가 적은 경우 압축률이 낮다는 단점이 있습니다. 예를 들면 abcabcdede 와 같은 문자열은 전혀 압축되지 않습니다.
- 어피치는 이러한 단점을 해결하기 위해 문자열을 1 개 이상의 단위로 잘라서 압축하여 더 짧은 문자열로 표현할 수 있는지 방법을 찾아보려고 합니다.
- 예를 들어 ababcdcdababcdcd 의 경우 문자를 1 개 단위로 자르면 전혀 압축되지 않지만 2개 단위로 잘라서 압축한다면 2ab2cd2ab2cd로 표현할 수 있습니다.
- 다른 방법으로 8개 단위로 잘라서 압축한다면 2ababcdcd로 표현할 수 있으며 이때가 가장 짧게 압축하여 표현할 수 있는 방법입니다.
- 다른 예로 abcabcdede 와 같은 경우 문자를 2개 단위로 잘라서 압축하면 abcabc2de 가 되지만 3개 단위로 자른다면 2abcdede가 되어 3개 단위가 가장 짧은 압축 방법이 됩니다. 이때 3개 단위로 자르고 마지막에 남는 문자열은 그대로 붙여주면 됩니다.

문자열

❖ 문자열 압축

✓ 문제

- 압축할 문자열 s가 매개변수로 주어질 때, 위에 설명한 방법으로 1 개 이상 단위로 문자열을 잘라 압축하여 표현한 문자열 중 가장 짧은 것의 길이를 return 하도록 solution 함수를 완성해주세요

✓ 제한사항

- s의 길이는 1 이상 1,000 이하입니다.
- s는 알파벳 소문자로만 이루어져 있습니다.

문자열

- 문자열 압축
  - ✓ 입출력 예
    - aabbaccc 7
    - ababcdcdababcdcd 9
    - abcabcdede 8
    - abcabcabcabcdededededede 14
    - xababcdcdababcdcd 17

문자열

- ❖ 문자열 압축
  - ✓ 입출력 예 설명
    - 입출력 예 #1
      - ◆ 문자열을 1 개 단위로 잘라 압축했을 때 가장 짧습니다.
    - 입출력 예 #2
      - ◆ 문자열을 8개 단위로 잘라 압축했을 때 가장 짧습니다.
    - 입출력 예 #3
      - ◆ 문자열을 3개 단위로 잘라 압축했을 때 가장 짧습니다.
    - 입출력 예 #4
      - ◆ 문자열을 2개 단위로 자르면 abcabcabcabc6de 가 됩니다.
      - ◆ 문자열을 3개 단위로 자르면 4abcdedededededede 가 됩니다.
      - ◆ 문자열을 4개 단위로 자르면 abcabcabcabc3dede가됩니다.
      - ◆ 문자열을 6개 단위로 자를 경우 2abcabc2dedede가 되며 이때의 길이가 14로 가장 짧습니다.
    - 입출력 예 #5
      - ◆ 문자열은 제일 앞부터 정해진 길이만큼 잘라야 하므로 주어진 문자열을 x / ababcdcd / ababcdcd로 자르는 것은 불가능
      - ◆ 어떻게 문자열을 잘라도 압축되지 않으므로 가장 짧은 길이는 17이 됩니다

문자열

❖ 문자열 압축

✓ Java

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

public class Solution {
    private static List<String> split(String source, int length) {
        List<String> tokens = new ArrayList<>();
        for (int startIndex = 0; startIndex < source.length(); startIndex += length) {
            int endIndex = startIndex + length;
            if (endIndex > source.length()) endIndex = source.length();
            tokens.add(source.substring(startIndex, endIndex));
        }
        return tokens;
    }
```

문자열

❖ 문자열 압축

✓ Java

```
private static int compress(String source, int length) {
    StringBuilder builder = new StringBuilder();

    String last = "";
    int count = 0;
    for (String token : split(source, length)) {
        if (token.equals(last)) {
            count++;
        } else {
            if (count > 1) builder.append(count);
            builder.append(last);
            last = token;
            count = 1;
        }
    }
    if (count > 1) builder.append(count);
    builder.append(last);

    return builder.length();
}
```

문자열

❖ 문자열 압축

✓ Java

```
public static int solution(String s) {
    int min = Integer.MAX_VALUE;
    for (int length = 1; length <= s.length(); length++) {
        int compressed = compress(s, length);
        if (compressed < min) {
            min = compressed;
        }
    }
    return min;
}

public static void main(String [] args) {
      System.out.println(solution("aabbacccc"));
      System.out.println(solution("ababcdcdababcdcd"));
      System.out.println(solution("abcabcdede"));
      System.out.println(solution("abcabcabcabcdededededede"));
      System.out.println(solution("xababcdcdababcdcd"));
}
}
```

문자열

❖ 문자열 압축

✓ Python

```
def solution(s):
    answer = len(s)
    for x in range(1, len(s) // 2 + 1):
        comp_len = 0
        comp = ''
        cnt = 1
        for i in range(0, len(s) + 1, x):
            temp = s[i:i + x]
            if comp == temp:
                cnt += 1
            elif comp != temp:
                comp_len += len(temp)
                if cnt > 1: comp_len += len(str(cnt))
                cnt = 1
                comp = temp
        answer = min(answer, comp_len)

    return answer
```

문자열

❖ 문자열 압축

✓ Python

```
print(solution("aabbaccc"))
print(solution("ababcdcdababcdcd"))
print(solution("abcabcdede"))
print(solution("abcabcabcabcdededededede"))
print(solution("xababcdcdababcdcd"))
```

문자열

- ❖ 가장 흔한 단어 찾기
  - ✓ 문제
    - 금지된 단어를 제외한 가장 흔한 단어 찾기
    - 입력은 단어들의 list 와 금지

      paragraph = "Bob hit a ball, the hit BALL flew far after it was hit.

      banned = ["hit"]
    - 출력

      ball

문자열

❖ 가장 흔한 단어 찾기

✓ Java

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

public class Solution {
    public static String solution(String paragraph, String[] banned) {
        List<String> words = new ArrayList<>(
                Arrays.asList(
                        paragraph.toLowerCase()
                        .replaceAll("[^₩₩w]+", " ")
                        .split(" ")));
        words.removeAll(Arrays.asList(banned));

        String mostCommonWord = words.get(0);
        int maxCount = 0;
```

문자열

- 가장 흔한 단어 찾기
  - ✓ Java

```
        for (String first : words) {
            int count = 0;
            for (String second : words) {
                if (Objects.equals(first, second)) {
                    count++;
                }
            }

            if (count > maxCount) {
                maxCount = count;
                mostCommonWord = first;
            }
        }

        return mostCommonWord;
    }
```

문자열

- 가장 흔한 단어 찾기
  - ✓ Java

```
        public static void main(String [] args) {
         String paragraph = "Bob hit a ball, the hit BALL flew far after it was hit";
         String [] banned = {"hit"};
         System.out.println(solution(paragraph, banned));
        }
    }
```

문자열

❖ 가장 흔한 단어 찾기

✓ python

```
import re
import collections

def solution(paragraph, banned):
    words = [word for word in re.sub(r'[^₩w]', ' ', paragraph)
    .lower().split()
            if word not in banned]

    counts = collections.Counter(words)
    # 가장 흔하게 등장하는 단어의 첫 번째 인덱스 리턴
    return counts.most_common(1)[0][0]


paragraph = "Bob hit a ball, the hit BALL flew far after it was hit"
banned = ["hit"]
print(solution(paragraph, banned))
```

문자열

❖ 그룹 애너그램

✓ 문제

- 애너그램은 문자를 재배열하여 다른 뜻을 가진 단어로 바꾸는 것
- 입력은 단어들의 리스트

  ["eat", "tea", "tan", "ate", "nat", "bat"]
- 출력은 애너그램들의 리스트의 리스트

```
[
    ["ate","eat","tea"],
    ["nat",''tan"],
    ["bat"]
]
```

문자열

❖ 그룹 애너그램

✓ java

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
public class Solution {
    public static List<List<String>> solution(String[] strs) {
        if (strs.length == 0) {
            return new ArrayList<>();
        }
        Map<String, List> groupAnagrams = new HashMap<>();
        for (String str : strs) {
            char[] chars = str.toCharArray();
            Arrays.sort(chars);
            String key = String.valueOf(chars);
            if (!groupAnagrams.containsKey(key)) {
                groupAnagrams.put(key, new ArrayList<>());
            }
            groupAnagrams.get(key).add(str);
        }
        return new ArrayList(groupAnagrams.values());
    }
```

문자열

- 그룹 애너그램
  - ✓ java

```
        public static void main(String [] args) {
         String [] input = {"eat", "tea", "tan", "ate", "nat", "bat"};
         System.out.println(solution(input));
        }
    }
```

문자열

❖ 그룹 애너그램

✓ python

```
import collections


def solution(strs):
    anagrams = collections.defaultdict(list)

    for word in strs:
        # 정렬하여 딕셔너리에 추가
        anagrams[''.join(sorted(word))].append(word)
    return list(anagrams.values())


print(solution(["eat", "tea", "tan", "ate", "nat", "bat"]))
```

문자열

❖ 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기

✓ 문제

- 문자열 내에서 가장 긴 팰린드롬을 찾아서 출력

- 입력1: babad
- 출력1: bab 또는 aba

- 입력2: cbbd
- 출력2: bb

문자열

- 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기
  - ✓ java

```
public class Solution {
    private static boolean isPalindrom(String str){
        for(int i = 0; i<str.length()/2; i++){
            int compIdx = str.length()-i-1;
            if(str.charAt(i) == (str.charAt(compIdx)) == false){
                return false;
            }
        }
        return true;
    }
```

문자열

❖ 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기

✓ java

```
public static String solution(String s) {
    String answer = "";
    int len = s.length();
    for(int i = len ; i > 0 ; i--){
        for(int j = 0 ; j < len ; j++){
            int head = j;
            int tail = j+i;
             if(tail > len){
                break;
            }

            String temp = s.substring(head,tail);
            if( isPalindrom(temp)){
                if(answer.length() < temp.length()){
                    answer = temp;
                }
            }

        }
    }
    return answer;
}
```

문자열

- 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기
  - ✓ java

```
        public static void main(String [] args) {
         System.out.println(solution("babad"));
         System.out.println(solution("cbbd"));
        }
    }
```

문자열

❖ 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기

✓ python

```
def solution(s):
    # 팰린드롬 판별 및 투 포인터 확장
    def expand(left: int, right: int) -> str:
        while left >= 0 and right < len(s) and s[left] == s[right]:
            left -= 1
            right += 1
        return s[left + 1:right]

    # 해당 사항이 없을때 빠르게 리턴
    if len(s) < 2 or s == s[::-1]:
        return s

    result = ''
    # 슬라이딩 윈도우 우측으로 이동
    for i in range(len(s) - 1):
        result = max(result,
                     expand(i, i + 1),
                     expand(i, i + 2),
                     key=len)
    return result
```

문자열

- 가장 긴 팰린드롬(앞 뒤가 똑같은 단어나 문장 – 회문) 구하기
  - ✓ python

```
print(solution("babad"))
print(solution("cbbd"))
```

문자열

❖ 이진 변환 반복

✓ 문제

- 0과 1로 이루어진 어떤 문자열 x에 대한 이진 변환을 다음과 같이 정의
  - x의 모든 0을 제거합니다.
  - x의 길이를 c라고 하면 x를 c를 2진법으로 표현한 문자열로 바꿉니다.
- 예를 들어, x = 0111010 이라면, x에 이진 변환을 가하면 x = 0111010 -> 1111 -> 100 이 됩니다.
- 0과 1로 이루어진 문자열 s가 매개변수로 주어집니다. s가 1이 될 때까지 계속해서 s에 이진 변환을 가했을 때 이진 변환의 횟수와 변환 과정에서 제거된 모든 0의 개수를 각각 배열에 담아 return 하도록 solution 함수를 완성해주세요.
- 제한사항
  - s의 길이는 1 이상 150,000 이하입니다.
  - s에는 '1'이 최소 하나 이상 포함되어 있습니다.

문자열

- ❖ 이진 변환 반복
  - ✓ 입출력 예
    - 입력 출력
      - "110010101001" [3,8]
      - "01110" [3,3]
      - "1111111" [4,1]
    - 입출력 예1

회차	이진 변환 이전	제거할 0의 개수	0 제거 후 길이	이진 변환 결과
1	"110010101001"	6	6	"110"
2	"110"	1	2	"10"
3	"10"	1	1	"1"

문자열

- ❖ 이진 변환 반복
  - ✓ 입출력 예
    - ● 입출력 예2

회차	이진 변환 이전	제거할 0의 개수	0 제거 후 길이	이진 변환 결과
1	"01110"	2	3	"11"
2	"11"	0	2	"10"
3	"10"	1	1	"1"

문자열

- ❖ 이진 변환 반복
  - ✓ 입출력 예
    - ● 입출력 예3

회차	이진 변환 이전	제거할 0의 개수	0 제거 후 길이	이진 변환 결과
1	"1111111"	0	7	"111"
2	"111"	0	3	"11"
3	"11"	0	2	"10"
4	"10"	1	1	"1"

문자열

❖ 이진 변환 반복

✓ java

```
public class Solution {
    public static int[] solution(String s) {
        int loop = 0;
        int removed = 0;

        while (!s.equals("1")) {
            int zeros = 0;

            for (char c : s.toCharArray()) {
                if (c == '0') zeros++;
            }

            loop += 1;
            removed += zeros;

            int ones = s.length() - zeros;
            s = Integer.toString(ones, 2);
        }

        return new int[] {loop, removed};
    }
```

문자열

- 이진 변환 반복
  - ✓ java

```
        public static void main(String [] args) {
         System.out.println(Arrays.toString(solution("110010101001")));
         System.out.println(Arrays.toString(solution("01110")));
         System.out.println(Arrays.toString(solution("1111111")));
        }
    }
```

문자열

❖ 이진 변환 반복

✓ python

```
def solution(s):
    change, zero = 0, 0
    while s != '1':
        change += 1
        num = s.count('1')
        zero += len(s) - num
        s = bin(num)[2:]
    return [change, zero]


print(solution("110010101001"))
print(solution("01110"))
print(solution("1111111"))
```

배열

❖ 주몽의 명령

✓ 문제

- 주몽은 철기군을 양성하기 위한 프로젝트에 나섰다. 그래서 야철대장에게 철기군이 입을 갑옷을 만들라고 명령했다.
- 야철대장은 주몽의 명령에 따르기 위해 연구에 착수하던 중 갑옷을 만드는 재료들은 각각 고유한 번호가 있고, 갑옷은 2개의 재료로 만드는 데 2가지 재료의 고유한 번호를 합쳐 M(1 < M < 10,000,000)이 되면 갑옷이 만들어진다는 사실을 발견했다.
- 야철대장은 자신이 만들고 있는 재료로 갑옷을 몇 개나 만들 수 있는지 궁금해졌다.
- 야철대장의 궁금증을 풀어 주기 위해 N(1 < N < 15,000)개의 재료와 M이 주어졌을 때 몇 개의 갑옷을 만들 수 있는지를 구하는 함수를 작성하시오.
- 입력: 갑옷을 만드는데 필요한 재료 배열 과 갑옷이 완성되는 재료 의 합
- 출력: 갑옷이 만들어지는 개수
- [2, 7, 4, 1 , 5, 3] 과 9가 주어지면 출력은 2
- 2+7=9, 4+5=9

배열

❖ 주몽의 명령

✓ java

```
public class Solution {
    public static int solution(int [] inputs, int hap) {
        Arrays.sort(inputs);
        int count = 0;
        int i = 0;
        int j = inputs.length - 1;
        while (i < j) {
          if (inputs[i] + inputs[j] < hap) {
            i++;
          } else if (inputs[i] + inputs[j] > hap) {
            j--;
          } else {
            count++;
            i++;
            j--;
          }
        }
        return count;
    }
```

배열

- 주몽의 명령
  - ✓ java

```
        public static void main(String [] args) {
         int [] inputs = {2, 7, 4, 1, 5, 3};
         int hap = 9;
         System.out.println((solution(inputs, hap)));
        }
    }
```

배열

❖ 주몽의 명령

✓ python

```
def solution(inputs, hap):
    inputs.sort()
    count = int(0)
    i = int(0)
    j = int(len(inputs) - 1)
    while i < j:  # 투 포인터 이동 원칙 따라 포인터를 이동하며 처리
        if inputs[i] + inputs[j] < hap:
            i += 1
        elif inputs[i] + inputs[j] > hap:
            j -= 1
        else:
            count += 1
            i += 1
            j -= 1
    return count


print(solution([2, 7, 4, 1, 5, 3], 9))
```

배열

❖ 달팽이

✓ 문제 – 아래 와 같은 형태로 데이터를 저장하는 이차원 배열을 생성하시오.

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

배열

❖ 달팽이

✓ java

```
public class Solution {
    public static int[][] solution() {
     int[][] snail = new int[5][5];

        int print = 5;
        int k = 1;
        int right = -1;
        int bottom = 0;
        int top = 1;
```

배열

❖ 달팽이

✓ java

```
        for(int i=5; i>0; i--) {

            for(int j=0; j<print; j++) {
                right += top;
                snail[bottom][right] = k;
                k++;
            }

            print--;

            for(int j=0; j<print; j++) {
                bottom += top;
                snail[bottom][right] = k;
                k++;
            }

            top = top * (-1);
        }

        return snail;
    }
```

배열

- 달팽이
  - ✓ java

```
public static void main(String [] args) {
 int [][] snail = solution();
 for(int [] ar : snail) {
        for(int i : ar) {
              System.out.printf("%3d", i);
        }
        System.out.println();
    }
  }
}
```

배열

❖ 달팽이

✓ python

```
def solution():
    snail = [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
    end = 5
    k = 1
    right = -1
    bottom = 0
    top = 1
```

배열

❖ 달팽이

✓ python

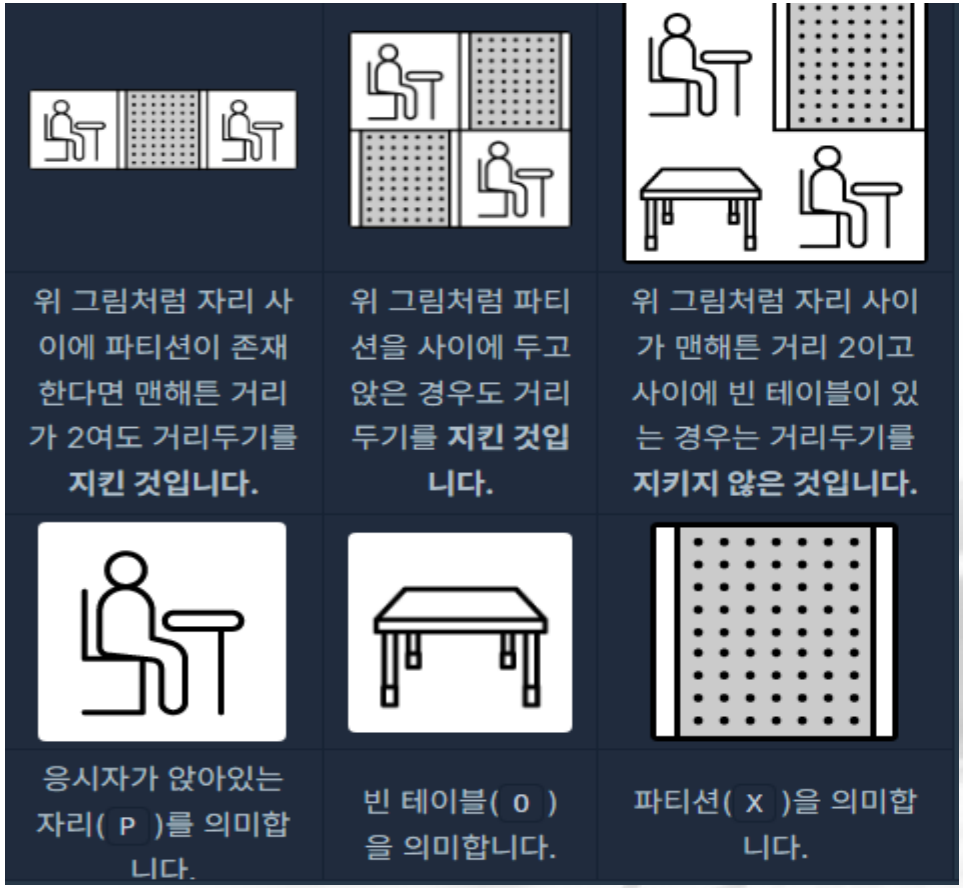
```
        for i in range(5, 0, -1):
            for j in range(0, end, 1):
                right += top
                snail[bottom][right] = k
                k += 1

            end -= 1

            for j in range(0, end, 1):
                bottom += top
                snail[bottom][right] = k
                k += 1

            top = top * (-1)

        return snail


    print(solution())
```

배열

- ❖ 거리 두기 확인
  - ✓ 문제
    - 개발자를 희망하는 죠르디가 카카오에 면접을 보러 왔습니다.
    - 코로나 바이러스 감염 예방을 위해 응시자들은 거리를 둬서 대기를 해야하는데 개발 직군 면접인 만큼 아래와 같은 규칙으로 대기실에 거리를 두고 앉도록 안내하고 있습니다.
    - 맨해튼 거리: 각 차원의 차이를 절대값으로 변환해서 구한 합계
    - 조건
      - ◆ 대기실은 5개이며, 각 대기실은 5x5 크기입니다.
      - ◆ 거리두기를 위하여 응시자들 끼리는 맨해튼 거리가 2 이하로 앉지 말아 주세요.
      - ◆ 단 응시자가 앉아있는 자리 사이가 파티션으로 막혀 있을 경우에는 허용합니다.

배열

- ❖ 거리 두기 확인
  - ✓ 문제
    - ● 조건



배열

❖ 거리 두기 확인

✓ 문제

- 5개의 대기실을 본 죠르디는 각 대기실에서 응시자들이 거리두기를 잘 기키고 있는지 알고 싶어졌습니다. 자리에 앉아있는 응시자들의 정보와 대기실 구조를 대기실별로 담은 2차원 문자열 배열 places가 매개변수로 주어집니다. 각 대기실별로 거리두기를 지키고 있으면 1을, 한 명이라도 지키지 않고 있으면 0을 배열에 담아 return 하도록 solution 함수를 완성해 주세요.

배열

- ❖ 거리 두기 확인
  - ✓ 제한사항
    - places의 행 길이(대기실 개수) = 5
    - places의 각 행은 하나의 대기실 구조를 나타냅니다.
    - places의 열 길이(대기실 세로 길이) = 5
    - places의 원소는 P,O,X로 이루어진 문자열입니다.
    - places 원소의 길이(대기실 가로 길이) = 5
    - P는 응시자가 앉아있는 자리를 의미합니다.
    - O는 빈 테이블을 의미합니다.
    - X는 파티션을 의미합니다.
    - 입력으로 주어지는 5개 대기실의 크기는 모두 5 x 5 입니다.
    - return 값 형식은 1차원 정수 배열에 5개의 원소를 담아서 return 합니다.
    - Places에 담겨 있는 5개 대기실의 순서대로 거리두기 준수 여부를 차례대로 배열에 담습니다.
    - 각 대기실 별로 모든 응시자가 거리두기를 지키고 있으면 1을, 한 명이라도 지키지 않고 있으면 0을 담습니다.

배열

❖ 거리 두기 확인

✓ 입출력

places	result
[["POOOP", "OXXOX", "OPXPX", "OOXOX", "POXXP"], ["POOPX", "OXPXP", "PXXXO", "OXXXO", "OOOPP"], ["PXOPX", "OXOXP", "OXPOX", "OXXOP", "PXPOX"], ["OOOXX", "XOOOX", "OOOXX", "OXOOX", "OOOOO"], ["PXPXP", "XPXPX", "PXPXP", "XPXPX", "PXPXP"]]	[1, 0, 1, 1, 1]

배열

❖ 거리 두기 확인

✓ 입출력

첫 번째 대기실

No.	0	1	2	3	4
0	P	O	O	O	P
1	O	X	X	O	X
2	O	P	X	P	X
3	O	O	X	O	X
4	P	O	X	X	P

- 모든 응시자가 거리두기를 지키고 있습니다.

배열

❖ 거리 두기 확인

✓ 입출력

두 번째 대기실

No.	0	1	2	3	4
0	P	O	O	P	X
1	O	X	P	X	P
2	P	X	X	X	O
3	O	X	X	X	O
4	O	O	O	P	P

- (0, 0) 자리의 응시자와 (2, 0) 자리의 응시자가 거리두기를 지키고 있지 않습니다.
- (1, 2) 자리의 응시자와 (0, 3) 자리의 응시자가 거리두기를 지키고 있지 않습니다.
- (4, 3) 자리의 응시자와 (4, 4) 자리의 응시자가 거리두기를 지키고 있지 않습니다.

배열

❖ 거리 두기 확인

✓ 입출력

세 번째 대기실

No.	0	1	2	3	4
0	P	X	O	P	X
1	O	X	O	X	P
2	O	X	P	O	X
3	O	X	X	O	P
4	P	X	P	O	X

- 모든 응시자가 거리두기를 지키고 있습니다.

배열

❖ 거리 두기 확인

✓ 입출력

네 번째 대기실

No.	0	1	2	3	4
0	O	O	O	X	X
1	X	O	O	O	X
2	O	O	O	X	X
3	O	X	O	O	X
4	O	O	O	O	O

- 대기실에 응시자가 없으므로 거리두기를 지키고 있습니다.

배열

❖ 거리 두기 확인

✓ 입출력

다섯 번째 대기실

No.	0	1	2	3	4
0	P	X	P	X	P
1	X	P	X	P	X
2	P	X	P	X	P
3	X	P	X	P	X
4	P	X	P	X	P

- 모든 응시자가 거리두기를 지키고 있습니다.

배열

❖ 거리 두기 확인

✓ java

```
import java.util.Arrays;

public class Solution {
        private static final int dx[] = {0, -1, 1, 0};
        private static final int dy[] = {-1, 0, 0, 1};

        private static boolean isNextToVolunteer(char[][] room, int x, int y, int exclude) {
            for (int d = 0; d < 4; d++) {
                if (d == exclude) continue;

                int nx = x + dx[d];
                int ny = y + dy[d];
                if (ny < 0 || ny >= room.length || nx < 0 || nx >= room[ny].length)
                    continue;
                if (room[ny][nx] == 'P') return true;
            }
            return false;
        }
```

배열

❖ 거리 두기 확인

✓ java

```
private static boolean isDistanced(char[][] room, int x, int y) {
    for (int d = 0; d < 4; d++) {
        int nx = x + dx[d];
        int ny = y + dy[d];
        if (ny < 0 || ny >= room.length || nx < 0 || nx >= room[ny].length)
            continue;

        switch (room[ny][nx]) {
            case 'P': return false;
            case 'O':
                if (isNextToVolunteer(room, nx, ny, 3- d)) return false;
                break;
        }
    }
    return true;
}
```

배열

- 거리 두기 확인
  - ✓ java

```
private static boolean isDistanced(char[][] room) {
    for (int y = 0; y < room.length; y++) {
        for (int x = 0; x < room[y].length; x++) {
            if (room[y][x] != 'P') continue;
            if (!isDistanced(room, x, y)) return false;
        }
    }
    return true;
}
```

배열

❖ 거리 두기 확인

✓ java

```
public static int[] solution(String[][] places) {
    int[] answer = new int[places.length];
    for (int i = 0; i < answer.length; i++) {
        String[] place = places[i];
        char[][] room = new char[place.length][];
        for (int j = 0; j < room.length; j++) {
            room[j] = place[j].toCharArray();
        }
        if (isDistanced(room)) {
            answer[i] = 1;
        } else {
            answer[i] = 0;
        }
    }
    return answer;
}
```

배열

❖ 거리 두기 확인

✓ java

```
public static void main(String [] args) {
 String [][] ar = {{"POOOP", "OXXOX", "OPXPX", "OOXOX", "POXXP"},
                   {"POOPX", "OXPXP", "PXXXO", "OXXXO", "OOOPP"},
                   {"PXOPX", "OXOXP", "OXPOX", "OXXOP", "PXPOX"},
                   {"OOOXX", "XOOOX", "OOOXX", "OXOOX", "OOOOO"},
                   {"PXPXP", "XPXPX", "PXPXP", "XPXPX", "PXPXP"}};

 System.out.println(Arrays.toString(solution(ar)));
}
}
```

배열

❖ 행렬의 곱

✓ 문제

- 2차원 행렬 arr1과 arr2를 입력받아, arr1에 arr2를 곱한 결과를 반환하는 함수 solution을 완성해주세요.

✓ 제한 조건

- 행렬 arr1, arr2의 행과 열의 길이는 2 이상 100 이하입니다.
- 행렬 arr1, arr2의 원소는 -10 이상 20 이하인 자연수입니다.
- 곱할 수 있는 배열만 주어집니다.

배열

❖ 행렬의 곱

✓ 입출력 예

arr1	arr2	return
[[1, 4], [3, 2], [4, 1]]	[[3, 3], [3, 3]]	[[15, 15], [15, 15], [15, 15]]
[[2, 3, 2], [4, 2, 4], [3, 1, 4]]	[[5, 4, 3], [2, 4, 1], [3, 1, 1]]	[[22, 22, 11], [36, 28, 18], [29, 20, 14]]

배열

❖ 행렬의 곱

✓ java

```
public class Solution {
    public static int[][] solution(int[][] ar1, int[][] ar2) {
        int[][] arr = new int[ar1.length][ar2[0].length];
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                arr[i][j] = 0;
                for (int k = 0; k < ar1[i].length; k++) {
                    arr[i][j] += ar1[i][k] * ar2[k][j];
                }
            }
        }
        return arr;
    }
```

배열

- 행렬의 곱
  - ✓ java

```
public static void main(String [] args) {
 int [][] ar1 = {{1,4}, {3, 2}, {4, 1}};
 int [][] ar2 = {{3, 3}, {3, 3}};

 int [][] result = solution(ar1, ar2);
 for(int [] i : result) {
          for(int j : i) {
                   System.out.printf("%3d", j);
          }
          System.out.println();
}

 System.out.println();
```

배열

❖ 행렬의 곱

✓ java

```
        int [][] ar3 = {{2, 3, 2}, {4, 2, 4}, {3, 1, 4}};
        int [][] ar4 = {{5, 4, 3}, {2, 4, 1}, {3, 1, 1}};

        result = solution(ar3, ar4);
        for(int [] i : result) {
                for(int j : i) {
                        System.out.printf("%3d", j);
                }
                System.out.println();
        }
       }
    }
```

배열

❖ 행렬의 곱

✓ python

```
def solution(ar1, ar2):
    answer = [[0 for _ in range(len(ar2[0]))] for _ in range(len(ar1))]
    for i in range(len(ar1)):
        for j in range(len(ar2[0])):
            for k in range(len(ar1[0])):
                answer[i][j] += (ar1[i][k] * ar2[k][j])
    return answer


print(solution([[1,4], [3, 2], [4, 1]], [[3, 3], [3, 3]]))
print(solution([[2, 3, 2], [4, 2, 4], [3, 1, 4]], [[5, 4, 3], [2, 4, 1], [3, 1, 1]]))
```

배열

- ❖ 주식의 팔고 사기 좋은 시점 찾기
  - ✓ 문제
    - 주식의 가격 정보가 들어있는 1차원 배열을 입력받아서 언제 사서 언제 파는 것이 가장 많은 이득을 남기는지 이득을 리턴해주는 함수를 작성하시오.
    - 입출력 예1
      - 입력: [7, 1, 5, 3, 6, 4]
      - 출력: 5
    - 입출력 예2
      - 입력: [7, 6, 4, 3, 1]
      - 출력: 0

배열

- 주식의 팔고 사기 좋은 시점 찾기
  - ✓ 제한 사항
    - 배열의 길이는 1부터 100 000 사이
    - 가격은 0부터 10,000 사이
    - 실행 시간은 1초 이내 – 시간 복잡도가 $O(N^2)$ 보다 작아야 함

배열

❖ 주식의 팔고 사기 좋은 시점 찾기

✓ java

```
public class Solution {
    public static int solution(int[] prices) {
            int min_num = Integer.MAX_VALUE;
            int min_index = -1;
            int res = 0;

            for(int i=0; i<prices.length; i++) {
                    if(prices[i] < min_num) {
                            min_num = prices[i];
                            min_index = i;
                    }
                    if((prices[i] > min_num) && (i > min_index)) {
                            res = Math.max(res, prices[i] - min_num);
                    }
            }
            return res;
    }
```

배열

❖ 주식의 팔고 사기 좋은 시점 찾기

✓ java

```
public static void main(String [] args) {
        int [] ar1 = {7, 1, 5, 3, 6, 4};
        int [] ar2 = {7, 6, 4, 3, 1};
        System.out.println(solution(ar1));
        System.out.println(solution(ar2));
    }
}
```

배열

❖ 주식의 팔고 사기 좋은 시점 찾기

✓ python

```
import sys


def solution(prices):
    profit = 0
    min_price = sys.maxsize

    # 최소값과 최대값 계속 갱신
    for price in prices:
        min_price = min(min_price, price)
        profit = max(profit, price - min_price)

    return profit


print(solution([7, 1, 5, 3, 6, 4]))
print(solution([7, 6, 4, 3, 1]))
```

선형 자료 구조

- ❖ 괄호 Shift
  - ✓ 문제
    - ● 다음 규칙을 지키는 문자열을 올바른 괄호 문자열이라고 정의
      - ◆ (), [], {} 는 모두 올바른 괄호 문자열입니다.
      - ◆ 만약 A가 올바른 괄호 문자열이라면, (A), [A], {A} 도 올바른 괄호 문자열입니다. 예를 들어, [] 가 올바른 괄호 문자열이므로, ([]) 도 올바른 괄호 문자열입니다.
      - ◆ 만약 A, B가 올바른 괄호 문자열이라면, AB 도 올바른 괄호 문자열입니다. 예를 들어, {} 와 ([]) 가 올바른 괄호 문자열이므로, {}([]) 도 올바른 괄호 문자열입니다.
      - ◆ 대괄호, 중괄호, 그리고 소괄호로 이루어진 문자열 s가 매개변수로 주어집니다. 이 s를 왼쪽으로 x (0 ≤ x < (s의 길이)) 칸만큼 이동시켜 회전시켰을 때 s가 올바른 괄호 문자열이 되게 하는 x의 개수를 return 하도록 solution 함수를 완성해주세요.

선형 자료 구조

- ❖ 괄호 Shift
  - ✓ 문제
    - ● 입출력

입력	출력
[ ] ( ) { }	3

x	s를 왼쪽으로 x칸만큼 shift	올바른 괄호 문자열?
0	[ ] ( ) { }	O
1	] ( ) { } [	X
2	( ) { } [ ]	O
3	) { } [ ] (	X
4	{ } [ ] ( )	O
5	} [ ] ( ) {	X

선형 자료 구조

- 괄호 Shift
  - ✓ java

```
public class Solution {
    private static boolean isCorrect(char[] str, int offset) {
        Stack<Character> stack = new Stack<>();

        for (int i = 0; i < str.length; i++) {
            char c = str[(offset + i) % str.length];
            switch (c) {
                case '(' -> stack.push(')');
                case '{' -> stack.push('}');
                case '[' -> stack.push(']');
                case ')', '}', ']' -> {
                    if (stack.isEmpty()) return false;
                    if (stack.pop() != c) return false;
                }
            }
        }

        return stack.isEmpty();
    }
```

선형 자료 구조

- 괄호 Shift
  - ✓ java

```
public static int solution(String s) {
    char[] str = s.toCharArray();

    int count = 0;
    for (int offset = 0; offset < str.length; offset++) {
        if (isCorrect(str, offset)) {
            count++;
        }
    }
    return count;
}
```

선형 자료 구조

- 괄호 Shift
  - ✓ java

```
public static void main(String [] args) {
        String s1 = "[](){}";
        String s2 = "}](){[{";
        String s3 = "](()[";
        String s4 = "}}}";
        System.out.println(solution(s1));
        System.out.println(solution(s2));
        System.out.println(solution(s3));
        System.out.println(solution(s4));
    }
}
```

선형 자료 구조

- 괄호 Shift
  - ✓ python

```
def isCorrect(s):
    stack = []
    for i in s:
        if len(stack) == 0:
            stack.append(i)
        else:
            if i == ")" and stack[-1] == "(":
                stack.pop()
            elif i == "]" and stack[-1] == "[":
                stack.pop()
            elif i == "}" and stack[-1] == "{":
                stack.pop()
            else:
                stack.append(i)
    return 1 if len(stack) == 0 else 0
```

선형 자료 구조

❖ 괄호 Shift

✓ python

```
def solution(s):
    answer = 0

    for i in range(len(s)):
        if isCorrect(s):
            answer += 1
        s = s[1:] + s[:1]
    return answer


s1 = "[](){}"
s2 = "}]()[{"
s3 = "[)(]"
s4 = "}}}"

print(solution(s1))
print(solution(s2))
print(solution(s3))
print(solution(s4))
```

선형 자료 구조

- 정렬된 리스트 병합
  - ✓ 문제
    - 연결된 목록의 2차원 배열을 제공받아서 모든 목록을 하나의 정렬 된 목록으로 병합하고 반환하는 함수를 작성하시오
    - 입력: [[1,4,5],[1,3,4],[2,6]]
    - 출력: [1,1,2,3,4,4,5,6]
    - 제한 사항
      - k == lists.length
      - 0 <= k <= 10^4
      - 0 <= lists[i].length <= 500
      - -10^4 <= lists[i][j] <= 10^4
      - lists[i] 는 오름차순으로 정렬된 상태

선형 자료 구조

❖ 정렬된 리스트 병합

✓ java

```
import java.util.Arrays;

public class Solution {

    public static int [] solution(int [][] lists) {
        int[] answer = {};
        int [] result = null;
        for(int [] ar : lists) {
                result = new int[answer.length + ar.length];
                int i=0;
                int j=0;
```

선형 자료 구조

- 정렬된 리스트 병합
  - ✓ java

```
            while(i < answer.length && j < ar.length) {
                    if(answer[i] > ar[j]) {
                            result[i+j] = ar[j];
                            j++;
                    }else {
                            result[i+j] = answer[i];
                            i++;
                    }
            }
            if(i < answer.length) {
                    for(; i < answer.length; i++) result[i+j] = answer[i];
            }
            if(j < ar.length) {
                    for(; j < ar.length; j++) result[i+j] = ar[j];

            }
            answer = result;
        }
        return result;
    }
```

선형 자료 구조

❖ 정렬된 리스트 병합

✓ java

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

선형 자료 구조

❖ 정렬된 리스트 병합

✓ python

```
def solution0(lists):
    data = []
    for li in lists:
        for i in li:
            data.append(i)
    data = sorted(data)
    return data
```

선형 자료 구조

❖ 정렬된 리스트 병합

✓ python

```
def solution1(lists):
    result = []
    for ar in lists:
        i = j = 0
        while i < len(result) and j < len(ar):
            if result[i] > ar[j]:
                result.insert(i, ar[j])
                j += 1
            else:
                i += 1
        if j < len(ar):
            result = result + ar[j:]

    return result

print(solution0([[1, 4, 5], [1, 3, 4], [2, 6]]))
print(solution1([[1, 4, 5], [1, 3, 4], [2, 6]]))
```

선형 자료 구조

- ❖ 주식 가격
  - ✓ 문제
    - 초 단위로 기록된 주식가격이 담긴 배열 prices가 매개변수로 주어질 때 가격이 떨어지지 않은 기간은 몇 초인지를 return 하도록 solution 함수를 완성하세요.
    - 제한사항
      - ◆ prices의 각 가격은 1 이상 10,000 이하인 자연수입니다.
      - ◆ prices의 길이는 2 이상 100,000 이하입니다.

선형 자료 구조

- ❖ 주식 가격
  - ✓ 문제
    - 입출력
      - 입력: [1, 2, 3, 2, 3]
      - 출력: [4, 3, 1, 1, 0]
    - 설명
      - 1초 시점의 1은 끝까지 가격이 떨어지지 않았습니다.
      - 2초 시점의 2은 끝까지 가격이 떨어지지 않았습니다.
      - 3초 시점의 3은 1초 뒤에 가격이 떨어집니다. 따라서 1초간 가격이 떨어지지 않은 것으로 봅니다.
      - 4초 시점의 2은 1초간 가격이 떨어지지 않았습니다.
      - 5초 시점의 3은 0초간 가격이 떨어지지 않았습니다.

선형 자료 구조

- 주식 가격
  - ✓ Java

```
import java.util.Arrays;
import java.util.Stack;
public class Solution {
    public static int [] solution(int [] prices) {
        int[] answer = new int[prices.length];

        Stack<Integer> stack = new Stack<>();
        for (int i = 0; i < prices.length; i++) {
            while (!stack.isEmpty() && prices[stack.peek()] > prices[i]) {
                int index = stack.pop();
                answer[index] = i - index;
            }

            stack.push(i);
        }

        while (!stack.isEmpty()) {
            int index = stack.pop();
            answer[index] = prices.length - index - 1;
        }
        return answer;
    }
```

선형 자료 구조

- 주식 가격
  - ✓ Java

```
        public static void main(String [] args) {
                int [] ar = {1, 2, 3, 2, 3};
                System.out.println(Arrays.toString(solution(ar)));
        }
    }
```

선형 자료 구조

❖ 주식 가격

✓ Python

```
def solution(prices):
    size = len(prices)
    answer = [0] * size
    for i in range(size):
        for j in range(i + 1, size):
            if prices[i] <= prices[j]:
                answer[i] += 1
            else:
                answer[i] += 1
                break
    return answer


print(solution([1, 2, 3, 2, 3]))
```

선형 자료 구조

- 오큰수 구하기
  - ✓ 문제
    - 크기가 N인 수열 A = A_1, A_2, ..., A_N이 있다. 수열의 각 원소 A_i에 대해서 오큰수 NGE(i)를 구하려고 한다. A_i의 오큰수는 오른쪽에 있으면서 A_i보다 큰 수 중에서 가장 왼쪽에 있는 수를 의미한다. 그러한 수가 없는 경우에 오큰수는 -1을 리턴하는 함수를 완성하시오
    - A = [3, 5, 2, 7]인 경우 NGE(1) = 5, NGE(2) = 7, NGE(3) = 7, NGE(4) = -1이다.
    - A = [9, 5, 4, 8]인 경우에는 NGE(1) = -1, NGE(2) = 8, NGE(3) = 8, NGE(4) = -1이다.

선형 자료 구조

❖ 오큰수 구하기

✓ java

```
import java.util.Arrays;
import java.util.Stack;

public class Solution {

    public static int [] solution(int [] ar) {
        int n = ar.length;

        int [] ans = new int[n]; // 정답 배열 생성

        Stack<Integer> myStack = new Stack<>();
        myStack.push(0); // 처음에는 항상 스택이 비어있으므로 최초 값을 push하여
    초기화

        for (int i = 1; i < n; i++) {
            //스택 비어있지 않고 현재 수열이 스택 TOP인덱스 가르키는 수열보다 크면
            while (!myStack.isEmpty() && ar[myStack.peek()] < ar[i]) {
                ans[myStack.pop()] = ar[i];  //정답 배열에 오큰수를 현재 수열로 저장하기
            }
            myStack.push(i); //신규데이터 push
        }
```

선형 자료 구조

- 오큰수 구하기
  - ✓ java

```
            while (!myStack.empty()) {
                // 반복문을 다 돌고 나왔는데 스택이 비어있지 않다면 빌 때 까지
                ans[myStack.pop()] = -1;
                // stack에 쌓인 index에 -1을 넣고
            }

            return ans;
        }

        public static void main(String [] args) {
                int [] ar1 = {3, 5, 2, 7};
                int [] ar2 = {9, 5, 4, 8};

                System.out.println(Arrays.toString(solution(ar1)));
                System.out.println(Arrays.toString(solution(ar2)));
        }
    }
```

선형 자료 구조

- 오큰수 구하기
  - ✓ python

```
def solution(ar):
    stack = []

    for i in range(len(ar)):
        # 스택이 비어 있지 않고 현재 수열이 스택 top 인덱스가 가리키는 수열보다 클
     경우
        while stack and ar[stack[-1]] < ar[i]:
            ar[stack.pop()] = ar[i]  # 정답 배열에 오큰수를 현재 수열로 저장하기
        stack.append(i)

    while stack:  # 반복문을 다 돌고 나왔는데 스택이 비어 있지 않다면 빌 때까지
        ar[stack.pop()] = -1  # 스 택에 쌓인 index에 -1을 넣기

    result = ""
    for i in range(len(ar)):
        result += str(ar[i]) + " "

    return result


print(solution([3, 5, 2, 7]))
print(solution([9, 5, 4, 8]))
```

선형 자료 구조

- ❖ 카드 게임
  - ✓ 문제
    - N장의 카드가 있다.
    - 각각의 카드는 차례로 1 에서 N까지의 번호가 붙어 있으며 1번 카드가 가장 위 N번 카드가 가장 아래인 상태로 놓여 있다.
    - 이제 다음과 같은 동작을 카드가 1 장 남을 때까지 반복한다.
      - ◆ 먼저 가장 위에 있는 카드를 바닥에 버리고 그 다음 가장 위에 있는 카드를 가장 아래에 있는 카드 밑으로 옮긴다.
      - ◆ 예를 들어 N = 4일 때를 생각해 보자. 카드는 가장 위에서부터 1,2, 3, 4의 순서대로 놓여 있다. 1 을 버리면 2, 3, 4가 남는다. 여기서 2를 가장 아래로 옮기면 순서가 3, 4, 2가 된다. 3을 버리면 4, 2가 남고, 4를 밑으로 옮기면 순서가 2, 4가 된다. 마지막으로 2를 버리면 카드 4가 남는다.
    - N이 주어졌을 때 가장 마지 막에 남는 카드를 구하는 프로그램을 작성하시오.
    - 입력은 정수 N(1 <= N <= 500000)
    - 예제 입력이 6이면 출력은 4

선형 자료 구조

- 카드 게임
  - ✓ java

```
import java.util.LinkedList;
import java.util.Queue;

public class Solution {
    public static int solution(int s) {
            Queue<Integer> myQueue = new LinkedList<>();
            for (int i = 1; i <= s; i++) { // 카드를 큐에 저장하기
                    myQueue.add(i);
            }
            while (myQueue.size() > 1) { // 카드가 1장 남을 때까지
                    myQueue.poll(); // 맨 위의 카드를 버림
                    myQueue.add(myQueue.poll()); // 맨 위의 카드를 가장 아래 카드
    밑으로 이동
            }
            return myQueue.poll(); // 마지막으로 남은 카드 출력
    }

    public static void main(String [] args) {
            System.out.println(solution(6));
    }
}
```

선형 자료 구조

- 카드 게임
  - ✓ python

```
from collections import deque


def solution(n):
    myqueue = deque()
    for i in range(1, n+1):
        myqueue.append(i)
    while len(myqueue) > 1:      # 카드가 1장 남을 때까지
        myqueue.popleft()        # 맨 위의 카드를 버림
        myqueue.append(myqueue.popleft())   # 맨 위의 카드를 가장 아래 카드 밑으로
     이동
    return myqueue[0]   # 마지막으로 남은 카드 출력


print(solution(6))
```