연습문제

연습문제

- ❖ 문자열 찾기
  - ✓ 문제
    - 총 N개의 문자열로 이뤄진 집합 S가 있을 때 입력으로 주어지는 M개의 문자열 중 집합 S에 포함돼 있는 것 이 총 몇 개인지 구하는 프로그램을 작성
  - ✓ 입력
    - 문자열 배열 2개를 입력
    - 앞의 문자열 배열이 S 이고 뒤의 문자열이 검사해야 하는 문자열
  - ✓ 입출력 예시
    - 입력: [“A”, “B”, “C”, “D”, “E”], [“B”, “C”, “D”, “E”, “F”, “G”]
    - 출력: 4

연습문제

❖ 문자열 찾기

✓ java

```
public class Solution {
    public static int solution(String [] textList, String [] checkList) {
            int count = 0;
            Set<String> set = new HashSet<String>(Arrays.asList(textList));
            for(String str : checkList) {
                    if(set.contains(str)) {
                            count++;
                    }
            }

        return count;
    }

    public static void main(String [] args) {
            String [] textList = {"A", "B", "C", "D", "E"};
            String [] checkList = {"B", "C", "D", "E", "F", "G"};

            System.out.println(solution(textList, checkList));
    }
}
```

연습문제

❖ 문자열 찾기

✓ python

```
def solution(textList, checkList):
    textSet = set(textList)
    count = 0
    for subText in checkList:
        if subText in textSet:
            count += 1
    return count


textList = ["A", "B", "C", "D", "E"]
checkList = ["B", "C", "D", "E", "F", "G"]
print(solution(textList, checkList))
```

연습문제

- 펠린드롬 페어
  - ✓ 문제
    - 문자열 목록에서 words[i] + words[j]가 팰린드롬이 되는 모든 인덱스 조합 (t, j) 를 구하라
  - ✓ 입출력 예시 1
    - 입력: ["abcd", "dcba", "lls", "s", "sssll"]
    - 출력: [[0,1],[1,0],[3,2],[2,4]]
    - 설명: ["abcddcba","dcbaabcd","slls","llssssll"]
  - ✓ 입출력 예시 2
    - 입력: ["bat","tab","cat"]
    - 출력: [[0, 1], [1, 0]]
    - 설명: ["battab","tabbat"]
  - ✓ 입출력 예시 3
    - 입력: ["a",""]
    - 출력: [[0,1],[1,0]]
    - 설명: ["a","a"]

연습문제

❖ 펠린드롬 페어

✓ java

```
public class Solution {
    private static boolean is_palindrome(String word) {
            StringBuilder sb = new StringBuilder(word);
            return word.equals(sb.reverse().toString());
    }
```

연습문제

- 펠린드롬 페어
  - ✓ java

```
public static List<List<Integer>>  solution(String [] word) {
        List<List<Integer>> list = new ArrayList<>();
        for(int i=0; i<word.length; i++) {
                for(int j=0; j<word.length; j++) {
                        if(i == j) {
                                continue;
                        }
                        if(is_palindrome(String.format("%s%s", word[i],
word[j]))) {
                                List<Integer> li = new ArrayList<>();
                                li.add(i);
                                li.add(j);
                                list.add(li);
                        }

                }
        }
    return list;
}
```

연습문제

- 펠린드롬 페어
  - ✓ java

```
public static void main(String [] args) {
        String [] ar1 = {"abcd", "dcba", "lls", "s", "sssll"};
        System.out.println(solution(ar1));
        String [] ar2 = {"bat", "tab", "cat"};
        System.out.println(solution(ar2));
        String [] ar3 = {"a", ""};
        System.out.println(solution(ar3));
    }
}
```

연습문제

❖ 펠린드롬 페어

✓ python

```
def solution(words):
    def is_palindrome(word):
        return word == word[::-1]

    output = []
    for i, word1 in enumerate(words):
        for j, word2 in enumerate(words):
            if i == j:
                continue
            if is_palindrome(word1 + word2):
                output.append([i, j])
    return output


print(solution(["abcd", "dcba", "lls", "s", "sssll"]))
print(solution(["bat", "tab", "cat"]))
print(solution(["a", ""]))
```

연습문제

- ❖ 보석 과 돌
  - ✓ 문제
    - 2개의 문자열을 입력받아서 첫번째 문자열의 각 문자가 두번째 문자열에 몇 개 포함되어 있는지 계산 – 대소문자 구분
    - jewels는 중복된 문자는 존재하지 않음
  - ✓ 입출력 예1
    - 입력: jewels = "aA", stones = "aAAbbbb"
    - 출력: 3
  - ✓ 입출력 예2
    - 입력: jewels = "z", stones = "ZZ"
    - 출력: 0

연습문제

❖ 보석 과 돌

✓ java

```
public class Solution {

    public static int solution(String jewels, String stones) {
        int count=0;

        for(char stone : stones.toCharArray()){
            for(char jewel : jewels.toCharArray()){
                if(stone==jewel)
                            count++;
            }
        }
        return count;
    }


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

연습문제

❖ 보석 과 돌

✓ python

```
def solution(j, s):
    return sum(i in j for i in s)

print(solution("aA", "aAAbbbb"))
```

연습문제

- 중복 없는 가장 긴 부분 문자열
  - ✓ 문제
    - 중복 문자가 없는 가장 긴 부분 문자열의 길이를 리턴
  - ✓ 입출력 예1
    - 입력: "abcabcbb"
    - 출력: 3
  - ✓ 입출력 예2
    - 입력: "bbbbb"
    - 출력: 1
  - ✓ 입출력 예3
    - 입력: "pwwkew"
    - 출력: 3

연습문제

❖ 중복 없는 가장 긴 부분 문자열

✓ java

```
public class Solution {
    public static int solution(String str) {
        Map<Character, Integer> map = new HashMap<>();
        int max_length = 0;
        int start = 0;

        char[] ar = str.toCharArray();
        for(int i=0; i<ar.length; i++) {
            if(map.get(ar[i]) != null && start <= map.get(ar[i])) {
                start = map.get(ar[i]) + 1;
            }else {
                max_length = max_length > i-start+1 ? max_length: i-start+1;

            }
            map.put(ar[i], i);
        }
        return max_length;
    }
```

연습문제

❖ 중복 없는 가장 긴 부분 문자열

✓ java

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

연습문제

❖ 중복 없는 가장 긴 부분 문자열

✓ python

```
def solution(s):
    used = {}
    max_length = start = 0
    for index, char in enumerate(s):
        # 이미 등장했던 문자라면 `start` 위치 갱신
        if char in used and start <= used[char]:
            start = used[char] + 1
        else:  # 최대 부분 문자열 길이 갱신
            max_length = max(max_length, index - start + 1)

        # 현재 문자의 위치 삽입
        used[char] = index

    return max_length


print(solution("abcabcbb"))
print(solution("bbbbb"))
print(solution("pwwkew"))
```

연습문제

- 상위 K 빈도 요소
  - ✓ 문제
    - 상위 k번 이상 등장하는 요소를 추출
  - ✓ 입출력 예1
    - 입력: nums = [1,1,1,2,2,3], k = 2
    - 출력: [1, 2]
  - ✓ 입출력 예2
    - 입력: nums = [1], k = 1
    - 출력: [1]

연습문제

❖ 상위 K 빈도 요소

✓ java

```
public class Solution {
    public static int [] solution(int [] ar, int k) {
     Map<Integer, Integer> map = new HashMap<>();
        for (int num : ar) {
            map.put(num, map.getOrDefault(num, 0) + 1);
        }

        List<Integer>[] answer = new ArrayList[ar.length + 1];
        for (Integer key : map.keySet()) {
            int count = map.get(key);

            if (answer[count] == null) {
               answer[count] = new ArrayList<>();
            }

            answer[count].add(key);
        }
```

연습문제

- 상위 K 빈도 요소
  - ✓ java

```
        List<Integer> list = new ArrayList<>();
        for (int i = answer.length - 1; i >= 0 && list.size() < k; i--) {
            if (answer[i] == null) continue;
            list.addAll(answer[i]);
        }

        // list to array
        int[] res = new int[k];
        for (int i = 0; i < k; i++) {
            res[i] = list.get(i);
        }

        return res;
    }
```

연습문제

❖ 상위 K 빈도 요소

✓ java

```
public static void main(String [] args) {
        int [] ar1 = {1, 1, 1, 2, 2, 3};
        int [] ar2 = {1};
        System.out.println(Arrays.toString(solution(ar1, 2)));
        System.out.println(Arrays.toString(solution(ar2, 1)));
    }
}
```

연습문제

❖ 상위 K 빈도 요소

✓ python

```
import collections

def solution(nums, k):
    return list(zip(*collections.Counter(nums).most_common(k)))[0]


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

연습문제

- 신기한 소수 찾기
  - ✓ 문제
    - 수빈이가 세상에서 가장 좋아하는 것은 소수이고 취미는 소수를 이용해 노는 것
    - 수빈이가 가장 관심 있어 하는 소수는 7331
    - 7331 은 신기하게도 733도 소수, 73도 소수, 7도 소수인데 왼쪽부터 1 자리, 2자리, 3자리, 4자릿수 모두 소수
    - 수빈이는 이런 숫자를 신기한 소수라고 이름 붙였다.
    - 수빈이는 N의 자리의 숫자 중 어떤 수들이 신기한 소수인지 궁금해졌다.
    - 숫자 N이 주어졌을 때 N의 자리 숫자 중 신기한 소수를 모두 찾아보자.
    - 1번째 줄에 N(1 <= N <= 8) 이 주어짐
    - N의 자리 숫자 중 신기한 소수를 찾아서 오름차순 정렬해서 출력

연습문제

- 신기한 소수 찾기
  - ✓ 입력: 4
  - ✓ 출력
    - 2333
    - 2339
    - 2393
    - 2399
    - 2939
    - 3119
    - 3137
    - 3733
    - 3739
    - 3793
    - 3797
    - 5939
    - 7193
    - 7331
    - 7333
    - 7393

연습문제

❖ 신기한 소수 찾기

✓ 에라토스테네스의 체

- 2부터 소수를 구하고자 하는 구간의 모든 수를 나열한다
- 2는 소수이므로 오른쪽에 2를 쓴다
- 자기 자신을 제외한 2의 배수를 모두 지운다.

- 남아있는 수 가운데 3은 소수이므로 오른쪽에 3을 쓴다
- 자기 자신을 제외한 3의 배수를 모두 지운다.

- 남아있는 수 가운데 5는 소수이므로 오른쪽에 5를 쓴다
- 자기 자신을 제외한 5의 배수를 모두 지운다.

- 남아있는 수 가운데 7은 소수이므로 오른쪽에 7을 쓴다. (노란색)
- 자기 자신을 제외한 7의 배수를 모두 지운다.

- 위의 과정을 반복하면 구하는 구간의 모든 소수가 남는다.

연습문제

- ❖ 신기한 소수 찾기
  - ✓ 해결
    - 소수는 약수가 1과 자기 자신인 수
    - 4는 약수가 1, 2, 4이므로 소수가 아니 고 7은 1,7이므로 소수
    - 우선 자릿수가 한 개인 소수는 2, 3, 5, 7이므로 이 수부터 탐색을 시작
    - 4, 6, 8, 9를 제외한 가지치기 방식을 적용한 것
    - 자릿수가 두 개인 현재 수 * 10 + a를 계산하여 이 수가 소수인지 판단하고 소수라면 재귀 함수로 자릿수를 하나 늘리는데 a가 짝수 인 경우는 항상 2를 약수로 가지므로 가지 치기로 a가 짝수인 경우를 제외
    - 위의 방식으로 자릿수를 n 까지 확장했을 때 그 값이 소수라면 해당 값을 출력

연습문제

❖ 신기한 소수 찾기

✓ java

```
public class Solution {

    public static void main(String[] args) {
        // 일의 자리 소수는 2 3 5 7 이므로 4개 수에서만 시작
        solution(4);

    }

    private static void solution(int input) {
        DFS(2, 1, input);
        DFS(3, 1, input);
        DFS(5, 1, input);
        DFS(7, 1, input);
    }
```

연습문제

❖ 신기한 소수 찾기

✓ java

```
//전체 탐색
private static void DFS(int number, int jarisu, int input) {
        if (jarisu == input) {
                if (isPrime(number)) {
                        System.out.println(number);
                }
                return;
        }

        for (int i = 1; i < 10; i++) {
                if (i % 2 == 0) { // 짝수이면 더 이상 탐색할 필요가 없음
                        continue;
                }
                if (isPrime(number * 10 + i)) {
                        // 소수이면 재귀로 자리수를 늘려갑니다.
                        DFS(number * 10 + i, jarisu + 1, input);
                }
        }

}
```

연습문제

❖ 신기한 소수 찾기

✓ java

```
//소수 구하는 메서드
private static boolean isPrime(int num) {
        for (int i = 2; i <= num / 2; i++)
                if (num % i == 0)
                        return false;
        return true;
}
}
```

연습문제

❖ 신기한 소수 찾기

✓ python

```
def isPrime(num):
    for i in range(2, int(num / 2 + 1)):
        if num % i == 0:
            return False
    return True


def DFS(number, input):
    if len(str(number)) == input:
        print(number)
    else:
        for i in range(1, 10):
            if i % 2 == 0:  # 짝수라면 더 이상 탐색 불필요
                continue
            if isPrime(number * 10 + i):    # 소수이면 자릿수 늘림
                DFS(number * 10 + i, input)
```

연습문제

- 신기한 소수 찾기
  - ✓ python

```
def solution(input):
    DFS(2, input)
    DFS(3, input)
    DFS(5, input)
    DFS(7, input)

solution(4)
```

연습문제

❖ 입국 심사

✓ 문제

- n명이 입국심사를 위해 줄을 서서 기다리고 있습니다. 각 입국심사대에 있는 심사관마다 심사하는데 걸리는 시간은 다릅니다.
- 처음에 모든 심사대는 비어있습니다. 한 심사대에서는 동시에 한 명만 심사를 할 수 있습니다. 가장 앞에 서 있는 사람은 비어 있는 심사대로 가서 심사를 받을 수 있습니다. 하지만 더 빨리 끝나는 심사대가 있으면 기다렸다가 그곳으로 가서 심사를 받을 수도 있습니다.
- 모든 사람이 심사를 받는데 걸리는 시간을 최소로 하고 싶습니다.
- 입국심사를 기다리는 사람 수 n, 각 심사관이 한 명을 심사하는데 걸리는 시간이 담긴 배열 times가 매개변수로 주어질 때, 모든 사람이 심사를 받는데 걸리는 시간의 최솟값을 return 하도록 solution 함수를 작성해주세요.

✓ 제한사항

- 입국심사를 기다리는 사람은 1명 이상 1,000,000,000명 이하입니다.
- 각 심사관이 한 명을 심사하는데 걸리는 시간은 1분 이상 1,000,000,000분 이하입니다.
- 심사관은 1명 이상 100,000명 이하입니다.

연습문제

- 입국 심사
  - ✓ 입출력 예
    - 입력 N – 6
    - 입력 times – [7, 10]
    - 출력: 28

연습문제

- 입국 심사
  - ✓ java

```
public class Solution {
    private static boolean isValid(long t, int n, int[] times) {
        long c = 0;
        for (int time : times) {
            c += t / time;
        }
        return c >= n;
    }
```

연습문제

- 입국 심사
  - ✓ java

```
private static long solution(int n, int[] times) {
    long start = 1;
    long end = 10000000000000000000L;

    while (end > start) {
        long t = (start + end) / 2;

        if (isValid(t, n, times)) {
            end = t;
        } else {
            start = t + 1;
        }
    }

    return start;
}
```

연습문제

- 입국 심사
  - ✓ java

```
        public static void main(String [] args) {
         int N = 6;
         int [] times = {7, 10};
         System.out.println(solution(N, times));
        }
    }
```

연습문제

❖ 입국 심사

✓ python

```
def solution(n, times):
    answer = 0
    left, right = 1, max(times) * n

    while left <= right:
        mid = (left + right) // 2
        people = 0
        for time in times:
            people += mid // time
            if people >= n: break

        if people >= n:
            answer = mid
            right = mid - 1

        elif people < n:
            left = mid + 1

    return answer

print(solution(6, [7, 10]))
```

연습문제

❖ 순위

✓ 문제 설명

- n명의 권투선수가 권투 대회에 참여했고 각각 1번부터 n번까지 번호를 받았습니다.
- 권투 경기는 1대1 방식으로 진행이 되고 만약 A 선수가 B 선수보다 실력이 좋다면 A 선수는 B 선수를 항상 이깁니다. 심판은 주어진 경기 결과를 가지고 선수들의 순위를 매기려 합니다. 하지만 몇몇 경기 결과를 분실하여 정확하게 순위를 매길 수 없습니다.
- 선수의 수 n, 경기 결과를 담은 2차원 배열 results가 매개변수로 주어질 때 정확하게 순위를 매길 수 있는 선수의 수를 return 하도록 solution 함수를 작성해주세요.

✓ 제한사항

- 선수의 수는 1명 이상 100명 이하입니다.
- 경기 결과는 1개 이상 4,500개 이하입니다.
- results 배열 각 행 [A, B]는 A 선수가 B 선수를 이겼다는 의미입니다.
- 모든 경기 결과에는 모순이 없습니다.

연습문제

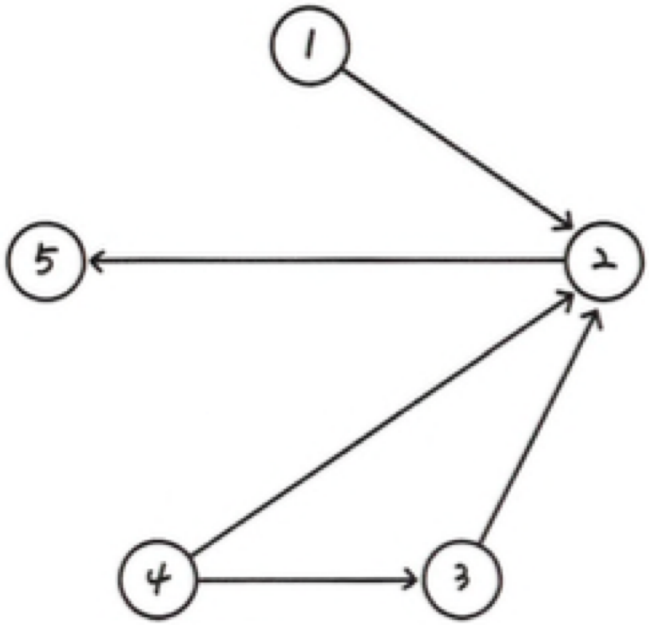
- 순위
  - ✓ 입출력 예
    - 입력 n: 5
    - 입력 results: [[4, 3], [4, 2], [3, 2], [1, 2], [2, 5]]
    - 출력: 2
  - ✓ 설명
    - 2번 선수는 [1, 3, 4] 선수에게 패배했고 5번 선수에게 승리했기 때문에 4위입니다.
    - 5번 선수는 4위인 2번 선수에게 패배했기 때문에 5위입니다.

연습문제

- ❖ 순위
  - ✓ 풀이
    - 선수 A가 선수 B를 이겼다는 것은 방향이 있는 정보
    - 경기 결과는 방향 그래프로 나타낼 수 있음
    - 선수 수가 100명 이하이기 때문에 인접 행렬 방식으로 그래프를 충분히 표현할 수 있음
    - 한 선수의 경기 결과를 정하려면 해당 선수를 이긴 선수들과 해당 선수한테 진 선수들을 모두 결정해야 함
    - (이긴 선수들의 수 + 진 선수들의 수 + 1)이 전체 선수의 수가 되는지 검사해서 알 수 있음
    - 이긴 선수들의 수와 진 선수들의 수를 알아낼 수 있으면 문제는 해결됨

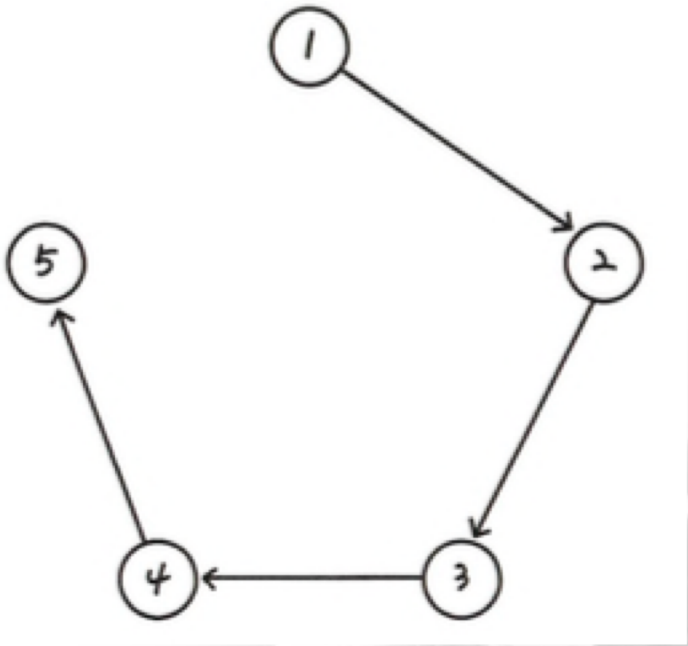
연습문제

- ❖ 순위
  - ✓ 풀이
    - [[4, 3], [4, 2], [3, 2], [1, 2], [2, 5]]



연습문제

- ❖ 순위
  - ✓ 풀이
    - ● [[1,2], [2,3], [3,4], [4,5]]: 경기를 전부하지 않아도 순위가 결정되는 경우
      - ◆ 경기 결과에서 각 선수의 경기 결과는 하나밖에 남지 않았지만 모두 순위가 결정되는데 입력으로 들어온 경기 결과로 그래프를 구성하고 재귀를 이용하여 화살표를 따라가면서 이긴 선수 수와 진 선수 수를 세야 함



연습문제

❖ 순위

✓ Java

```
public class Solution {
    private static int countForward(
            int u, boolean[][] graph, boolean[] isVisited) {
        int count = 1;

        for (int v = 0; v < graph[u].length; v++) {
            if (!graph[u][v] || isVisited[v]) continue;
            isVisited[v] = true;
            count += countForward(v, graph, isVisited);
        }

        return count;
    }
```

연습문제

❖ 순위

✓ Java

```
private static int countBackward(
        int u, boolean[][] graph, boolean[] isVisited) {
    int count = 1;

    for (int v = 0; v < graph.length; v++) {
        if (!graph[v][u] || isVisited[v]) continue;
        isVisited[v] = true;
        count += countBackward(v, graph, isVisited);
    }

    return count;
}
```

연습문제

❖ 순위

✓ Java

```
public static int solution(int n, int[][] results) {
    boolean[][] graph = new boolean[n][n];
    for (int[] edge : results) {
        int u = edge[0] - 1;
        int v = edge[1] - 1;
        graph[u][v] = true;
    }

    int count = 0;
    for (int u = 0; u < n; u++) {
        int wins = countForward(u, graph, new boolean[n]) - 1;
        int loses = countBackward(u, graph, new boolean[n]) - 1;
        if (wins + loses + 1 == n) {
            count++;
        }
    }

    return count;
}
```

연습문제

❖ 순위

✓ Java

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

연습문제

❖ 순위

✓ Python

```
from collections import defaultdict
def solution(n, results):
    answer = 0
    win = defaultdict(set)
    lose =  defaultdict(set)

    for winner, loser in results:
        win[loser].add(winner)
        lose[winner].add(loser)

    for i in range(1, n + 1):
        for winner in win[i]:
            lose[winner].update(lose[i])
        for loser in lose[i]:
            win[loser].update(win[i])

    for i in range(1, n + 1):
        if len(win[i]) + len(lose[i]) == n - 1:
            answer += 1

    return answer
```

연습문제

❖ 순위

✓ Python

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

연습문제

❖ 보석 쇼핑

✓ 문제

- 개발자 출신으로 세계 최고의 갑부가 된 어피치는 스트레스를 받을 때면 이를 풀기 위해 오프라인 매장에 쇼핑을 하러 가곤 합니다.
- 어피치는 쇼핑을 할 때면 매장 진열대의 특정 범위의 물건들을 모두 싹쓸이 구매하는 습관이 있습니다.
- 어느 날 스트레스를 풀기 위해 보석 매장에 쇼핑을 하러 간 어피치는 이전처럼 진열대의 특정 범위의 보석을 모두 구매하되 특별히 아래 목적을 달성하고 싶었습니다.
- 진열된 모든 종류의 보석을 적어도 1개 이상 포함하는 가장 짧은 구간을 찾아서 출력

연습문제

❖ 보석 쇼핑

✓ 문제

- 예를 들어 아래 진열대는 4종류의 보석(RUBY, DIA, EMERALD, SAPPHIRE) 8개가 진열된 예시

진열대	1	2	3	4	5	6	7	8
보석	DIA	RUBY	RUBY	DIA	DIA	EMERALD	SAPPHIRE	DIA

◆ 진열대의 3번부터 7번까지 5개의 보석을 구매하면 모든 종류의 보석을 적어도 하나 이상씩 포함하게 됩니다.

◆ 진열대의 3, 4, 6, 7번의 보석만 구매하는 것은 중간에 특정 구간(5번)이 빠지게 되므로 어피치의 쇼핑 습관에 맞지 않습니다.

◆ 진열대 번호 순서대로 보석들의 이름이 저장된 배열 gems가 매개변수로 주어집니다. 이때 모든 보석을 하나 이상 포함하는 가장 짧은 구간을 찾아서 return 하도록 solution 함수를 완성해주세요.

◆ 가장 짧은 구간의 시작 진열대 번호와 끝 진열대 번호를 차례대로 배열에 담아서 return 하도록 하며, 만약 가장 짧은 구간이 여러 개라면 시작 진열대 번호가 가장 작은 구간을 return 합니다.

연습문제

❖ 보석 쇼핑

✓ 제한 사항

- gems 배열의 크기는 1 이상 100,000 이하입니다.
- gems 배열의 각 원소는 진열대에 나열된 보석을 나타냅니다.
- gems 배열에는 1번 진열대부터 진열대 번호 순서대로 보석이름이 차례대로 저장되어 있습니다.
- gems 배열의 각 원소는 길이가 1 이상 10 이하인 알파벳 대문자로만 구성된 문자열입니다.

연습문제

- 보석 쇼핑
  - ✓ 입출력 예

gems	result
["DIA", "RUBY", "RUBY", "DIA", "DIA", "EMERALD", "SAPPHIRE", "DIA"]	[3, 7]
["AA", "AB", "AC", "AA", "AC"]	[1, 3]
["XYZ", "XYZ", "XYZ"]	[1, 1]
["ZZZ", "YYY", "NNNN", "YYY", "BBB"]	[1, 5]

연습문제

❖ 보석 쇼핑

✓ Java

```
public class Solution {
    public static int[] solution(String[] gems) {
        int start = 0;
        int end = gems.length - 1;

        Set<String> gemSet = new HashSet<>(List.of(gems));

        int s = 0;
        int e = s;
        Map<String, Integer> includes = new HashMap<>();
        includes.put(gems[s], 1);
```

연습문제

❖ 보석 쇼핑

✓ Java

```
        while (s < gems.length) {
            if (includes.keySet().size() == gemSet.size()) {
                if (e - s < end - start) {
                    start = s;
                    end = e;
                }
                includes.put(gems[s], includes.get(gems[s]) - 1);
                if (includes.get(gems[s]) == 0) {
                    includes.remove(gems[s]);
                }
                s++;
            } else if (e < gems.length - 1) {
                e++;
                includes.put(gems[e],
                        includes.getOrDefault(gems[e], 0) + 1);
            } else {
                break;
            }
        }
        return new int[] {start + 1, end + 1};
    }
```

연습문제

❖ 보석 쇼핑

✓ Java

```
public static void main(String [] args) {
 String [] gems1 = {"DIA", "RUBY", "RUBY", "DIA", "DIA", "EMERALD", "SAPPHIRE",
 "DIA"};
 System.out.println(Arrays.toString(solution(gems1)));
 String [] gems2 = {"AA", "AB", "AC", "AA", "AC"};
 System.out.println(Arrays.toString(solution(gems2)));
 String [] gems3 = {"XYZ", "XYZ", "XYZ"};
 System.out.println(Arrays.toString(solution(gems3)));
 String [] gems4 = {"ZZZ", "YYY", "NNNN", "YYY", "BBB"};
 System.out.println(Arrays.toString(solution(gems4)));
}
}
```

연습문제

❖ 보석 쇼핑

✓ Python

```
def solution(gems):
    kind = len(set(gems))
    size = len(gems)
    answer = [0, size - 1]

    dic = {gems[0]: 1}
    start = end = 0

    while end < size:
        if len(dic) < kind:
            end += 1
            if end == size: break
            dic[gems[end]] = dic.get(gems[end], 0) + 1
        else:
            if (end - start + 1) < (answer[1] - answer[0] + 1): answer = [start, end]
            if dic[gems[start]] == 1:
                del dic[gems[start]]
            else:
                dic[gems[start]] -= 1
            start += 1
```

연습문제

❖ 보석 쇼핑

✓ Python

```
        answer[0] += 1
        answer[1] += 1

        return answer
```

연습문제

❖ 보석 쇼핑

✓ Python

```
print(solution(["DIA", "RUBY", "RUBY", "DIA", "DIA", "EMERALD", "SAPPHIRE", "DIA"]))
print(solution(["AA", "AB", "AC", "AA", "AC"]))
print(solution(["XYZ", "XYZ", "XYZ"]))
print(solution(["ZZZ", "YYY", "NNNN", "YYY", "BBB"]))
```

연습문제

❖ 섬 연결하기

✓ 문제

- n개의 섬 사이에 다리를 건설하는 비용(costs)이 주어질 때, 최소의 비용으로 모든 섬이 서로 통행 가능하도록 만들 때 필요한 최소 비용을 return 하도록 solution을 완성하세요.
- 다리를 여러 번 건너더라도, 도달할 수만 있으면 통행 가능하다고 봅니다. 예를 들어 A 섬과 B 섬 사이에 다리가 있고, B 섬과 C 섬 사이에 다리가 있으면 A 섬과 C 섬은 서로 통행 가능합니다.

✓ 제한사항

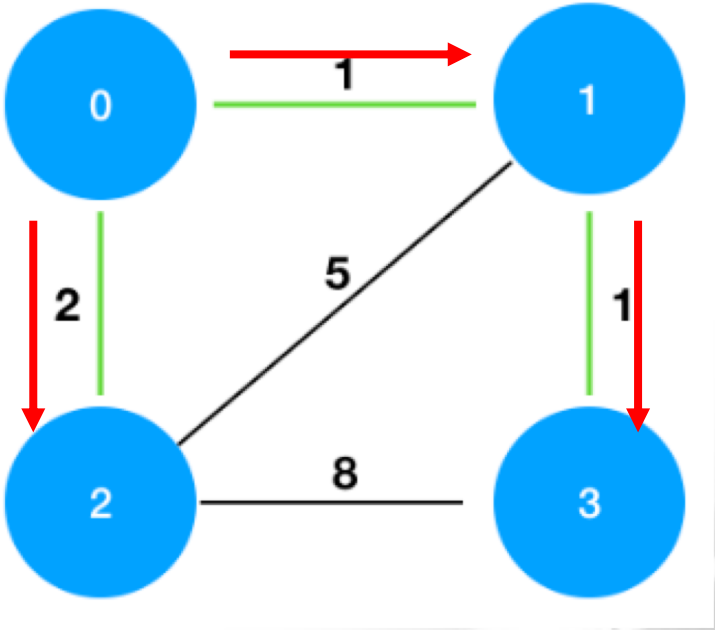
- 섬의 개수 n은 1 이상 100 이하입니다.
- costs의 길이는 ((n-1) * n) / 2이하입니다.
- 임의의 i에 대해, costs[i][0] 와 costs[i] [1]에는 다리가 연결되는 두 섬의 번호가 들어있고, costs[i] [2]에는 이 두 섬을 연결하는 다리를 건설할 때 드는 비용입니다.
- 같은 연결은 두 번 주어지지 않습니다. 또한 순서가 바뀌더라도 같은 연결로 봅니다. 즉 0과 1 사이를 연결하는 비용이 주어졌을 때, 1과 0의 비용이 주어지지 않습니다.
- 모든 섬 사이의 다리 건설 비용이 주어지지 않습니다. 이 경우, 두 섬 사이의 건설이 불가능한 것으로 봅니다.
- 연결할 수 없는 섬은 주어지지 않습니다.

연습문제

❖ 섬 연결하기

✓ 입출력 예

n	costs	return
4	[[0,1,1],[0,2,2],[1,2,5],[1,3,1],[2,3,8]]	4



연습문제

❖ 섬 연결하기

✓ 해결

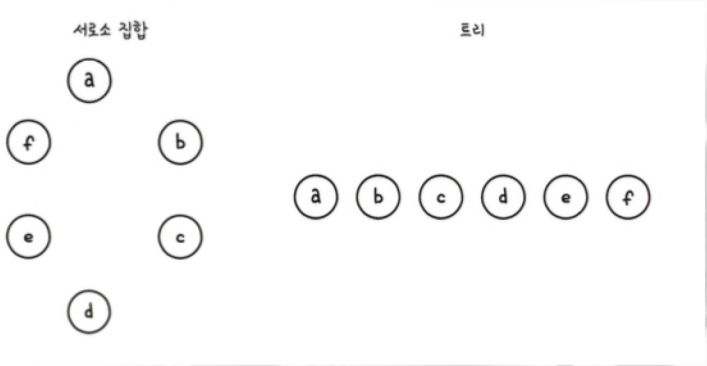
- 그래프에 포함된 정점을 최소 비용으로 모두 연결하는 문제
- 문제에서 이미 연결되어 있는 두 정점을 연결하는 것은 무의미하므로 이전에 연결되지 않은 정점들을 계속해서 연결
- 이렇게 계속 연결하다 보면 결국 모든 정점이 연결되고 가장 효율적인 방식으로 연결했기 때문 에 두 정점 사이의 경로는 하나만 존재
- 이렇게 해서 구성된 정점들과 간선들은 트리 모양을 이루는데 간선에 가중치가 있는 그래프에서 간선의 가중치 합이 최소가 되는 트리가 최소 신장 트리(minimum spanning tree)
- 간선을 가중치 순으로 정렬한 후 순서대로 순회하면서 연결되지 않은 두 정점을 잇는 간선을 채택해 나가면 됨
- 이 방식으로 최소 신장 트리를 구하는 것이 크루스칼 알고리즘(kruskal algorithm)
- 크루스칼 알고리즘은 두 정점이 연결되었는지 검사하기 위해 유니온 파인드를 활용

연습문제

❖ 섬 연결하기

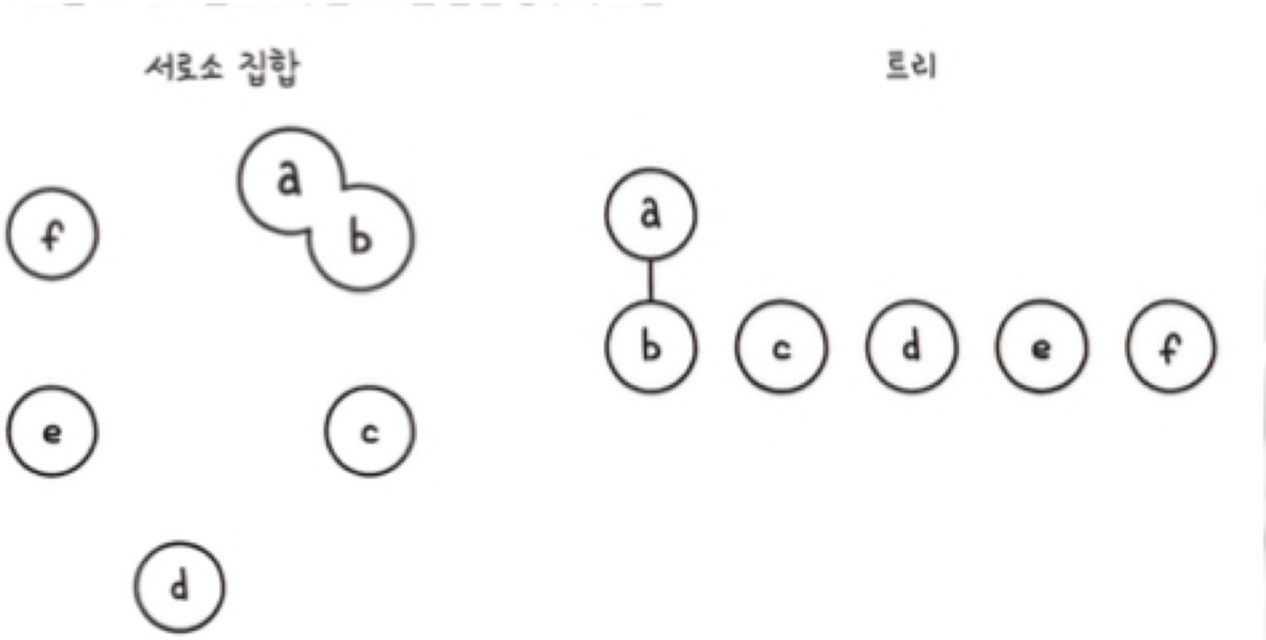
✓ Union Find

- 유니온 파인드(union find)는 서로소 집합(disjoint set) 자료 구조를 사용하는 알고리즘
- 서로소 집합은 공통된 원소를 가지고 있지 않은 2개 이상의 집합으로 유니온 파인드에서는 모든 원소가 자신만 들어있는 집합에 속한 상태로 시작
- 이후 원소들이 서로 연결되는 정보를 이용하여 집합이 점점 합쳐짐
- 서로소 집합의 구현은 트리를 이용
- 집합 내 각 원소들은 트리에서 노드로 표현되며 트리에서 루트 노드가 같다면 같은 서로소 집합에 속함
- 6개의 a, b. c, d, e. f 원소가 있다고 하면 처음에 이 원소들은 각각의 집합에 속해 있음
- 이는 모든 원소가 별도의 트리를 구성하는 것으로 표현



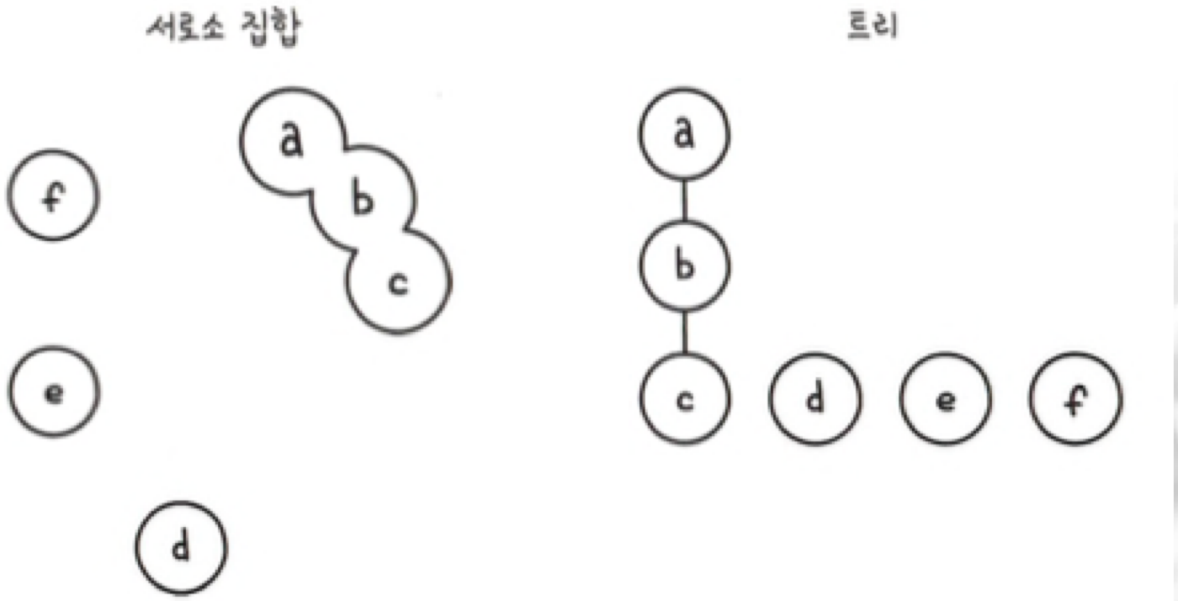
연습문제

- ❖ 섬 연결하기
  - ✓ Union Find
    - 원소 a와 원소 b를 연결
    - 서로소 집합에서는 두 원소를 하나의 집합에 넣는데 이는 트리에서 한 원소가 다른 원소의 자식 노드로 들어가게 해서 표현



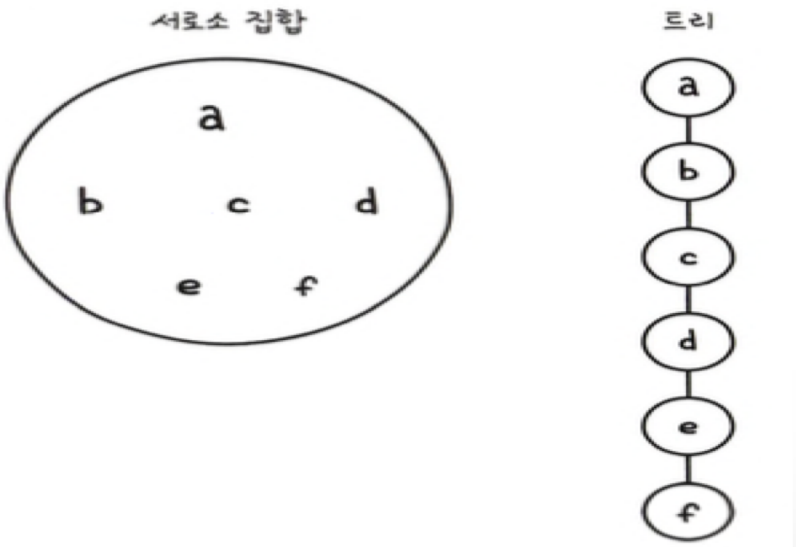
연습문제

- 섬 연결하기
  - ✓ Union Find
    - 트리의 표현에서 노드 a와 노드 b는 모두 루트 노드가 a이므로 같은 서로소 집합에 속해 있음을 알 수 있음
    - 노드 b와 노드 c를 합치는데 노드 b는 이미 부모 노드를 가지고 있으므로 노드 c의 자식 노드가 될 수 없으므로 노드 c가 노드 b의 자식 노드가 될 것



연습문제

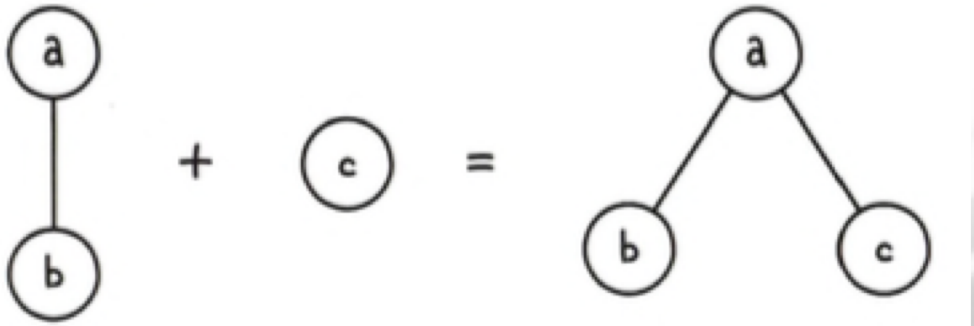
- ❖ 섬 연결하기
  - ✓ Union Find
    - ● 최악의 경우



- ◆ 이 경우 모든 노드의 루트 노드는 a로 같기 때문에 서로소 집합은 잘 나타내지만 실제로 같은 집합인지 검사하기 위해서는 원소 개수에 비례하는 시간 복잡도가 소요되므로 두 원소를 합칠 때 그저 한 트리의 루트를 노드의 자식으로 붙이기보다 더욱 효율적인 방식으로 합쳐야 함

연습문제

- 섬 연결하기
  - ✓ Union Find
    - 가장 간단하고 유용한 방식은 트리의 깊이를 이용하는 것
    - 트리의 깊이가 더 얕은 트리 의 루트 노드를 다른 쪽 트리에 있는 루트 노드의 자식 노드로 이어줌
    - 이 방식대로 노드 a와 노드 b가 이어진 상태의 서로소 집합에 새로 노드 c를 추가하면 노드 a와 노드 b가 연결된 트리의 깊이는 2고 노드 c만 들어 있는 트리의 깊이는 1이므로 노드 c가 루트 노드인 노드 a의 자식으로 들어갑니다.



연습문제

- ❖ 섬 연결하기
  - ✓ Union Find
    - ● 이 방식대로 트리를 구성하면 트리의 깊이는 합친 두 트리의 깊이가 같을 때만 증가하게 되서 전체 원소 개수가 N일 때 O(logN)의 시간 복잡도 만에 두 원소가 같은 서로소 집합에 있는지 검사할 수 있음

연습문제

❖ 섬 연결하기

✓ Java

```
public class Solution {
    private static class Node {
        private int depth = 1;
        private Node parent = null;

        public boolean isConnected(Node o) {
            return root() == o.root();
        }
```

연습문제

❖ 섬 연결하기

✓ Java

```
    public void merge(Node o) {
        if (isConnected(o)) return;

        Node root1 = root();
        Node root2 = o.root();

        if (root1.depth > root2.depth) {
            root2.parent = root1;
        } else if (root1.depth < root2.depth) {
            root1.parent = root2;
        } else {
            root2.parent = root1;
            root1.depth += 1;
        }
    }

    private Node root() {
        if (parent == null) return this;
        return parent.root();
    }
}
```

연습문제

- 섬 연결하기
  - ✓ Java

```
private static class Edge {
    public final int u;
    public final int v;
    public final int cost;

    private Edge(int u, int v, int cost) {
        this.u = u;
        this.v = v;
        this.cost = cost;
    }
}
```

연습문제

❖ 섬 연결하기

✓ Java

```
private static int solution(int n, int[][] costs) {
    Edge[] edges = Arrays.stream(costs)
            .map(edge -> new Edge(edge[0], edge[1], edge[2]))
            .sorted(Comparator.comparingInt(e -> e.cost))
            .toArray(Edge[]::new);
    Node[] nodes = new Node[n];
    for (int i = 0; i < n; i++) {
        nodes[i] = new Node();
    }
    int totalCost = 0;
    for (Edge edge : edges) {
        Node node1 = nodes[edge.u];
        Node node2 = nodes[edge.v];

        if (node1.isConnected(node2)) continue;
        node1.merge(node2);
        totalCost += edge.cost;
    }
    return totalCost;
}
```

연습문제

❖ 섬 연결하기

✓ Java

```
        public static void main(String [] args) {
         int n = 4;
         int [][] costs = {{0,1,1}, {0,2,2}, {1,2,5}, {1,3,1}, {2,3,8}};
         System.out.println(solution(n, costs));
        }
    }
```

연습문제

❖ 섬 연결하기

✓ Python

```
def union(parent, x, y):
    x = find(parent, x)
    y = find(parent, y)
    if x == y: return
    parent[x] = y


def find(parent, x):
    if parent[x] == x: return x
    parent[x] = find(parent, parent[x])
    return parent[x]
```

연습문제

❖ 섬 연결하기

✓ Python

```
def solution(n, costs):
    answer = cnt = 0
    parent = [i for i in range(n)]
    costs = sorted(costs, key=lambda x: x[2])

    for i in range(len(costs)):
        if find(parent, costs[i][0]) != find(parent, costs[i][1]):
            union(parent, costs[i][0], costs[i][1])
            answer += costs[i][2]
            cnt += 1
        if cnt == n: break

    return answer

print(solution(4, [[0,1,1],[0,2,2],[1,2,5],[1,3,1],[2,3,8]]))
```

연습문제

- 생성되는 전화번호 전화 번호의 개수 구하기
  - ✓ 문제
    - 누를 수 있는 버튼이 배열로 제공될 때 나올 수 있는 모든 전화번호(문자의 조합)를 출력

연습문제

- 생성되는 전화번호 전화 번호의 개수 구하기
  - ✓ 입출력 예
    - 입력 “23” -> 출력 ["ad","ae","af","bd","be","bf","cd","ce","cf"]
    - 입력 “” -> 출력 []
    - 입력 “2” -> 출력["a","b","c"]
  - ✓ 제약 조건
    - 입력의 길이는 0보다 크거나 같고 4보다 작거나 같음
    - 각 문자는 2 부터 9까지

연습문제

❖ 생성되는 전화번호 전화 번호의 개수 구하기

✓ Java

```
public class Solution {
    public static List<String> solution(String digits) {
    //각 번호에 해당하는 문자들을 Map에 저장
    Map<String, List<String>> map = new HashMap<>();
      {
          map.put("2", List.of("a", "b", "c"));
          map.put("3", List.of("d", "e", "f"));
          map.put("4", List.of("g", "h", "i"));
          map.put("5", List.of("j", "k", "l"));
          map.put("6", List.of("m", "n", "o"));
          map.put("7", List.of("p", "q", "r", "s"));
          map.put("8", List.of("t", "u", "v"));
          map.put("9", List.of("w", "x", "y", "z"));
      }
```

연습문제

- 생성되는 전화번호 전화 번호의 개수 구하기
  - ✓ Java

```
List<String> answer = new ArrayList<>();
//문자가 없을 때는 빈 List 리턴
if (digits.length() == 0) {
    return answer;
}
//문자가 1개 일 때는 Map에서 찾아서 리턴
else if (digits.length() == 1) {
    return map.get(digits);
}
```

연습문제

❖ 생성되는 전화번호 전화 번호의 개수 구하기

✓ Java

```
        //첫번째 문자 가져오기
        String digit = digits.substring(0, 1);
        //첫번째 문자에 해당하는 리스트 가져오기
        List<String> str = map.get(digit);
        //이후 문자들을 동일한 방법으로 가져오기
        List<String> rightStr = solution(digits.substring(1));
        //가져온 문자들을 이어붙이기
        for (int i = 0; i < str.size(); i++) {
            String s = str.get(i);
            for (int j = 0; j < rightStr.size(); j++) {
                String rs = rightStr.get(j);
                answer.add(s + rs);
            }
        }
        return answer;
    }
```

연습문제

- 생성되는 전화번호 전화 번호의 개수 구하기
  - ✓ Java

```
        public static void main(String [] args) {
         System.out.println(solution("23"));
         System.out.println(solution(""));
         System.out.println(solution("2"));
        }
    }
```

연습문제

❖ 생성되는 전화번호 전화 번호의 개수 구하기

✓ Python

```
def solution(digits):
    def dfs(index, path):
        # 끝까지 탐색하면 백트래킹
        if len(path) == len(digits):
            result.append(path)
            return
        # 입력값 자릿수 단위 반복
        for i in range(index, len(digits)):
            # 숫자에 해당하는 모든 문자열 반복
            for j in dic[digits[i]]:
                dfs(i + 1, path + j)

    # 예외 처리
    if not digits:
        return []
    dic = {"2": "abc", "3": "def", "4": "ghi", "5": "jkl",
           "6": "mno", "7": "pqrs", "8": "tuv", "9": "wxyz"}
    result = []
    dfs(0, "")
    return result
```

연습문제

- 생성되는 전화번호 전화 번호의 개수 구하기
  - ✓ Python

```
print(solution("23"))
print(solution(""))
print(solution("2"))
```

연습문제

- 순열
  - ✓ 문제
    - 서로 다른 정수 배열을 입력받아서 가능한 모든 순열을 출력
  - ✓ 입출력 예1
    - 입력: [1, 2, 3]
    - 출력: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]
  - ✓ 입출력 예2
    - 입력: [0,1]
    - 출력: [[0,1],[1,0]]

연습문제

❖ 순열

✓ Java

```
public class Solution {
    private static void backtracking(List<List<Integer>> rst, int[] nums, List<Integer>
    list) {
            if (list.size() == nums.length) {
                    rst.add(new ArrayList<>(list));
                    return;
            }

            for (int i : nums) {
                    // 더 탐색할 하위 노드 탐색
                    if (!list.contains(i)) {
                            list.add(i);
                            backtracking(rst, nums, list);
                            list.remove(list.size() - 1);
                    }
            }
    }
```

연습문제

❖ 순열

✓ Java

```
public static List<List<Integer>> solution(int[] nums) {
        List<List<Integer>> rst = new ArrayList<>();

        backtracking(rst, nums, new ArrayList<Integer>());
        return rst;
}
```

연습문제

❖ 순열

✓ Java

```
        public static void main(String [] args) {
                int [] nums1 = {1, 2, 3};
                List<List<Integer>> answer = solution(nums1);
                for(List<Integer> imsi : answer) {
                        System.out.println(imsi);
                }

                int [] nums2 = {0,1};
                answer = solution(nums2);
                for(List<Integer> imsi : answer) {
                        System.out.println(imsi);
                }
        }
    }
```

연습문제

❖ 순열

✓ Python – itertools를 사용 가능

```
import itertools


def solution(nums):
    return list(itertools.permutations(nums))


print(solution([1,2,3]))
print(solution([0, 1]))
```

연습문제

❖ 순열

✓ Python – itertools를 사용 불가능

```
def solution(nums):
    results = []
    prev_elements = []
    def dfs(elements):
        # 리프 노드일때 결과 추가
        if len(elements) == 0:
            results.append(prev_elements[:])

        # 순열 생성 재귀 호출
        for e in elements:
            next_elements = elements[:]
            next_elements.remove(e)

            prev_elements.append(e)
            dfs(next_elements)
            prev_elements.pop()

    dfs(nums)
    return results
```

연습문제

- 순열
  - ✓ Python – itertools를 사용 불가능

```
print(solution([1,2,3]))
print(solution([0, 1]))
```

연습문제

- 조합
  - ✓ 문제
    - 전체 수 n을 입력받아 k개의 조합을 리턴
  - ✓ 입출력 예1
    - 입력: n = 4, k = 2
    - 출력: [[1,2],[1,3],[1,4],[2,3],[2,4],[3,4]]
  - ✓ 입출력 예2
    - 입력: n = 1, k = 1
    - 출력: [1]
  - ✓ 제약 조건
    - 1 <= n <= 20
    - 1 <= k <= n

연습문제

❖ 조합

✓ java

```
public class Solution {
    public static List<List<Integer>> solution(int n, int k) {
        List<List<Integer>> result = new ArrayList<List<Integer>>();
        backTracking(result, new ArrayList<Integer>(), 1, n, k);
        return result;
    }

    public static void backTracking(List<List<Integer>> result, List<Integer> comb, int
    start, int n, int level) {
        if (level == 0) { //ending case
            result.add(new ArrayList<Integer>(comb));
            return;
        }

        for (int i = start; i <= n-level+1; i++) {
            comb.add(i);
            backTracking(result, comb, i+1, n, level-1);
            comb.remove(comb.size() - 1);
        }
    }
```

연습문제

❖ 조합

✓ java

```
public static void main(String [] args) {
        List<List<Integer>> answer = solution(4, 2);
        for(List<Integer> imsi : answer) {
                System.out.println(imsi);
        }

        answer = solution(1, 1);
        for(List<Integer> imsi : answer) {
                System.out.println(imsi);
        }
    }
}
```

연습문제

❖ 조합

✓ python

```
def solution(n, k):
    results = []

    def dfs(elements, start: int, k: int):
        if k == 0:
            results.append(elements[:])
            return

        # 자신 이전의 모든 값을 고정하여 재귀 호출
        for i in range(start, n + 1):
            elements.append(i)
            dfs(elements, i + 1, k - 1)
            elements.pop()

    dfs([], 1, k)
    return results


print(solution(4, 2))
print(solution(1, 1))
```