스택(Stack)

1. 스택(Stack)의 개념

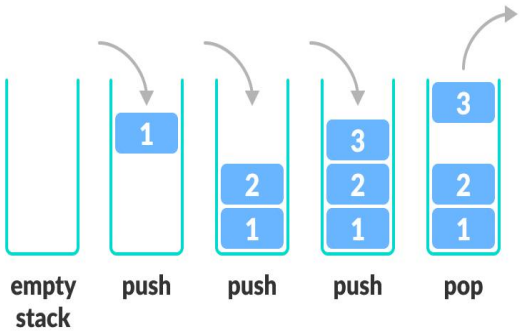
한 쪽 끝에서만 자료를 넣고 뺄 수 있는 LIFO(Last In First Out) 형식의 자료 구조이다.
즉, 가장 최근에 스택에 추가한 항목이 가장 먼저 제거되는 자료구조이다.

2. 스택(Stack)의 연산(자바 기준)

1) push(item): item 하나를 스택의 가장 윗 부분에 추가한다.
2) pop(): 스택에서 가장 위에 있는 항목을 제거한다.
3) peek(): 스택의 가장 위에 있는 항목을 반환한다.
4) empty(): 스택이 비어 있을 때에 true를 반환한다.

3. 자바에서 스택사용하기

```
public void solution(){
    Stack<Integer> stack = new Stack<>();
    stack.push(1);
    stack.push(2);
    stack.push(3);
    stack.pop();
    System.out.println(stack.isEmpty());
    System.out.println(stack.peek());
    stack.pop();
    System.out.println(stack.size());
    stack.pop();
    System.out.println(stack.isEmpty());
}
```



큐(Queue)

1. 큐(Queue)의 개념

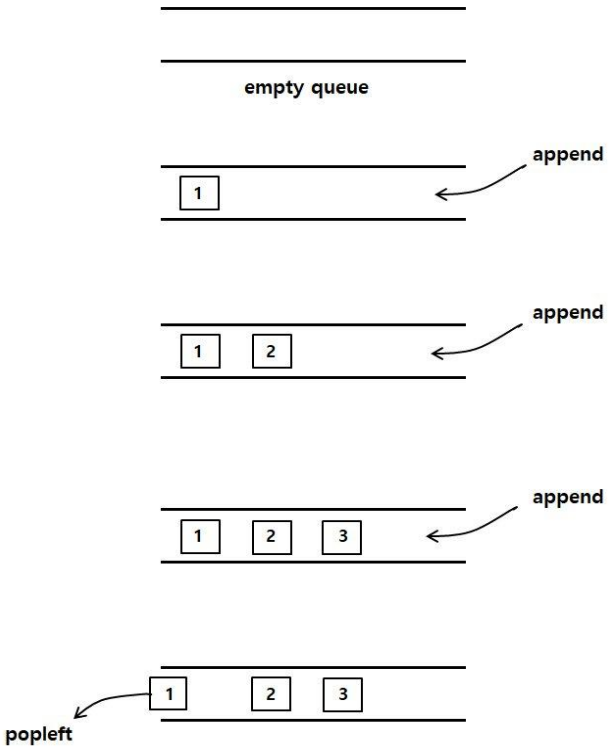
한 쪽 끝에서 자료가 삽입되고, 반대쪽 끝에서 자료가 삭제되는 FIFO(First In First Out) 형식의 자료 구조이다.
즉, 먼저 추가한 항목이 가장 먼저 제거되는 자료구조이다.

2. 큐(Queue)의 연산

1) offer(item): item 하나를 큐의 뒷 부분에 추가한다.
2) poll(): 큐에 가장 앞에 있는 항목을 제거하고 반환한다.

3. 자바에서 큐사용하기

```
public void solution(){
    Queue<Integer> Q = new LinkedList<>();
    Q.offer(1);
    Q.offer(2);
    Q.offer(3);
    Q.poll();
    System.out.println(Q.isEmpty());
    System.out.println(Q.peek());
    Q.poll();
    System.out.println(Q.size());
    Q.poll();
    System.out.println(Q.isEmpty());
}
```



Backspace

현수는 주어진 문자열의 문자 순서대로 키보드 자판의 문자를 쳐 화면에 s문자열을 작성합니다. 문자열에는 '#'문자가 있는데 이 문자는 Backspace키를 의미합니다.
매개변수 s에 현수가 키보드 자판을 쳐야할 순서인 문자열이 주어지면 현수가 s문자열을 작성했을 때 최종적으로 화면에 작성된 문자열을 반환하는 프로그램을 작성하세요.
화면에는 적어도 문자 한 개는 작성되어 있습니다.

입출력 예:

s	answer
"abc##ec#ab"	"aeab"
"kefd#ef##s##"	"ke"
"teac#cher##er"	"teacher"
"englitk##shabcde##ff##ef##ashe####"	"englishabc"
"itistime####gold"	"itisgold"

제한사항:

- 문자열 s의 길이는 1,000을 넘지 않습니다.

연속된 문자 지우기

매개변수 s에 문자열이 주어지면 이웃한 두 개의 문자가 같으면 두 문자를 제거합니다. 이 과정을 반복해서 최종적으로 남는 문자만으로 이루어진 문자열을 반환하는 프로그램을 작성하세요.
만약 "acbbcaa"라는 문자열이 주어진다면 최초 bb가 연속되어 있어 제거하고 나면 "accaa"가 되고, 다시 cc가 연속되어 제거하면 "aaa"가 되고 "aa"연속되어 제거하면 "a"가 최종적으로 남습니다.

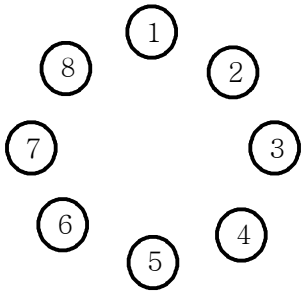
입출력 예:

s	answer
"acbbcaa"	"a"
"bacccaba"	"bacaba"
"aabaababbaa"	"a"
"bcaacccbaabccabbaa"	"ba"
"cacaabbc"	"ca"

제한사항:
- 문자열 s의 길이는 100,000을 넘지 않습니다.
- 문자열 s는 소문자로만 이루어져 있습니다.

공주 구하기

정보 왕국의 이웃 나라 외동딸 공주가 숲속의 괴물에게 잡혀갔습니다.
정보 왕국에는 왕자가 n명이 있는데 서로 공주를 구하러 가겠다고 합니다. 정보왕국의 왕은 다음과 같은 방법으로 공주를 구하러 갈 왕자를 결정하기로 했습니다.
왕은 왕자들을 나이 순으로 1번부터 n번까지 차례로 번호를 매긴다. 그리고 1번 왕자부터 N번 왕자까지 순서대로 시계 방향으로 돌아가며 동그랗게 앉게 한다. 그리고 1번 왕자부터 시계방향으로 돌아가며 1부터 시작하여 번호를 외치게 한다. 한 왕자가 k(특정숫자)를 외치면 그 왕자는 공주를 구하러 가는데서 제외되고 원 밖으로 나오게 된다. 그리고 다음 왕자부터 다시 1부터 시작하여 번호를 외친다.
이렇게 해서 마지막까지 남은 왕자가 공주를 구하러 갈 수 있다.



예를 들어 총 8명의 왕자가 있고, 3을 외친 왕자가 제외된다고 하자. 처음에는 3번 왕자가 3을 외쳐 제외된다. 이어 6, 1, 5, 2, 8, 4번 왕자가 차례대로 제외되고 마지막까지 남게 된 7번 왕자에게 공주를 구하러갑니다.
n과 k가 주어질 때 공주를 구하러 갈 왕자의 번호를 출력하는 프로그램을 작성하시오.

입출력 예:

n	k	result
8	3	7
20	3	20
100	5	47

제한사항:
- N(5<=N<=1,000)
- K(2<=K<=9)

int[] nums = {3, 1, 2, 5, 4};

1. 오름차순 정렬 : Arrays.sort(nums)
   nums 배열을 오름차순 정렬한다.

2. 내림차순 정렬 :
   Integer[] tmp = Arrays.stream(nums).boxed().toArray(Integer[]::new);
   Arrays.sort(tmp, (a, b) -> b - a);
   //음수 끝지점은 잘 정렬이 안됨(-2147483645정도까지만 정렬)

3. 좌표정렬하기

1) [x, y]에서 x값에 의한 오름차순 정렬

```
class Solution {
        public void solution(int[][] nums){
                Arrays.sort(nums, (a, b) -> a[0] - b[0]);
                for(int[] x : nums) System.out.println(x[0] + " " + x[1]);
        }
        public static void main(String[] args){
                Solution T = new Solution();
                T.solution(new int[][]{{1, 2}, {3, 1}, {2, 3}, {1, 5}});
        }
}
```

2) [x, y]에서 y값에 의한 오름차순 정렬

```
class Solution {
        public void solution(int[][] nums){
                Arrays.sort(nums, (a, b) -> a[1] - b[1]);
                for(int[] x : nums) System.out.println(x[0] + " " + x[1]);
        }
        public static void main(String[] args){
                Solution T = new Solution();
                T.solution(new int[][]{{1, 2}, {3, 1}, {2, 3}, {1, 5}});
        }
}
```

3) [x, y]에서 x값에 의한 내림차순 정렬

```
class Solution {
        public void solution(int[][] nums){
                Arrays.sort(nums, (a, b) -> b[0] - a[0]);
                for(int[] x : nums) System.out.println(x[0] + " " + x[1]);
        }
        public static void main(String[] args){
                Solution T = new Solution();
                T.solution(new int[][]{{1, 2}, {3, 1}, {2, 3}, {1, 5}});
        }
}
```

4) [x, y]에서 x값에 의한 오름차순을 하되 x값이 같은 경우는 y값에 따라 내림차순한다.

```
class Solution {
        public void solution(int[][] nums){
                Arrays.sort(nums, (a, b) -> a[0] == b[0] ? b[1] - a[1] : a[0] - b[0]);
                for(int[] x : nums) System.out.println(x[0] + " " + x[1]);
        }
        public static void main(String[] args){
                Solution T = new Solution();
                T.solution(new int[][]{{1, 2}, {3, 1}, {2, 3}, {1, 5}});
        }
}
```

5) ArrayList 정렬 : [x, y]에서 x값에 의한 오름차순 정렬

```
class Solution {
        public int solution(int[][] meetings){
                ArrayList<int[]> list = new ArrayList<>();
                for(int[] x : meetings){
                    list.add(new int[]{x[0], 1});
                    list.add(new int[]{x[1], 2});
                }
                list.sort((a, b) -> a[0] - b[0]);
                //Collections.sort(list, (a, b) -> a[0] - b[0]);
        }
}
```

이진수 정렬

매개변수 nums에 숫자가 주어지면 nums의 원소들을 이진수로 변환했을 때 1의 개수가 적은 것부터 많은 것 순으로 정렬하여 반환하는 프로그램을 작성하세요.
만약 nums = [5, 6, 7, 8, 9]이고 이 원소들을 이진수로 변환하면
5 --> 101 : 1이 2개
6 --> 110 : 1이 2개
7 --> 111 : 1이 3개
8 --> 1000 : 1이 1개
9 --> 1001 : 1이 2개
이고, 이 수들을 이진수에서 1의 개수에 의해 오름차순 정렬하면 [8, 5, 6, 9, 7]이다.
위에 5, 6, 9는 이진수로 변환했을 때 1의 개수가 2개로 동일하면 십진수가 작은순(오름차순)으로 정렬합니다.

입출력 예

nums	answer
[5, 6, 7, 8, 9]	[8, 5, 6, 9, 7]
[5, 4, 3, 2, 1]	[1, 2, 4, 3, 5]
[12, 5, 7, 23, 45, 21, 17]	[5, 12, 17, 7, 21, 23, 45]

제한사항:
- nums의 길이는 1,000을 넘지 않습니다.
- 1 <= nums[i] <= 100,000

그리디 알고리즘(Greedy Algorithm)

탐욕 알고리즘이라고도 하며, 말 그대로 선택의 순간마다 당장 눈앞에 보이는 최고의 상황만을 쫓아 최종적인 해답에 도달하는 방식이다.

문제 :
일렬로 놓여 있는 숫자 카드에서 왼쪽 맨 끝과 오른쪽 맨 끝 카드 중 하나를 가져가는 방식으로 4개의 카드를 가져갔을 때 가져간 카드의 숫자 합의 최댓값은?
[2, 3, 7, 1, 2, 1, 5]

답 : 5 + 2 + 3 + 7 = 17

최대 사과의 개수

여러 종류의 사과박스가 있습니다.
각 박스의 종류에 따라 박스에 담겨있는 사과의 개수가 다릅니다.
트럭에 박스를 실으려고 합니다. 트럭에 박스을 실을 수 있는 최대 개수 제한이 있습니다.
매개변수 box에 각 박스 종류의 정보가 주어지고, limit에 트럭의 실을 수 있는 박스의 최대 개수가 주어지면 트럭에 실을 수 있는 사과의 최대 개수를 반환하는 프로그램을 작성하세요.

입출력 예:

box	limit	answer
[[2, 20], [2, 10], [3, 15], [2, 30]]	5	115
[[1, 50], [2, 20], [3, 30], [2, 31], [5, 25]]	10	302
[[3, 40], [5, 20], [5, 70], [1, 80], [5, 30], [3, 35]]	15	745
[[2, 70], [5, 100], [3, 90], [1, 95]]	8	775
[[80, 20], [50, 10], [70, 15], [70, 30], [80, 70], [90, 88], [70, 75]]	500	23920

제한사항:
• box의 길이는 100,000을 넘지 않습니다.
• box[i][0]은 i 종류 박스의 개수, box[i][1]은 i 종류의 박스 한 개에 들어 있는 사과의 개수입니다. 서로 다른 종류의 박스라도 담아 있는 사과의 개수는 같을 수 있습니다.
• 1 <= box[i][0] <= 100, 1 <= box[i][1] <= 100
• 1 <= limit <= 10,000,000

선긋기

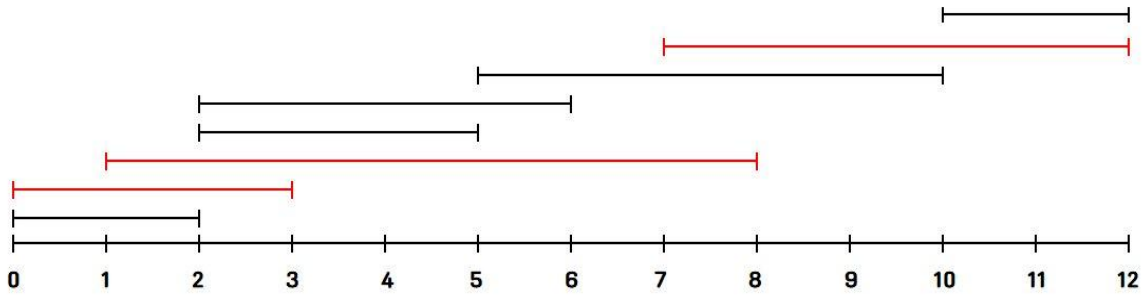
한 번의 선긋기는 수직선상의 한 점에서 다른 한 점까지 선을 긋는 것입니다.
선을 그을 때는 이미 선이 있는 위치에 겹쳐서 그을 수도 있습니다.
여러번 그은 곳과 한 번 그은 곳의 차이는 없습니다.
수직선은 0번 지점부터 m번 지점까지의 길이를 갖고 있습니다.
매개변수 nums에 각각의 선긋기 정보가 주어지면 0번 지점부터 m번 지점까지 연속적인 선이 그어지도록 하기 위한 선긋기 최소횟수를 반환하는 프로그램을 작성하세요.
모든 입력은 0번 지점부터 m번지점까지 연속적인 선이 그어집니다.

입출력 예:

m	nums	answer
12	[[5, 10], [1, 8], [0, 2], [0, 3], [2, 5], [2, 6], [10, 12], [7, 12]]	3
15	[[1, 10], [0, 8], [0, 7], [0, 10], [12, 5], [0, 12], [8, 15], [5, 14]]	2
20	[[3, 7], [5, 8], [15, 20], [0, 5], [7, 14], [3, 10], [0, 8], [13, 18], [5, 9]]	4
30	[[5, 7], [3, 9], [2, 7], [0, 1], [0, 2], [0, 3], [19, 30], [8, 15], [7, 12], [13, 20]]	5
25	[[10, 15], [15, 20], [5, 15], [3, 16], [0, 5], [0, 3], [12, 25]]	3

제한사항:
- 3 <= m <= 10,000
- nums의 길이는 100,000을 넘지 않습니다.
- nums[i][0]은 i번째 선긋기의 시작 점, nums[i][1]은 i번째 선긋기의 끝점입니다.
- 0 <= nums[i][0] < nums[i][1] <= 10,000

예제 1번 설명:



카드 점수(과제)

N개의 카드가 일렬로 놓여져 있습니다. 각 카드에는 숫자가 적혀있습니다.
현수는 카드가 일렬로 놓여진 줄의 양 끝 즉 왼쪽 맨 끝카드와 오른쪽 맨 끝 카드 둘 중 하나를 가져갈 수 있습니다. 현수는 양 끝에서 가져가는 방식으로 k개의 카드를 가져갈 수 있습니다. 그리고 가져간 카드에 적혀진 숫자의 총합이 현수가 얻는 점수입니다.
일려로 놓여진 각 카드의 숫자가 매개변수 nums에 주어지고, 현수가 가져갈 수 있는 카드의 개수 k가 주어지면 현수가 얻을 수 있는 최대점수를 반환하는 프로그램을 작성하세요.

입출력 예:

nums	k	answer
[2, 3, 7, 1, 2, 1, 5]	4	17
[1, 2, 3, 5, 6, 7, 1, 3, 9]	5	26
[1, 30, 3, 5, 6, 7]	3	38
[1, 2, 15, 3, 6, 7, 8, 9]	5	35
[12, 5, 6, 12, 34, 35, 13, 3, 7, 8, 9]	7	117

제한사항:
• nums의 길이는 300,000을 넘지 않습니다.
• nums의 원소는 100을 넘지 않는 자연수입니다..
• 2 <= k < nums의 길이

입력예제 1번 설명 :
왼쪽에서 2, 3, 7, 오른쪽에서 5 이렇게 4개를 가져가면 최대가 됩니다. 2+3+7+5=17입니다.

프로그래머스 Lv.1 - 크레인 인형 뽑기
https://school.programmers.co.kr/learn/courses/30/lessons/64061

정답코드

```
import java.util.*;
class Solution {
    public int solution(int[][] board, int[] moves) {
        int answer = 0;
        Stack<Integer> stack = new Stack<>();
        for(int pos : moves){
            for(int i = 0; i < board.length; i++){
                if(board[i][pos-1] != 0){
                    int tmp = board[i][pos-1];
                    board[i][pos-1] = 0;
                    if(!stack.isEmpty() && stack.peek() == tmp){
                        stack.pop();
                        answer += 2;
                    }
                    else stack.push(tmp);
                    break;
                }
            }
        }
        return answer;
    }
}
```

프로그래머스 Lv.0 - 전국 대회 선발 고사
https://school.programmers.co.kr/learn/courses/30/lessons/181851

정답코드

```
import java.util.*;
class Solution {
    public int solution(int[] rank, boolean[] attendance) {
        int answer = 0;
        ArrayList<int[]> al = new ArrayList<>();
        for(int i = 0; i < rank.length; i++){
            if(attendance[i]){
                al.add(new int[]{rank[i], i});
            }
        }
        al.sort((a, b) -> a[0] - b[0]);
        answer += al.get(0)[1] * 10000;
        answer += al.get(1)[1] * 100;
        answer += al.get(2)[1];
        return answer;
    }
}
```

프로그래머스 Lv.1 - 과일 장수
https://school.programmers.co.kr/learn/courses/30/lessons/135808

정답코드

```
import java.util.*;
class Solution {
    public int solution(int k, int m, int[] score) {
        int answer = 0;
        Integer[] tmp = Arrays.stream(score).boxed().toArray(Integer[]::new);
        Arrays.sort(tmp, (a, b) -> b - a);
        for(int i = m-1; i < tmp.length; i+=m){
            answer += (tmp[i] * m);
        }
        return answer;
    }
}
```

프로그래머스 Lv.1 - 귤 고르기
https://school.programmers.co.kr/learn/courses/30/lessons/138476

정답코드

```
import java.util.*;
class Solution {
    public int solution(int k, int[] tangerine) {
        int answer = 0;
        HashMap<Integer, Integer> nH = new HashMap<>();

        for(int x : tangerine){
            nH.put(x, nH.getOrDefault(x, 0)+1);
        }
        ArrayList<Integer> al = new ArrayList<>(nH.values());
        Collections.sort(al, (a, b) -> b - a);

        for(int x : al){
            if(k > 0){
                k -= x;
                answer++;
            }
            else break;
        }
        return answer;
    }
}
```