코딩역량 강화 과정 Java & SQL
1일차

장은실

강좌 소개

- **강좌명 : 코딩역량 강화 과정 Java & SQL (초급)**
  - Java와 SQL의 알고리즘 구현능력과 자료구조의 이해와 응용 방법에 대한 코딩역량 강화 실습 기반 과정
- **강의 요일 및 시간**
  - 7월 25일 ~ 26일, 오전 9시 ~ 오후 6시 (총 2일, 16시간)
- **강의 방식**
  - 이론 설명 및 실습 병행
- **참고 자료**
  - 교재 – 알고리즘 코딩 테스트, 이지스퍼블리싱 / 쉽게 배우는 자료구조 with 자바, 한빛 아카데미 등
  - 사이트 – 프로그래머스 / 백준 온라인 저지 등

카카오톡
오픈채팅방

강좌 개요 및 내용

- **자바의 자료구조 및 알고리즘 코딩 역량 강화**
  - 1일차 – 알고리즘과 복잡도 / 배열과 리스트 / 문자열 처리 / 자료구조(스택과 큐) / 정렬 알고리즘
  - 2일차 – 정수론, 해시, 탐색, 그리디 알고리즘, 코딩역량 강화 문제 풀기
- **SQL 코딩 역량 강화**
  - 기본 Select 쿼리문, 다양한 Where 조건문, Null 데이터 처리, 함수 활용, SQL 문제 풀기

0

자료구조와 알고리즘 개요

1 자료구조란

2 자료구조 종류

3 자료구조와 알고리즘 관계

1 자료구조란

- **자료를 저장, 조직, 관리하는 방법**

자료구조 적용 전(왼쪽)과 자료구조 적용 후(오른쪽)

자료구조의 일상 예

1 자료구조란

- **문제 해결에 사용할 부품**

- **건축물을 만들려면**
  - 건축 재료와 구조 모듈에 대한 이해가 필요하다
  - 철근, 시멘트, 강화 유리, 벽돌, ...
  - 샷시, 철골, 거푸집, 배수 구조, 전기 및 인터넷 연결 구조, ...

- **프로그래밍과 문제 해결에도**
  - 데이터와 구조 모듈에 대한 이해가 필요하다
  - 프로그래밍 언어, 정수, 문자열, ...
  - 리스트, 스택, 큐, 우선순위 큐, 검색 트리, 해시 테이블, 그래프, ...

다양한 형태의 건축물과 건축 재료

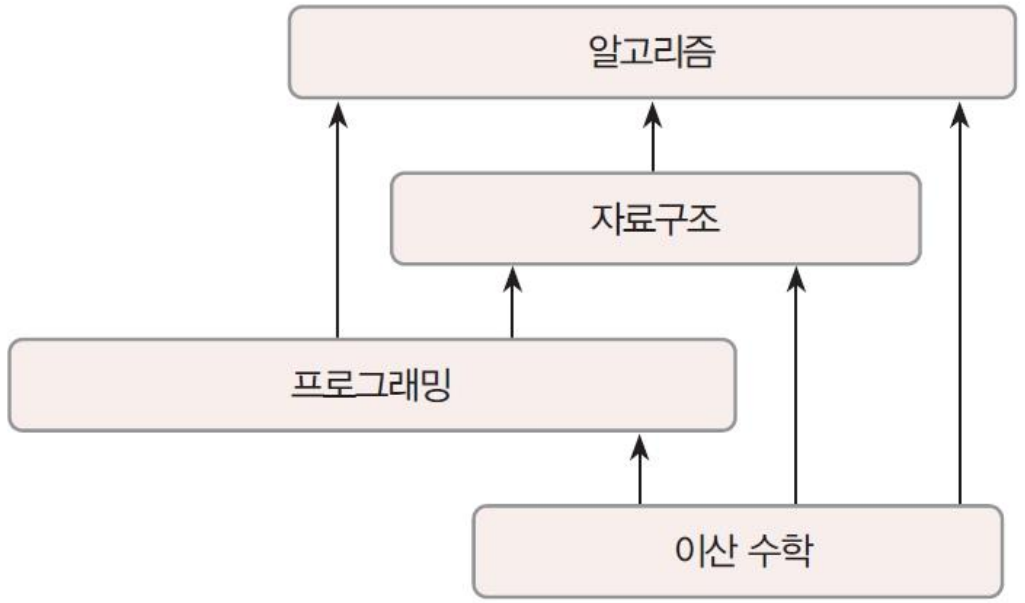
부품(자료구조) 선택의 중요성

1 자료구조란

- **생각하는 방법을 훈련하는 도구**
  - 자료구조는 그 자체로 중요하다
  - 못지 않게, 생각하는 방법 훈련도 중요하다
  - 자료구조를 이용해서 문제를 해결하는 과정
  - 문제 해결 과정에서 논리의 골격이 구성되는 방법 및 스타일
  - 의미의 단위(의미의 매듭)를 설정하는 방법

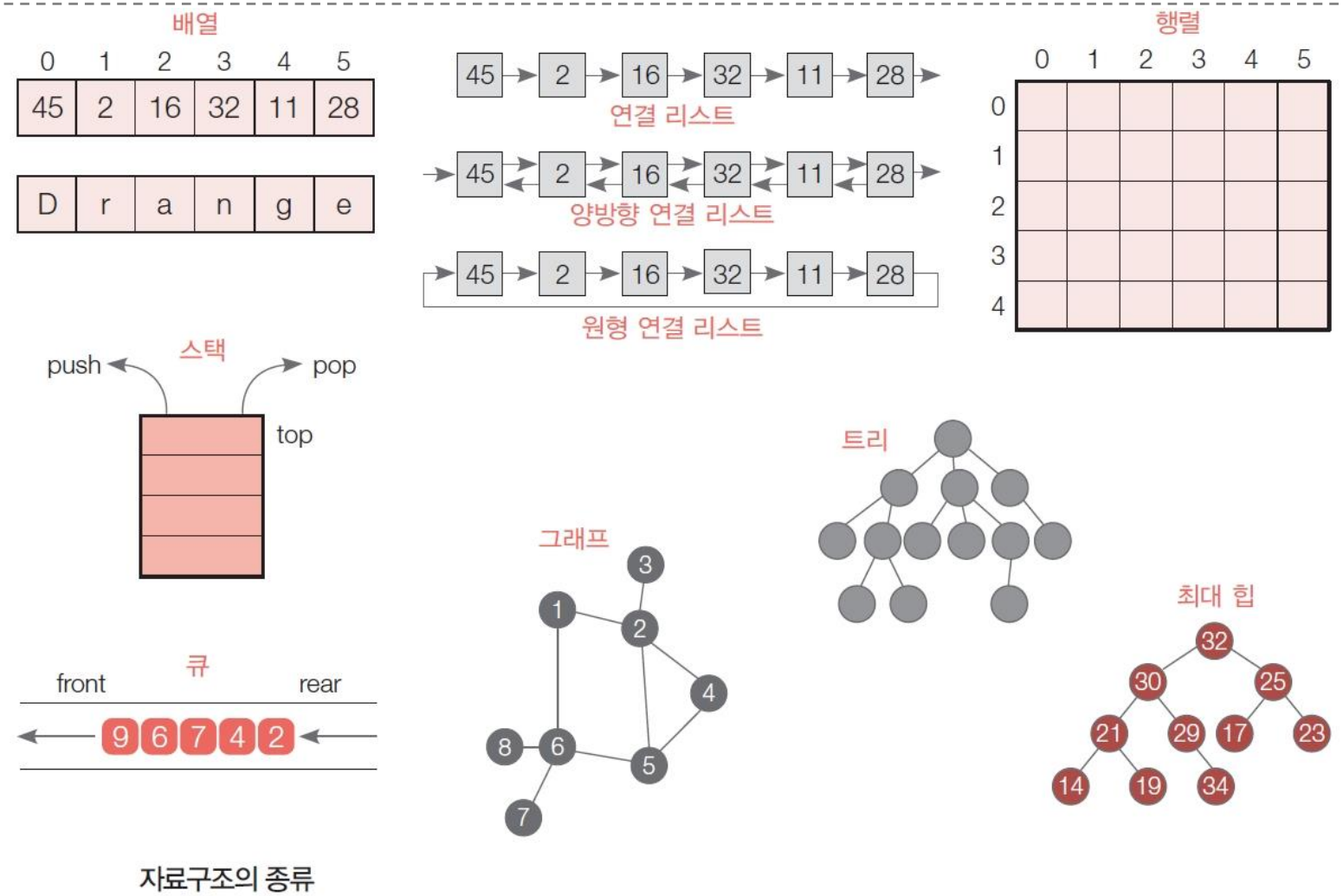
- **본 과정에서 다루는 내용**

자료구조	알고리즘
배열과 리스트	정수론
문자열 처리	정렬
스택과 큐	탐색
해시	그리디



알고리즘, 자료구조, 프로그래밍, 이산 수학의 관계

2 자료구조 종류

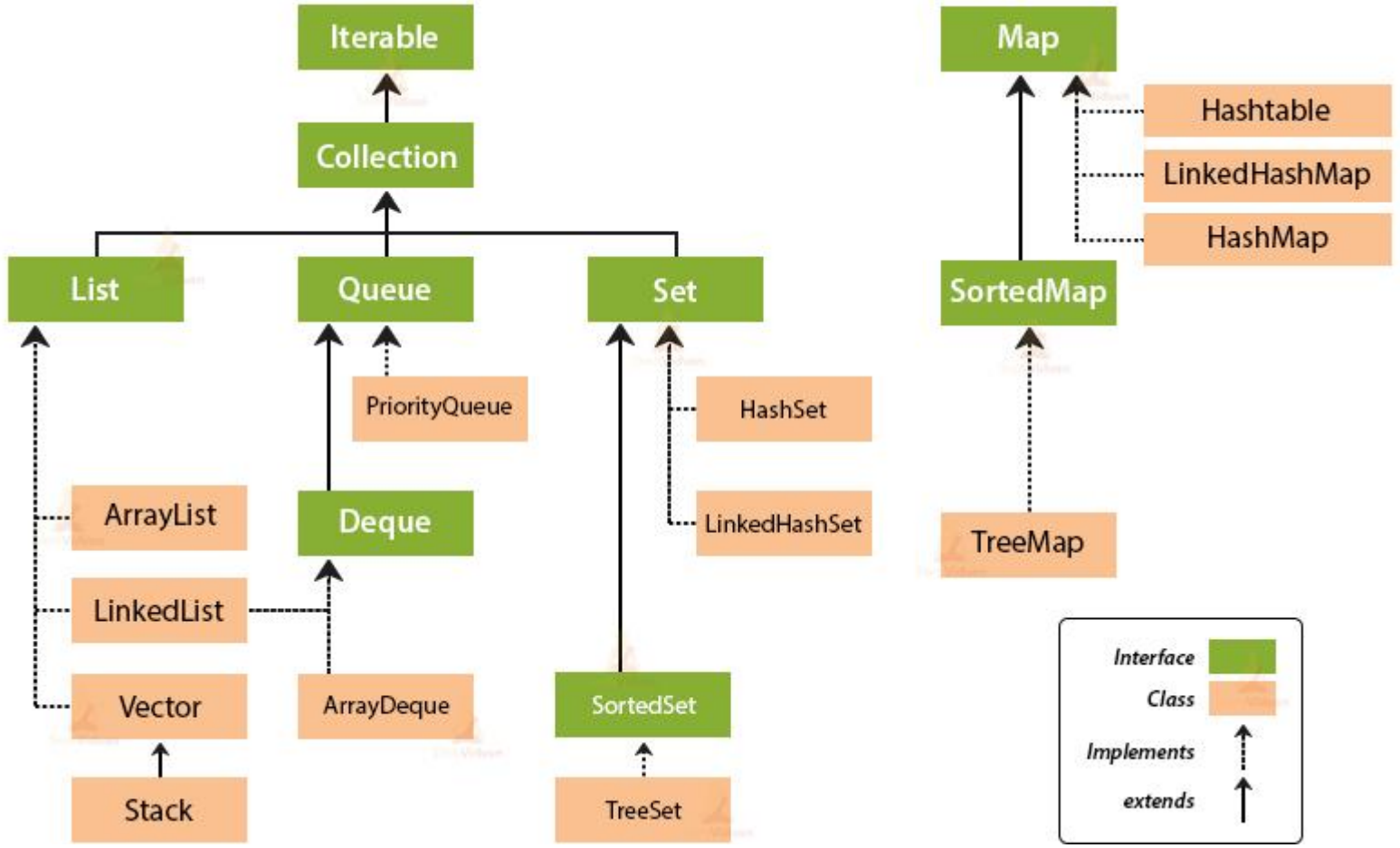


자료구조의 종류

2 자료구조 종류


선형 자료구조
리스트
스택
큐
색인 자료구조
검색 트리
이진 검색 트리
균형 검색 트리
해시 테이블
효율적인 자료구조
우선순위 큐: 힙
관계 처리 자료구조
그래프

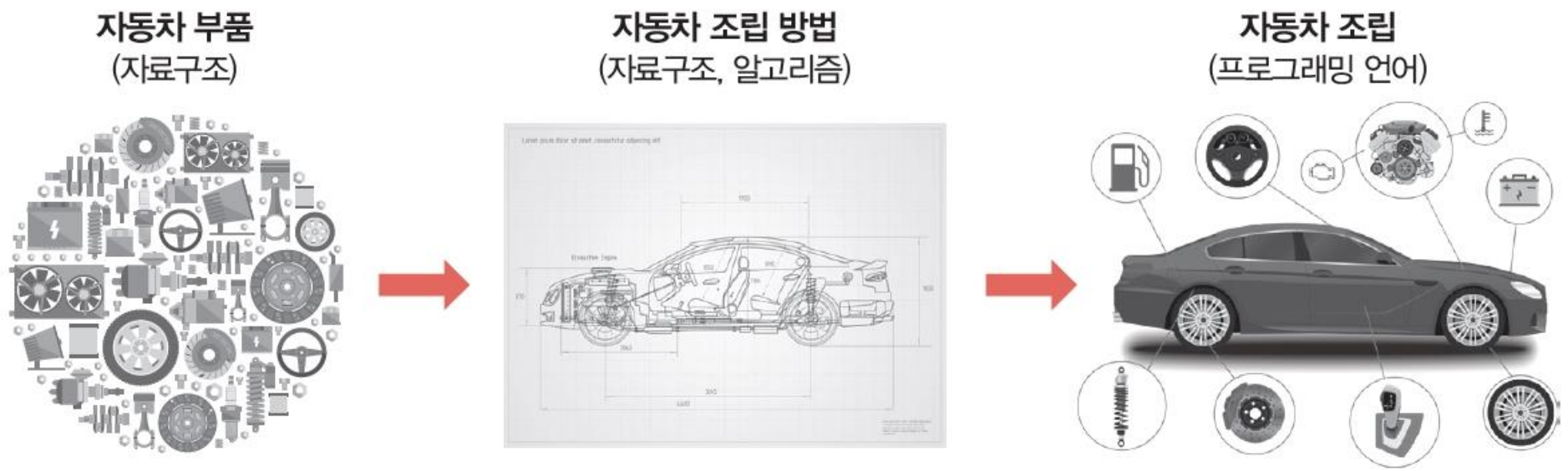
2 자료구조 종류



자바의 컬렉션 프레임워크 계층

3 자료구조와 알고리즘 관계

- **자료구조는 알고리즘 과목의 직전 단계이기도 하고, 그 자체로 여러 알고리즘을 포함**



자료구조와 알고리즘의 관계

3 자료구조와 알고리즘 관계

- **알고리즘 표기법**
  - 자연어를 이용한 서술적 표현
    - 서술적일 뿐만 아니라 쓰는 사람에 따라 일관성이나 명확성을 유지하기 어려움
    - 누구라도 쉽게 이해하고 쓸 수 있어야 하는 알고리즘을 표현하는 데는 한계가 있음
  - 순서도를 이용한 도식화
    - 명령의 흐름을 쉽게 파악할 수 있지만 복잡한 알고리즘을 표현하는 데는 한계가 있음
  - 프로그래밍 언어를 이용한 구체화
    - 추가로 구체화할 필요가 없으나 해당 언어를 모르면 이해하기 어려움
    - 다른 프로그래밍 언어로 프로그램을 개발하는 경우에는 다른 프로그래밍 언어로 변환해야 하므로 범용성이 떨어짐
  - 가상코드를 이용한 추상화
    - 가상코드Pseudo-Code는 직접 실행할 수는 없지만 일반적인 프로그래밍 언어와 형태가 유사해 프로그래밍 언어로
    - 구체화하기가 쉬움

0

Java Programming 되돌아 보기

1 짝수와 홀수

2 정수 제곱근 판별

3 직사각형 별 찍기

⋮

Java Programming 되돌아 보기

1. 짝수와 홀수

<문제 설명>

정수 num이 짝수일 경우 "Even"을 반환하고 홀수인 경우 "Odd"를 반환하는 함수, solution을 완성해 주세요.

<제한 조건>

- num은 int 범위의 정수입니다.
- 0은 짝수입니다.

<입출력 예>

num	return
3	"Odd"
4	"Even"

Java Programming 되돌아 보기

2. 정수 제곱근 판별

<문제 설명>

임의의 양의 정수 n에 대해, n이 어떤 양의 정수 x의 제곱인지 아닌지 판단하려 합니다.

n이 양의 정수 x의 제곱이라면 x+1의 제곱을 리턴하고, n이 양의 정수 x의 제곱이 아니라면 -1을 리턴하는 함수를 완성하세요.

<제한 조건>

- n은 1이상, 50000000000000 이하인 양의 정수입니다.

<입출력 예>

n	return
121	144
3	-1

입출력 예#1

121은 양의 정수 11의 제곱이므로, (11+1)를 제곱한 144를 리턴합니다.

입출력 예#2

3은 양의 정수의 제곱이 아니므로, -1을 리턴합니다.

Java Programming 되돌아 보기

3. 직사각형 별 찍기

<문제 설명>

이 문제에는 표준 입력으로 두 개의 정수 n과 m이 주어집니다.

별(*) 문자를 이용해 가로의 길이가 n, 세로의 길이가 m인 직사각형 형태를 출력해보세요.

<제한 조건>

- n과 m은 각각 1000 이하인 자연수입니다.

<입력 예>

5 3

<출력 예>

*****

*****

*****

Java Programming 되돌아 보기

4. 두 정수 사이의 합

<문제 설명>

두 정수 a, b가 주어졌을 때 a와 b 사이에 속한 모든 정수의 합을 리턴하는 함수, solution을 완성하세요.

예를 들어 a = 3, b = 5인 경우, 3 + 4 + 5 = 12이므로 12를 리턴합니다.

<입출력 예>

a	b	return
3	5	12
3	3	3
5	3	12

<제한 조건>

- a와 b가 같은 경우는 둘 중 아무 수나 리턴하세요.
- a와 b는 -10,000,000 이상 10,000,000 이하인 정수입니다.
- a와 b의 대소관계는 정해져있지 않습니다.

Java Programming 되돌아 보기

5. 나머지가 1이 되는 수 찾기

<문제 설명>

자연수 n이 매개변수로 주어집니다. n을 x로 나눈 나머지가 1이 되도록 하는 가장 작은 자연수 x를 return 하도록 solution 함수를 완성해주세요. 답이 항상 존재함은 증명될 수 있습니다.

<제한 조건>

- $3 \leq n \leq 1{,}000{,}000$

<입출력 예>

n	return
10	3
12	11

입출력 예#1

10을 3으로 나눈 나머지가 1이고, 3보다 작은 자연수 중에서 문제의 조건을 만족하는 수가 없으므로, 3을 return 해야 합니다.

입출력 예#2

12를 11로 나눈 나머지가 1이고, 11보다 작은 자연수 중에서 문제의 조건을 만족하는 수가 없으므로, 11을 return 해야 합니다.

Java Programming 되돌아 보기

6. 부족한 금액 계산하기

<문제 설명>

새로 생긴 놀이기구는 인기가 매우 많아 줄이 끊이질 않습니다. 이 놀이기구의 원래 이용료는 price원 인데, 놀이기구를 N 번째 이용한다면 원래 이용료의 N배를 받기로 하였습니다. 즉, 처음 이용료가 100이었다면 2번째에는 200, 3번째에는 300으로 요금이 인상됩니다.

놀이기구를 count번 타게 되면 현재 자신이 가지고 있는 금액에서 얼마가 모자라는지를 return 하도록 solution 함수를 완성하세요.

단, 금액이 부족하지 않으면 0을 return 하세요.

<제한 조건>

- 놀이기구의 이용료 price : 1 ≤ price ≤ 2,500, price는 자연수
- 처음 가지고 있던 금액 money : 1 ≤ money ≤ 1,000,000,000, money는 자연수
- 놀이기구의 이용 횟수 count : 1 ≤ count ≤ 2,500, count는 자연수

<입출력 예>

price	money	count	result
3	20	4	10

입출력 예#1

이용금액이 3인 놀이기구를 4번 타고 싶은 고객이 현재 가진 금액이 20이라면, 총 필요한 놀이기구의 이용 금액은 30 (= 3+6+9+12) 이 되어 10만큼 부족하므로 10을 return 합니다.

01

어떤 알고리즘으로 풀어야 할까?

01-1 알고리즘 개요

01-2 시간 복잡도 표기법

01-3 시간 복잡도 활용

01-1 알고리즘 개요

알고리즘 개념 및 특징

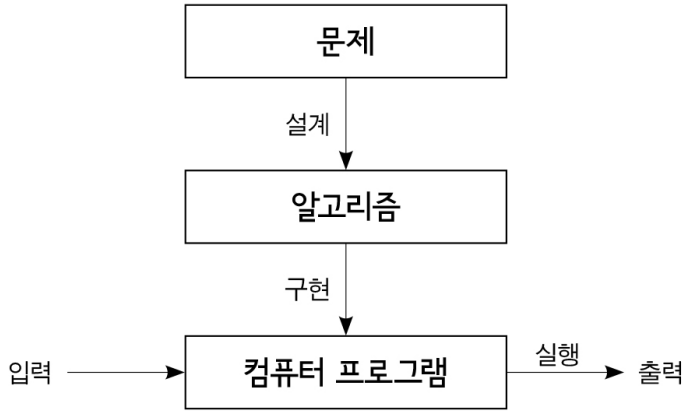
알고리즘이란?

문제를 해결하기 위한 단계적 절차 또는 방법으로 순서대로 구체적이고 명확하게 기술해야 한다.

유효한 입력을 받아 실행한 결과인 해(답)를 출력한다. 예) 요리법, 길 찾기

알고리즘 특징

- 명확성 : 알고리즘의 각 단계는 애매모호하지 않고 명확해야 한다.
- 정확성 : 모든 유효한 입력에 대해 올바른 해를 출력해야 한다.
- 유한성 : 유효한 입력이 주어지면 유한한 시간 내에 종료되어야 한다.



01-2 시간 복잡도 표기법 알아보기

알고리즘에서 시간 복잡도는 주어진 문제를 해결하기 위한 연산 횟수를 말한다.

일반적으로 1억 번의 연산을 1초의 시간으로 간주하여 예측한다.

시간 복잡도 정의하기

시간 복잡도를 정의하는 3가지 유형은 다음과 같다.

시간 복잡도 유형

- 빅-오메가($\Omega(n)$): 최선일 때(best case)의 연산 횟수를 나타낸 표기법
- 빅-세타($\Theta(n)$): 보통일 때(average case)의 연산 횟수를 나타낸 표기법
- 빅-오($O(n)$): 최악일 때(worst case)의 연산 횟수를 나타낸 표기법

01-2 시간 복잡도 표기법 알아보기

코딩 테스트에서는 어떤 시간 복잡 유형을 사용해야 할까?

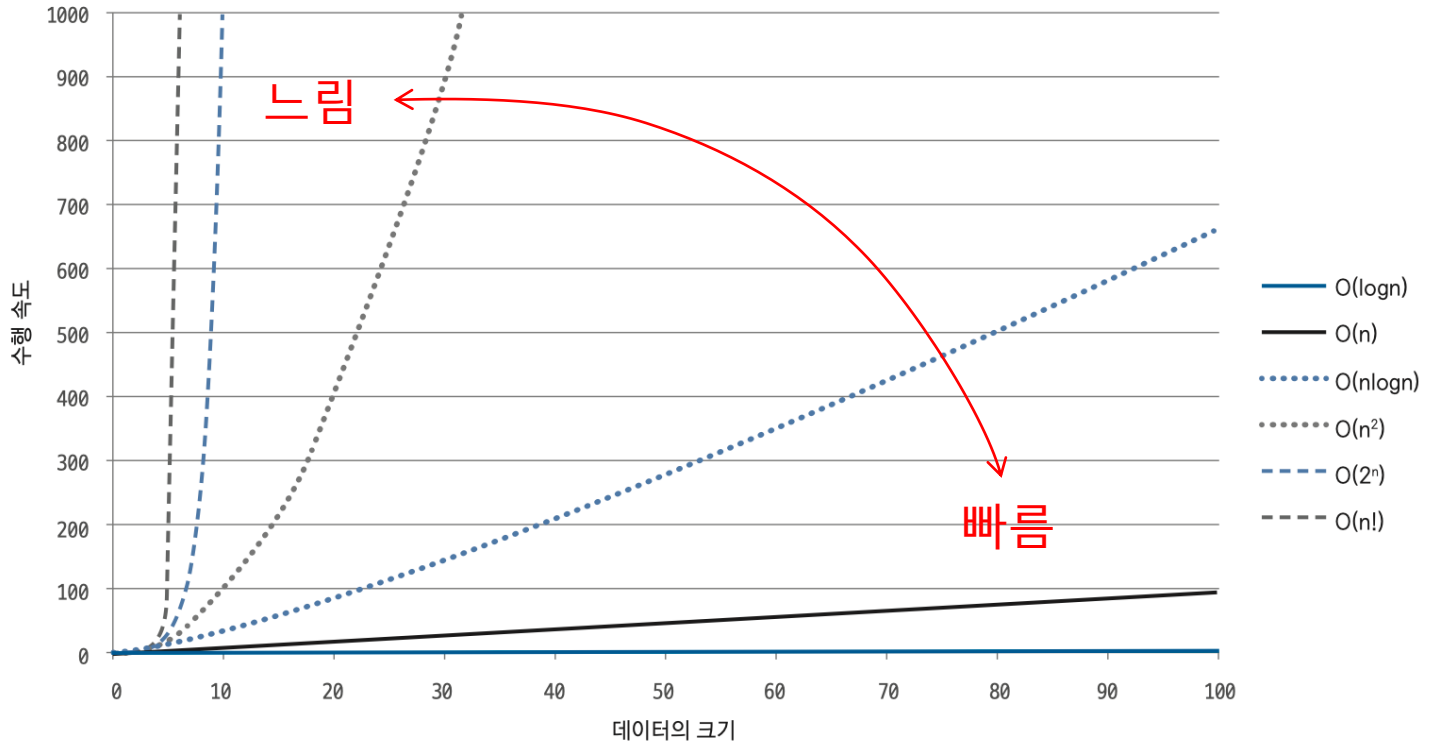
코딩 테스트에서는 **빅-오 표기법(O(n))**을 기준으로 수행시간을 계산하는 것이 좋다.

모든 테스트 케이스를 통과해야만 합격으로 판단하므로
시간 복잡도를 판단할 때는 최악일 때를 염두에 둬야 한다.

01-2 시간 복잡도 표기법 알아보기

다음은 빅-오 표기법(O(n))으로 표현한 시간 복잡도 그래프다.

각각의 시간 복잡도는 데이터 크기(N)의 증가에 따라 성능(수행 시간)이 다르다는 것을 확인할 수 있다.



빅-오 표기법(O(n))의 시간 복잡도

01-3 시간 복잡도 활용하기

시간 제한이 2초이므로 이 조건을 만족하려면 **2억 번 이하의 연산 횟수**로 문제를 해결해야 한다.

시간 제한과 데이터 크기를 바탕으로 어떤 정렬 알고리즘을 사용해야 할 것인지 판단할 수 있다.

연산 횟수 계산 방법

- 연산 횟수 = 알고리즘 시간 복잡도 X 데이터의 크기

01-3 시간 복잡도 활용하기

버블 정렬은 약 10억 번의 연산 횟수가 필요하므로 이 문제를 풀기에 적합한 알고리즘이 아니다.

병합 정렬은 약 2,000만 번의 연산 횟수가 필요하므로 이 문제를 풀기에 적합한 알고리즘이다.

알고리즘 적합성 평가

- 버블 정렬 = $(1{,}000{,}000)^2 = 1{,}000{,}000{,}000{,}000 > 200{,}000{,}000$ → 부적합 알고리즘
- 병합 정렬 = $1{,}000{,}000\log(1{,}000{,}000) = 약\ 20{,}000{,}000 < 200{,}000{,}000$ → 적합 알고리즘

이와 같이 시간 복잡도 분석으로 문제에서 사용할 수 있는 알고리즘을 선택할 수 있다.

즉, 데이터의 크기(N)를 단서로 사용해야 하는 알고리즘을 추측해 볼 수 있다.

01-3 시간 복잡도 활용하기

시간 복잡도를 바탕으로 코드 로직 개선하기

시간 복잡도는 작성한 코드의 비효율적인 로직을 개선하는 바탕으로 사용할 수 있다.

이 부분을 활용하려면 가장 먼저 코드의 시간 복잡도를 도출할 수 있어야 한다.

시간 복잡도를 도출하려면 다음 2가지 기준을 고려한다.

시간 복잡도 도출 기준

① 상수는 시간 복잡도 계산에서 제외한다.

② 가장 많이 중첩된 반복문의 수행 횟수가 시간 복잡도의 기준이 된다.

02

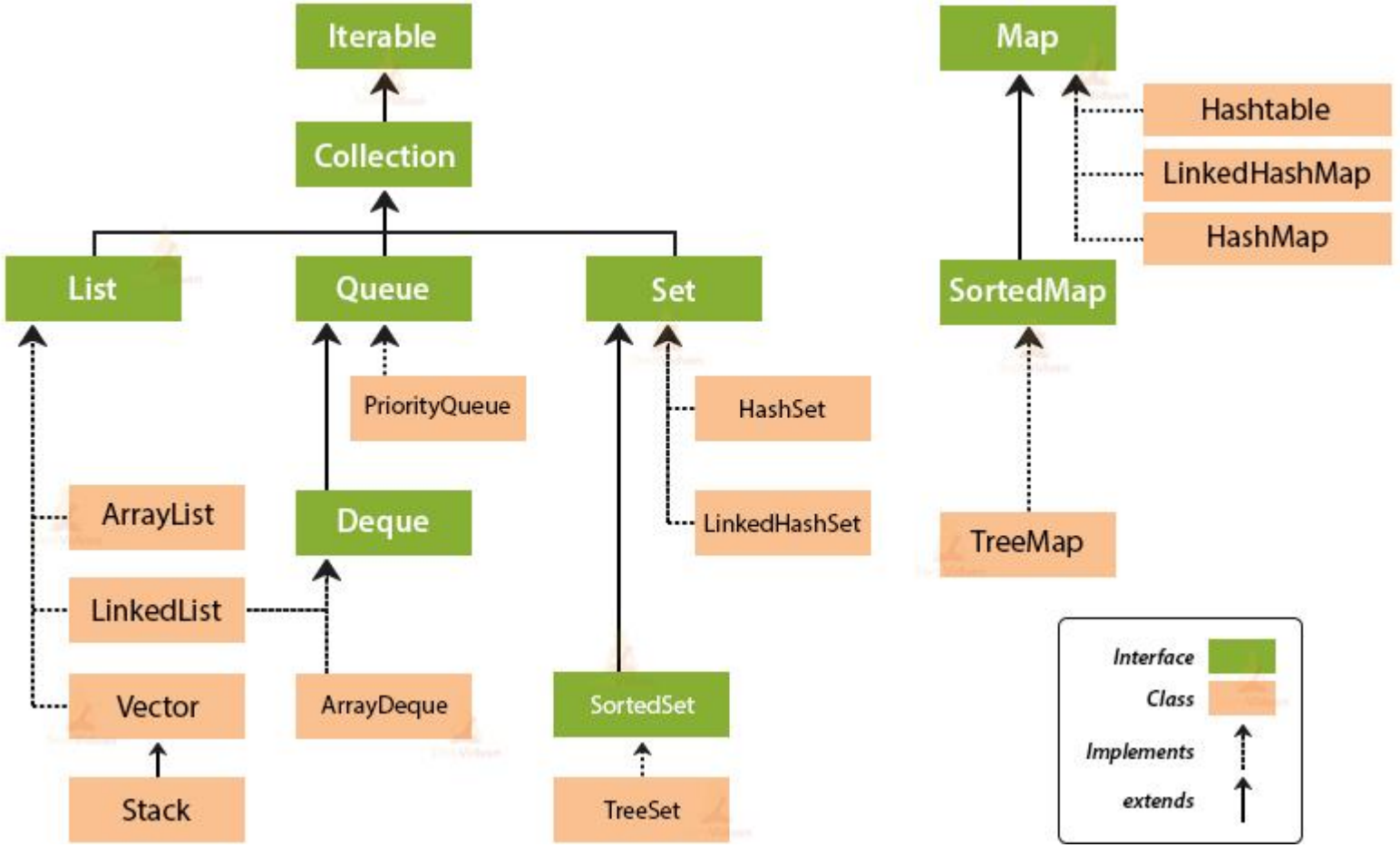
자료구조

02-1 배열과 리스트

02-2 문자열 처리

02-3 스택과 큐

자바의 컬렉션 프레임워크 계층



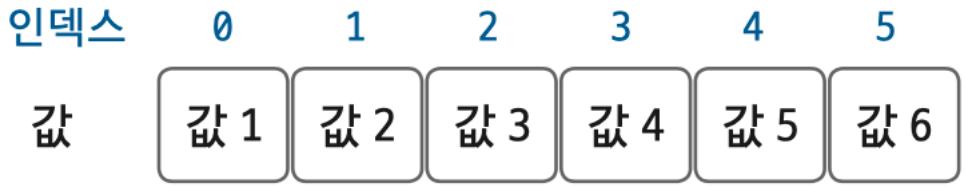
02-1 배열과 리스트

배열과 리스트의 핵심 이론

배열

배열은 메모리의 연속 공간에 값이 채워져 있는 형태의 자료구조다.

배열의 값은 인덱스를 통해 참조할 수 있으며, 선언한 자료형의 값만 저장할 수 있다.



배열의 구조

02-1 배열과 리스트

배열의 특징

① 인덱스를 사용하여 값에 바로 접근할 수 있다.

② 새로운 값을 삽입하거나 특정 인덱스에 있는 값을 삭제하기 어렵다. 값을 삽입하거나 삭제하려면 해당 인덱스 주변에 있는 값을 이동시키는 과정이 필요하다.

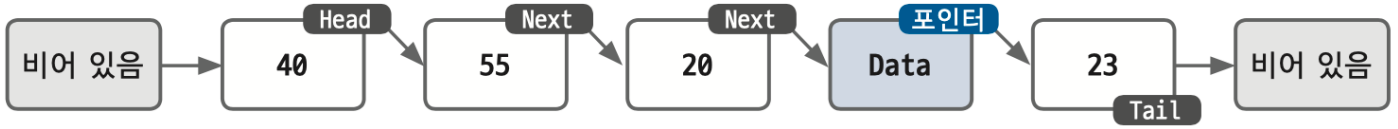
③ 배열의 크기는 선언할 때 지정할 수 있으며, 한 번 선언하면 크기를 늘리거나 줄일 수 없다.

④ 구조가 간단하므로 코딩 테스트에서 많이 사용한다.

02-1 배열과 리스트

리스트

리스트는 값과 포인터를 묶는 노드라는 것을 포인터로 연결한 자료구조다.



리스트의 구조

리스트의 특징

① 인덱스가 없으므로 값에 접근하려면 Head 포인터부터 순서대로 접근해야 한다. 다시 말해 값에 접근하는 속도가 느리다.

② 포인터로 연결되어 있으므로 데이터를 삽입하거나 삭제하는 연산 속도가 빠르다.

③ 선언할 때 크기를 별도로 지정하지 않아도 된다. 다시 말해 리스트의 크기는 정해져 있지 않으며, 크기가 변하기 쉬운 데이터를 다룰 때 적절하다.

④ 포인터를 저장할 공간이 필요하므로 배열보다 구조가 복잡하다.

02-1 Array(배열)과 ArrayList(리스트)

	Array	ArrayList
사이즈	초기화시 고정 int[] myArray = new int[6];	초기화시 사이즈를 표시하지 않음. 유동적 ArrayList<Integer> myArrayList = new ArrayList<>();
속도	초기화시 메모리에 할당되어 속도가 빠르다.	추가시 메모리를 재할당하여 속도가 느리다.
변경	사이즈 변경 불가	추가 삭제 가능 add(), remove() 로 가능
다차원	가능 int[][][] muttiArray = new int [3][3][3];	불가능

02-1 배열과 리스트

1. 평균 구하기

<문제 설명>

정수를 담고 있는 배열 arr의 평균값을 return하는 함수, solution을 완성해보세요.

<제한 조건>

- arr은 길이 1 이상, 100 이하인 배열입니다.
- arr의 원소는 -10,000 이상 10,000 이하인 정수입니다.

<입출력 예>

arr	return
[1,2,3,4]	2.5
[5,5]	5

02-1 배열과 리스트

2. 내적

<문제 설명>

길이가 같은 두 1차원 정수 배열 a, b가 매개변수로 주어집니다. a와 b의 내적을 return 하도록 solution 함수를 완성해주세요.

이때, a와 b의 내적은 a[0]*b[0] + a[1]*b[1] + ... + a[n-1]*b[n-1] 입니다. (n은 a, b의 길이)

<제한 조건>

- a, b의 길이는 1 이상 1,000 이하입니다.
- a, b의 모든 수는 -1,000 이상 1,000 이하입니다.

<입출력 예>

a	b	result
[1,2,3,4]	[-3,-1,0,2]	3
[-1,0,1]	[1,0,-1]	-2

입출력 예#1

a와 b의 내적은 1*(-3) + 2*(-1) + 3*0 + 4*2 = 3 입니다.

입출력 예#2

a와 b의 내적은 (-1)*1 + 0*0 + 1*(-1) = -2 입니다

02-1 배열과 리스트

3. 음양 더하기

<문제 설명>

어떤 정수들이 있습니다. 이 정수들의 절댓값을 차례대로 담은 정수 배열 absolutes와 이 정수들의 부호를 차례대로 담은 불리언 배열 signs가 매개변수로 주어집니다. 실제 정수들의 합을 구하여 return 하도록 solution 함수를 완성해주세요.

<제한 조건>

- absolutes의 길이는 1 이상 1,000 이하입니다.
  - absolutes의 모든 수는 각각 1 이상 1,000 이하입니다.
- signs의 길이는 absolutes의 길이와 같습니다.
  - signs[i] 가 참이면 absolutes[i] 의 실제 정수가 양수임을, 그렇지 않으면 음수임을 의미합니다.

<입출력 예>

absolutes	signs	result
[4,7,12]	[true,false,true]	9
[1,2,3]	[false,false,true]	0

입출력 예#1

signs가 [true,false,true] 이므로, 실제 수들의 값은 각각 4, -7, 12입니다. 따라서 세 수의 합인 9를 return 해야 합니다.

입출력 예#2

signs가 [false,false,true] 이므로, 실제 수들의 값은 각각 -1, -2, 3입니다. 따라서 세 수의 합인 0을 return 해야 합니다.

02-1 배열과 리스트

4. 없는 숫자 더하기

<문제 설명>

0부터 9까지의 숫자 중 일부가 들어있는 정수 배열 numbers가 매개변수로 주어집니다. numbers에서 찾을 수 없는 0부터 9까지의 숫자를 모두 찾아 더한 수를 return 하도록 solution 함수를 완성해주세요.

<제한 조건>

- 1 ≤ numbers의 길이 ≤ 9
  - 0 ≤ numbers의 모든 원소 ≤ 9
  - numbers의 모든 원소는 서로 다릅니다.

<입출력 예>

numbers	result
[1,2,3,4,6,7,8,0]	14
[5,8,4,0,6,7,9]	6

입출력 예#1

5, 9가 numbers에 없으므로, 5 + 9 = 14를 return 해야 합니다.

입출력 예#2

1, 2, 3이 numbers에 없으므로, 1 + 2 + 3 = 6을 return 해야 합니다.

02-1 배열과 리스트

5. x만큼 간격이 있는 n개의 숫자

<문제 설명>

함수 solution은 정수 x와 자연수 n을 입력 받아, x부터 시작해 x씩 증가하는 숫자를 n개 지니는 리스트를 리턴해야 합니다. 다음 제한 조건을 보고, 조건을 만족하는 함수, solution을 완성해주세요.

<입출력 예>

x	n	Answer
2	5	[2,4,6,8,10]
4	3	[4,8,12]
-4	2	[-4, -8]

<제한 조건>

- x는 -10000000 이상, 10000000 이하인 정수입니다.
- n은 1000 이하인 자연수입니다.

02-1 배열과 리스트

6. 행렬의 덧셈

<문제 설명>

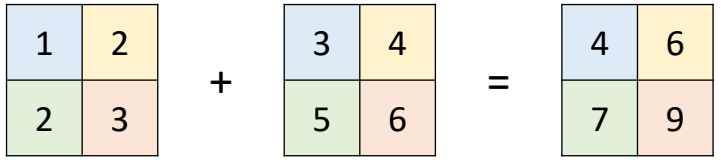
행렬의 덧셈은 행과 열의 크기가 같은 두 행렬의 같은 행, 같은 열의 값을 서로 더한 결과가 됩니다. 2개의 행렬 arr1과 arr2를 입력받아, 행렬 덧셈의 결과를 반환하는 함수, solution을 완성해주세요.

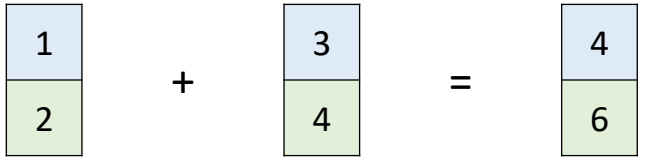
<제한 조건>

- 행렬 arr1, arr2의 행과 열의 길이는 500을 넘지 않습니다.

<입출력 예>

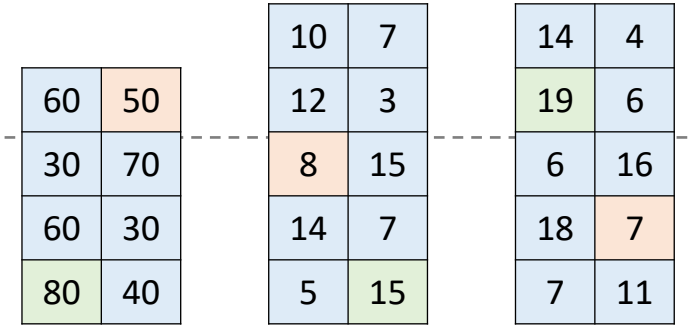
arr1	arr2	return
[[1,2],[2,3]]	[[3,4],[5,6]]	[[4,6],[7,9]]
[[1],[2]]	[[3],[4]]	[[4],[6]]





02-1 배열과 리스트

7. 최소직사각형



<문제 설명>

명함 지갑을 만드는 회사에서 지갑의 크기를 정하려고 합니다. 다양한 모양과 크기의 명함들을 모두 수납할 수 있으면서, 작아서 들고 다니기 편한 지갑을 만들어야 합니다. 이러한 요건을 만족하는 지갑을 만들기 위해 디자인팀은 모든 명함의 가로 길이와 세로 길이를 조사했습니다. 아래 표는 4가지 명함의 가로 길이와 세로 길이를 나타냅니다.

명함 번호	가로 길이	세로 길이
1	60	50
2	30	70
3	60	30
4	80	40

<제한 조건>

- sizes의 길이는 1 이상 10,000 이하.
  - sizes의 원소는 [w, h] 형식입니다.
  - w는 명함의 가로 길이를 나타냅니다.
  - h는 명함의 세로 길이를 나타냅니다.
  - w와 h는 1 이상 1,000 이하인 자연수입니다.

<입출력 예>

sizes	result
[[60, 50], [30, 70], [60, 30], [80, 40]]	4000
[[10, 7], [12, 3], [8, 15], [14, 7], [5, 15]]	122
[[14, 4], [19, 6], [6, 16], [18, 7], [7, 11]]	133

입출력 예#1
명함들을 적절히 회전시켜 겹쳤을 때, 모든 명함을 포함하는 가장 작은 지갑의 크기는 4000(=80 x 50)입니다.

입출력 예#2
명함들을 적절히 회전시켜 겹쳤을 때, 3번째 명함(가로: 8, 세로: 15)이 다른 모든 명함보다 크기가 큽니다. 따라서 지갑의 크기는 3번째 명함의 크기와 같으며, 120(=8 x 15)을 return.

입출력 예#3
명함들을 적절히 회전시켜 겹쳤을 때, 모든 명함을 포함하는 가장 작은 지갑의 크기는 133(=19 x 7)입니다.

02-1 배열과 리스트

8. 제일 작은 수 제거하기

<문제 설명>

정수를 저장한 배열, arr 에서 가장 작은 수를 제거한 배열을 리턴하는 함수, solution을 완성해주세요. 단, 리턴하려는 배열이 빈 배열인 경우엔 배열에 -1을 채워 리턴하세요. 예를들어 arr이 [4,3,2,1]인 경우는 [4,3,2]를 리턴 하고, [10]면 [-1]을 리턴 합니다.

<입출력 예>

arr	return
[4,3,2,1]	[4,3,2]
[10]	[-1]

<제한 조건>

- arr은 길이 1 이상인 배열입니다.
- 인덱스 i, j에 대해 i ≠ j이면 arr[i] ≠ arr[j] 입니다.

02-1 배열과 리스트

9. 같은 숫자는 싫어

<문제 설명>

배열 arr의 각 원소는 숫자 0부터 9까지로 이루어져 있습니다. 이때, 배열 arr에서 연속적으로 나타나는 숫자는 하나만 남기고 전부 제거하려고 합니다. 단, 제거된 후 남은 수들을 반환할 때는 배열 arr의 원소들의 순서를 유지해야 합니다. 예를 들면,

- arr = [1, 1, 3, 3, 0, 1, 1] 이면 [1, 3, 0, 1] 을 return 합니다.
- arr = [4, 4, 4, 3, 3] 이면 [4, 3] 을 return 합니다.

배열 arr에서 연속적으로 나타나는 숫자는 제거하고 남은 수들을 return 하는 solution 함수를 완성해 주세요.

<제한 조건>

- 배열 arr의 크기 : 1,000,000 이하의 자연수
- 배열 arr의 원소의 크기 : 0보다 크거나 같고 9보다 작거나 같은 정수

<입출력 예>

arr	return
[1,1,3,3,0,1,1]	[1,3,0,1]
[4,4,4,3,3]	[4,3]

02-2 문자열 처리

문자열의 가변(mutable)과 불변(immutable)

문자열의 불변이란?

말그대로 "변경할 수 없다"입니다. 불변이기에 문자 변경이 생긴다면 새로운 문자열 객체를 생성하게 됩니다.

※문자열 관련 클래스: java.lang.String

- 장점: 간단한 조작, 동기화 이슈 없음(Tread-safe), 내부 데이터 공유 가능
- 단점: 문자열 연산이 많은 경우, 시간과 자원을 많이 사용함에 오버헤드가 발생하여 성능이 다소 좋지 않습니다.

문자열의 가변이란?

말그대로 "변경할 수 있다"입니다. 가변이기에 문자 변경이 생긴다면 기존의 문자에서 새로운 문자열이 추가됩니다.

※문자열 관련 클래스: java.lang.StringBuffer or java.lang.StringBuilder

- 장점: 문자열 연산(추가, 수정, 삭제)이 많은 경우, 성능이 우수합니다.
- 단점: 수없이 읽는 경우 새로운 String 객체를 생성하게 됨으로 성능 저하가 발생한다. StringBuilder 클래스는 동기화를 지원하지 않기 때문에 멀티 스레드 환경에는 적합하지 않다. 멀티 스레드 환경에서는 StringBuffer 클래스를 사용한다.

02-2 문자열 처리

StringBuffer와 StringBuilder의 차이점

동기화의 유무

StringBuffer : 동기화를 지원하여 멀티쓰레드 환경에서 안전합니다.(Thread-safe)

StringBuilder : 동기화를 지원하지 않습니다.

※참고: String은 불변 매개체이기 때문에 Tread-safe 합니다.

언제 사용하는가?

StringBuffer : 멀티쓰레드 환경

StringBuilder : 싱글쓰레드 환경 또는 멀티쓰레드 환경이어도 동기화가 필요치 않은 경우

※참고: StringBuilder는 동기화를 고려하지 않기 때문에 싱글쓰레드 환경에서 연산처리가 빠른 장점이 있습니다.

02-2 String 클래스 주요 메서드

length() – 문자열 데이터의 길이 반환

concat() – 문자열과 문자열 결합

substring() – 문자열을 지정한 시작 인덱스에서 종료 인덱스까지 자름

indexOf() – 문자가 위치한 인덱스 반환

charAt() – indexOf()와 반대로 인덱스에 위치한 문자 반환

isEmpty() – 해당 문자열이 빈 값인지 확인

lastIndexOf() – indexOf()와 달리 뒤에서부터 조회해서 인덱스 반환

compareTo() – 해당 문자열을 인수로 전달된 문자열과 사전 편찬 순으로 비교

equals() – 두 문자열 객체가 같은지 비교

replace() – 문자열의 일부를 다른 문자열로 바꿔줌

replaceAll() – 특정 패턴의 문자열을 다른 문자열로 바꿔줌

toUpperCase() – 문자열을 대문자로 바꿔줌

toLowerCase() – 문자열을 소문자로 바꿔줌

trim() – 문자열 앞뒤의 공백 제거

split() – 문자열을 잘라서 배열로 반환

toCharArray() – split()과 같지만, 반환하는 배열의 타입이 문자형(char[]) 배열임

getBytes() – byte 배열(아스키 코드)로 반환

02-2 StringBuilder 클래스 주요 메서드

length() – 문자열 길이 반환

append() – 문자열을 맨 뒤에 추가

insert() – 지정 인덱스 위치에 삽입

delete() – 지정 시작 인덱스에서 종료 인덱스의 문자열 일부분 삭제

deleteCharAt() – 지정 인덱스의 문자 하나 삭제

reverse() – 문자열을 거꾸로 뒤집음

replace() – 지정 시작 인덱스에서 종료 인덱스의 문자열을 다른 문자열로 대체

substring() – 지정 시작 인덱스에서 종료 인덱스의 문자열 반환

toString() – 문자열로 변환

02-2 문자열 처리

1. 수박수박수박수박수?

<문제 설명>

길이가 n이고, "수박수박수박수...."와 같은 패턴을 유지하는 문자열을 리턴하는 함수, solution을 완성하세요. 예를들어 n이 4이면 "수박수박"을 리턴하고 3이라면 "수박수"를 리턴하면 됩니다.

<입출력 예>

n	return
3	"수박수"
4	"수박수박"

<제한 조건>

- n은 길이 10,000이하인 자연수입니다.

02-2 문자열 처리

2. 문자열을 정수로 바꾸기

<문제 설명>

문자열 s를 숫자로 변환한 결과를 반환하는 함수, solution을 완성하세요.

<제한 조건>

- s의 길이는 1 이상 5이하입니다.
- s의 맨앞에는 부호(+, -)가 올 수 있습니다.
- s는 부호와 숫자로만 이루어져있습니다.
- s는 "0"으로 시작하지 않습니다.

<입출력 예>

예를들어 str이 "1234"이면 1234를 반환하고, "-1234"이면 -1234를 반환하면 됩니다.

str은 부호(+,-)와 숫자로만 구성되어 있고, 잘못된 값이 입력되는 경우는 없습니다.

02-2 문자열 처리

3. 자릿수 더하기

<문제 설명>

자연수 N이 주어지면, N의 각 자릿수의 합을 구해서 return 하는 solution 함수를 만들어 주세요.

예를들어 N = 123이면 1 + 2 + 3 = 6을 return 하면 됩니다.

<제한 조건>

- N의 범위 : 100,000,000 이하의 자연수

<입출력 예>

n	return
123	6
987	24

입출력 예#1

문제의 예시와 같습니다.

입출력 예#2

9 + 8 + 7 = 24이므로 24를 return 하면 됩니다.

02-2 문자열 처리

4. 자연수 뒤집어 배열로 만들기

<문제 설명>

자연수 n을 뒤집어 각 자리 숫자를 원소로 가지는 배열 형태로 리턴해주세요. 예를들어 n이 12345이면 [5,4,3,2,1]을 리턴합니다.

<입출력 예>

n	return
123456	[5,4,3,2,1]

<제한 조건>

- n은 10,000,000,000이하인 자연수입니다.

02-2 문자열 처리

5. 정수 내림차순으로 배치하기

<문제 설명>

함수 solution은 정수 n을 매개변수로 입력받습니다. n의 각 자릿수를 큰것부터 작은 순으로 정렬한 새로운 정수를 리턴해주세요. 예를들어 n이 118372면 873211을 리턴하면 됩니다.

<입출력 예>

n	return
118372	873211

<제한 조건>

- n은 1이상 8000000000 이하인 자연수입니다.

02-2 문자열 처리

6. 핸드폰 번호 가리기

<문제 설명>

프로그래머스 모바일은 개인정보 보호를 위해 고지서를 보낼 때 고객들의 전화번호의 일부를 가립니다.

전화번호가 문자열 phone_number로 주어졌을 때, 전화번호의 뒷 4자리를 제외한 나머지 숫자를 전부 *으로 가린 문자열을 리턴하는 함수, solution을 완성해주세요.

<입출력 예>

phone_number	return
"01033334444"	"*******4444"
"027778888"	"*****8888"

<제한 조건>

- phone_number는 길이 4 이상, 20이하인 문자열입니다.

02-2 문자열 처리

7. 서울에서 김서방 찾기

<문제 설명>

String형 배열 seoul의 element중 "Kim"의 위치 x를 찾아, "김서방은 x에 있다"는 String을 반환하는 함수, solution을 완성하세요. seoul에 "Kim"은 오직 한 번만 나타나며 잘못된 값이 입력되는 경우는 없습니다.

<입출력 예>

seoul	return
["Jane", "Kim"]	"김서방은 1에 있다"

<제한 조건>

- seoul은 길이 1 이상, 1000 이하인 배열입니다.
- seoul의 원소는 길이 1 이상, 20 이하인 문자열입니다.
- "Kim"은 반드시 seoul 안에 포함되어 있습니다.

02-2 문자열 처리

8. 3진법 뒤집기

<문제 설명>

자연수 n이 매개변수로 주어집니다. n을 3진법 상에서 앞뒤로 뒤집은 후, 이를 다시 10진법으로 표현한 수를 return 하도록 solution 함수를 완성해주세요.

<제한 조건>

- n은 1 이상 100,000,000 이하인 자연수입니다.

<입출력 예>

n	result
45	7
125	229

입출력 예#1

n(10진법)	n(3진법)	반전(3진법)	10진법
45	1200	0021	7

입출력 예#2

n(10진법)	n(3진법)	반전(3진법)	10진법
125	11122	22111	229

02-2 문자열 처리

9. 문자열 내 p와 y의 개수

<문제 설명>

대문자와 소문자가 섞여있는 문자열 s가 주어집니다. s에 'p'의 개수와 'y'의 개수를 비교해 같으면 True, 다르면 False를 return 하는 solution를 완성하세요. 'p', 'y' 모두 하나도 없는 경우는 항상 True를 리턴합니다. 단, 개수를 비교할 때 대문자와 소문자는 구별하지 않습니다.

예를 들어 s가 "pPoooyY"면 true를 return하고 "Pyy"라면 false를 return합니다.

<제한 조건>

- 문자열 s의 길이 : 50 이하의 자연수
- 문자열 s는 알파벳으로만 이루어져 있습니다.

<입출력 예>

s	answer
"pPoooyY"	true
"Pyy"	false

입출력 예#1

'p'의 개수 2개, 'y'의 개수 2개로 같으므로 true를 return 합니다.

입출력 예#2

'p'의 개수 1개, 'y'의 개수 2개로 다르므로 false를 return 합니다.

02-2 문자열 처리

10. 가운데 글자 가져오기

<문제 설명>

단어 s의 가운데 글자를 반환하는 함수, solution을 만들어 보세요. 단어의 길이가 짝수라면 가운데 두글자를 반환하면 됩니다.

<입출력 예>

s	answer
"abcde"	"c"
"qwer"	"we"

<제한 조건>

- s는 길이가 1 이상, 100이하인 스트링입니다.

02-2 문자열 처리

11. 문자열 내 숫자 개수

<문제 설명>

문자열 s의 길이가 4 혹은 6이고, 숫자로만 구성돼있는지 확인해주는 함수, solution을 완성하세요. 예를 들어 s가 "a234"이면 False를 리턴하고 "1234"라면 True를 리턴하면 됩니다.

<입출력 예>

s	return
"a234"	false
"1234"	true

<제한 조건>

- s는 길이 1 이상, 길이 8 이하인 문자열입니다.

02-2 문자열 처리 응용

1. 숫자 문자열과 영단어

<문제 설명>

네오와 프로도가 숫자놀이를 하고 있습니다. 네오가 프로도에게 숫자를 건넬 때 일부 자릿수를 영단어로 바꾼 카드를 건네주면 프로도는 원래 숫자를 찾는 게임입니다.

- 1478 → "one4seveneight"
- 234567 → "23four5six7"
- 10203 → "1zerotwozero3"

숫자	영단어
0	zero
1	one
2	two
3	three
4	four
5	five
6	six
7	seven
8	eight
9	nine

<제한 조건>

- 1 ≤ s의 길이 ≤ 50
- s가 "zero" 또는 "0"으로 시작하는 경우는 주어지지 않습니다.
- return 값이 1 이상 2,000,000,000 이하의 정수가 되는 올바른 입력만 s로 주어집니다.

<입출력 예>

s	return
"one4seveneight"	1478
"23four5six7"	234567
"2three45sixseven"	234567
"123"	123

입출력 예#3

"three"는 3, "six"는 6, "seven"은 7에 대응되기 때문에 정답은 입출력 예 #2와 같은 234567이 됩니다.
입출력 예 #2와 #3과 같이 같은 정답을 가리키는 문자열이 여러 가지가 나올 수 있습니다.

입출력 예#4

s에는 영단어로 바뀐 부분이 없습니다.

02-2 문자열 처리 응용

2. 문자열 내림차순으로 배치하기

<문제 설명>

문자열 s에 나타나는 문자를 큰것부터 작은 순으로 정렬해 새로운 문자열을 리턴하는 함수, solution을 완성해주세요.

s는 영문 대소문자로만 구성되어 있으며, 대문자는 소문자보다 작은 것으로 간주합니다.

<입출력 예>

s	return
"Zbcdefg"	"gfedcbZ"

<제한 조건>

- str은 길이 1 이상인 문자열입니다.

02-2 문자열 처리 응용

3. 문자열 내 마음대로 정렬하기

<문제 설명>

문자열로 구성된 리스트 strings와, 정수 n이 주어졌을 때, 각 문자열의 인덱스 n번째 글자를 기준으로 오름차순 정렬하려 합니다. 예를 들어 strings가 ["sun", "bed", "car"]이고 n이 1이면 각 단어의 인덱스 1의 문자 "u", "e", "a"로 strings를 정렬합니다.

<제한 조건>

- strings는 길이 1 이상, 50이하인 배열입니다.
- strings의 원소는 소문자 알파벳으로 이루어져 있습니다.
- strings의 원소는 길이 1 이상, 100이하인 문자열입니다.
- 모든 strings의 원소의 길이는 n보다 큽니다.
- 인덱스 1의 문자가 같은 문자열이 여럿 일 경우, 사전순으로 앞선 문자열이 앞쪽에 위치합니다.

<입출력 예>

strings	n	return
["sun", "bed", "car"]	1	["car", "bed", "sun"]
["abce", "abcd", "cdx"]	2	["abcd", "abce", "cdx"]

입출력 예#1

"sun", "bed", "car"의 1번째 인덱스 값은 각각 "u", "e", "a" 입니다. 이를 기준으로 strings를 정렬하면 ["car", "bed", "sun"] 입니다.

입출력 예#2

"abce"와 "abcd", "cdx"의 2번째 인덱스 값은 "c", "c", "x"입니다. 따라서 정렬 후에는 "cdx"가 가장 뒤에 위치합니다. "abce"와 "abcd"는 사전순으로 정렬하면 "abcd"가 우선하므로, 답은 ["abcd", "abce", "cdx"] 입니다.

02-2 문자열 처리 응용

4. 이상한 문자 만들기

<문제 설명>

문자열 s는 한 개 이상의 단어로 구성되어 있습니다. 각 단어는 하나 이상의 공백문자로 구분되어 있습니다. 각 단어의 짝수번째 알파벳은 대문자로, 홀수번째 알파벳은 소문자로 바꾼 문자열을 리턴하는 함수, solution을 완성하세요.

<제한 조건>

- 문자열 전체의 짝/홀수 인덱스가 아니라, 단어(공백을 기준)별로 짝/홀수 인덱스를 판단해야합니다.
- 첫 번째 글자는 0번째 인덱스로 보아 짝수번째 알파벳으로 처리해야 합니다.

<입출력 예>

s	return
"try hello world"	"TrY HeLlO WoRlD"

입출력 예

"try hello world"는 세 단어 "try", "hello", "world"로 구성되어 있습니다. 각 단어의 짝수번째 문자를 대문자로, 홀수번째 문자를 소문자로 바꾸면 "TrY", "HeLlO", "WoRlD"입니다. 따라서 "TrY HeLlO WoRlD" 를 리턴합니다.

02-2 문자열 처리 응용

5. 시저 암호

<문제 설명>

어떤 문장의 각 알파벳을 일정한 거리만큼 밀어서 다른 알파벳으로 바꾸는 암호화 방식을 시저 암호라고 합니다. 예를 들어 "AB"는 1만큼 밀면 "BC"가 되고, 3만큼 밀면 "DE"가 됩니다. "z"는 1만큼 밀면 "a"가 됩니다. 문자열 s와 거리 n을 입력받아 s를 n만큼 민 암호문을 만드는 함수, solution을 완성해 보세요.

<입출력 예>

s	n	result
"AB"	1	"BC"
"z"	1	"a"
"a B z"	4	"e F d"

<제한 조건>

- 공백은 아무리 밀어도 공백입니다.
- s는 알파벳 소문자, 대문자, 공백으로만 이루어져 있습니다.
- s의 길이는 8000이하입니다.
- n은 1 이상, 25이하인 자연수입니다.

02-3 스택과 큐

스택과 큐의 핵심 이론

스택 개념

스택(Stack)은 삽입과 삭제 연산이 후입선출(LIFO : Last-in First-out)로 이뤄지는 자료구조다.

스택 개념의 일상 예

스택 개념의 일상 예

식판의 스택

최근에 쌓은 식판을 꺼낸다

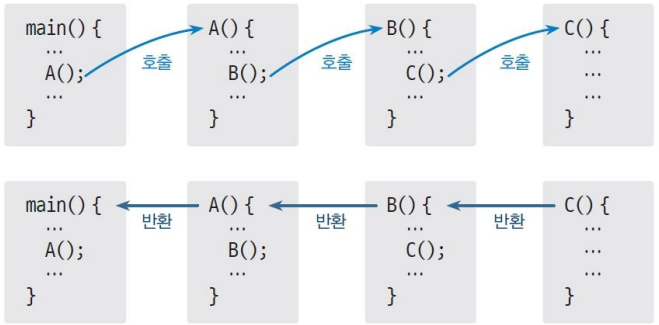
02-3 스택과 큐

스택과 큐의 핵심 이론

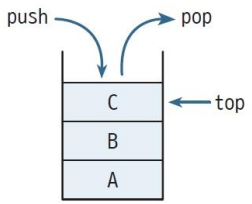
스택의 활용

스택의 활용(취소, 되돌리기)

- 키보드 입력을 하다가 백스페이스(←) 키를 누르면 최근에 입력한 글자를 지운다.
- 혼글, 워드, 파워포인트, 엑셀 등 모든 편집기는 최근에 한 작업순으로 취소하는 기능이 있다(보통 Ctrl+Z로 수행한다).
- 프로그램에서 함수 A가 함수 B를 호출하고 함수 B는 함수 C를 호출하는 호출 체인은 나중에 각 함수가 끝나면 돌아갈 때를 대비해서 호출 경로를 잘 관리해야 한다.



함수 호출 시 스택 사용 예



'←' in keyboard input line

예) abcd←←efgh←←←ij←km←

결과: abeik

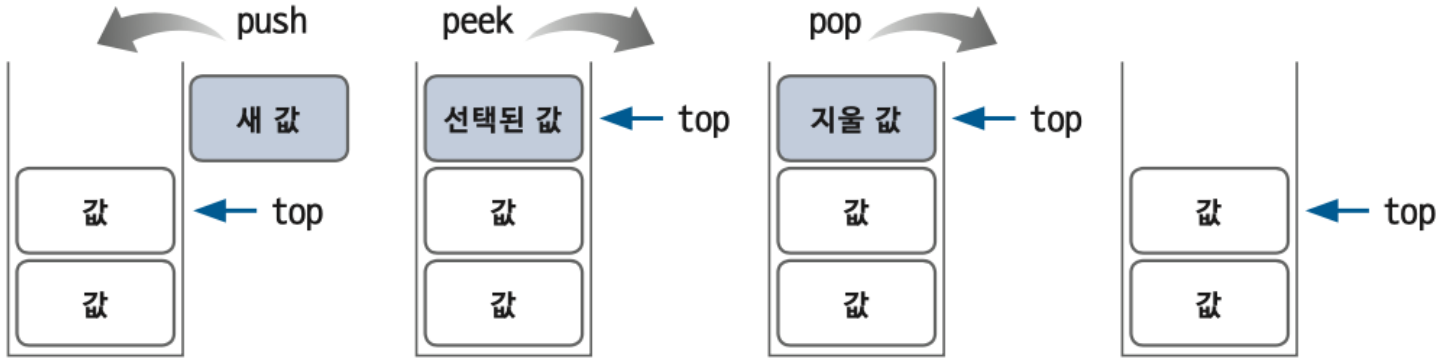
한 문자를 읽어 '←' 이 아니면 저장하고
'←' 이면 최근에 저장된 문자를 제거한다.

02-3 스택과 큐

스택과 큐의 핵심 이론

스택

스택은 깊이 우선 탐색(DFS: Depth First Search), 백트랙킹 종류의 코딩 테스트에서 효과적이다.



스택 연산 과정

02-3 스택과 큐

스택 용어

위치

- top: 삽입과 삭제가 일어나는 위치를 뜻한다.

연산

- push: top 위치에 새로운 데이터를 삽입하는 연산이다.
- pop: top 위치에 현재 있는 데이터를 삭제하고 확인하는 연산이다.
- peek: top 위치에 현재 있는 데이터를 단순 확인하는 연산이다.

02-3 스택과 큐

스택과 큐의 핵심 이론

큐 개념

큐(Queue)은 삽입과 삭제 연산이 후입선출(LIFO : Last-in First-out)로 이뤄지는 자료구조다.

큐 개념의 일상 예

큐 개념의 일상 예

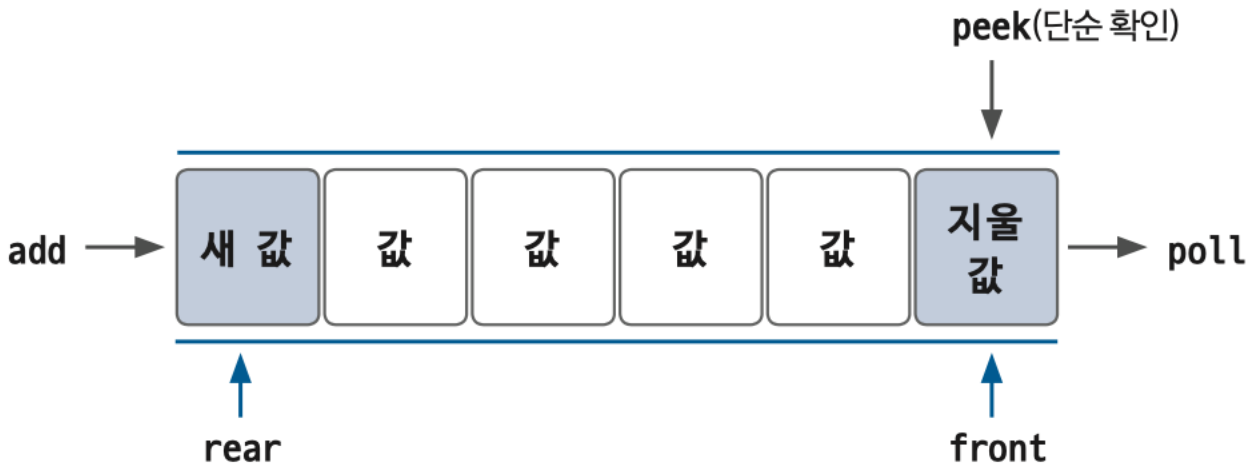
- 오래된 우유부터 먹기
- 식당에서 기다리는 사람들의 열
- Lotto 복권을 사려는 사람들의 열

02-3 스택과 큐

스택과 큐의 핵심 이론

큐

큐는 너비 우선 탐색(BFS: Breadth First Search)에서 자주 사용하므로 스택과 함께 잘 알아 두어야 한다.



큐 연산 과정

02-3 스택과 큐

큐 용어

- rear: 큐에서 가장 끝 데이터를 가리키는 영역이다.
- front: 큐에서 가장 앞의 데이터를 가리키는 영역이다.
- add: rear 부분에 새로운 데이터를 삽입하는 연산이다.
- poll: front 부분에 있는 데이터를 삭제하고 확인하는 연산이다.
- peek: 큐의 맨 앞(front)에 있는 데이터를 확인할 때 사용하는 연산이다.

02-3 스택과 큐

스택과 큐의 핵심 이론

스택(Stack)

- 최근에 삽입된 원소를 제거한다.
- LIFO (Last-In-First-Out)

큐(Queue)

- 가장 먼저 삽입된 원소를 제거한다.
- FIFO (First-In-First-Out)

02-3 스택과 큐

1. 스택 예제

<문제 설명>

배열 arr과 정수 n이 주어졌을 때, 배열의 값을 하나씩 스택에 넣고 n이 1이면 값 하나를 삭제한 뒤 스택에 남아 있는 값을 반환하고, n이 2이면 단순 확인 후 스택에 남아 있는 값을 확인하는 함수, solution을 완성해 보세요.

<입출력 예>

arr	n	return
[8,4,3,6]	1	[8, 4, 3]
[1,9,5,2,7]	2	[1, 9, 5, 2, 7]

<제한 조건>

- arr 은 길이 1 이상 50이하인 배열입니다.

02-3 스택과 큐

2. 카드 게임 (큐 예제)

<문제 설명>

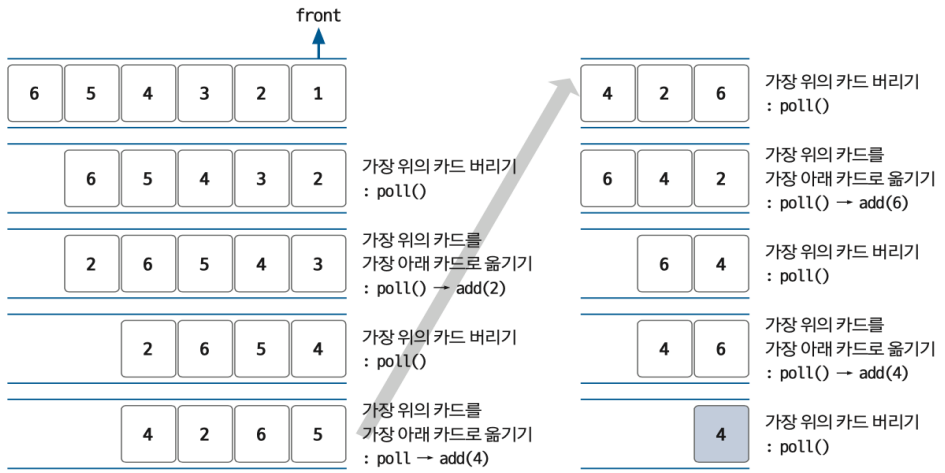
배열 arr이 주어졌을 때, 가장 위의 카드 하나를 버리고, 그 다음 카드 한 장을 가장 아래에 있는 카드 밑으로 옮기는 동작을 반복한다. 카드가 한 장 남게 되면 반환하는 함수, solution을 완성해 보세요. 큐의 선입선출 특징을 이용한다.

<제한 조건>

- 카드 개수의 최대는 500,000이다.

<입출력 예>

arr	return
[6,5,4,3,2,1]	[4]



03

정렬 알고리즘

03-1 정렬 알고리즘

03-2 자바의 정렬

03-1 자료 정렬

- **자료 정렬이란**
  - 자료를 원하는 기준에 맞게 순서를 재배치
  - 자료 처리의 효율성을 향상시키기 위한 작업
- **자료 정렬의 장점 ⇒ 자료 정렬은 자료 처리의 기본에 해당**
  - 많은 양의 자료 속에서 원하는 정보를 찾고자 할 때 자료가 정렬되어 있다면 훨씬 빠르게 처리할 수 있음
  - 자료의 생성은 자료의 내용을 컴퓨터에 저장한 후 효율적으로 사용하기 위함
    - 자료의 양이 방대할 때 원하는 자료를 찾아서 처리해야 한다면, 자료 정렬은 자료 검색을 위한 우선 단계
  - 자료 정렬을 통하여 필요한 자료를 빠르게 검색하고, 검색된 자료를 대상으로 원하는 작업을 실행하는 것이 컴퓨터에서 자료를 처리하는 가장 효율적 방법

03-1 정렬 및 효율성 기준

- **순서 없이 나열된 자료를 정렬하기 위해서는 정렬 기준을 정해야 함**
  - 오름차순(ascending order)
  - 내림차순(descending order)
- **여러 가지 자료 정렬 알고리즘 가운데 주어진 상황에 가장 적합한 알고리즘을 선택할 시 고려 사항**

① 자료의 양

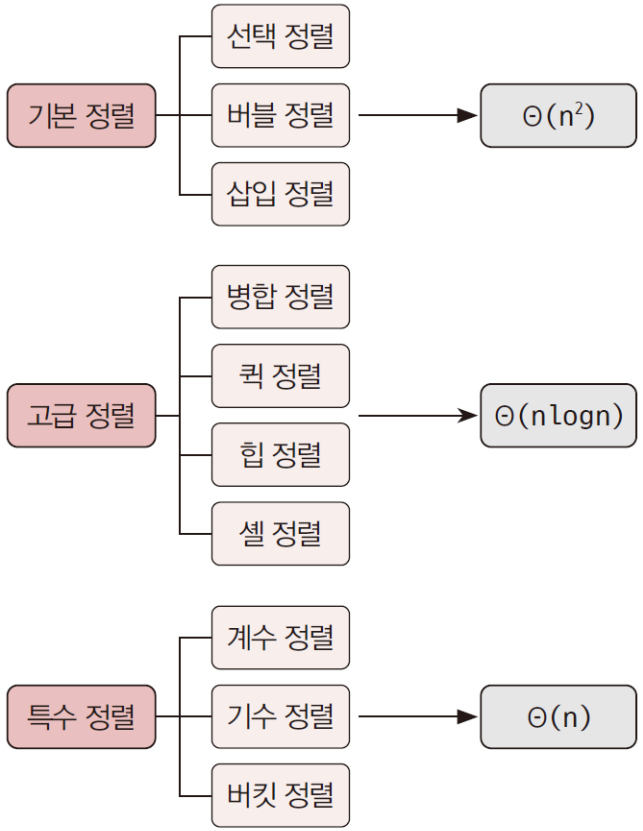
② 사용 가능한 기억 공간의 크기

③ 정렬을 위한 자료 이동 빈도 수

03-1 정렬 알고리즘 종류

정렬 알고리즘 정의

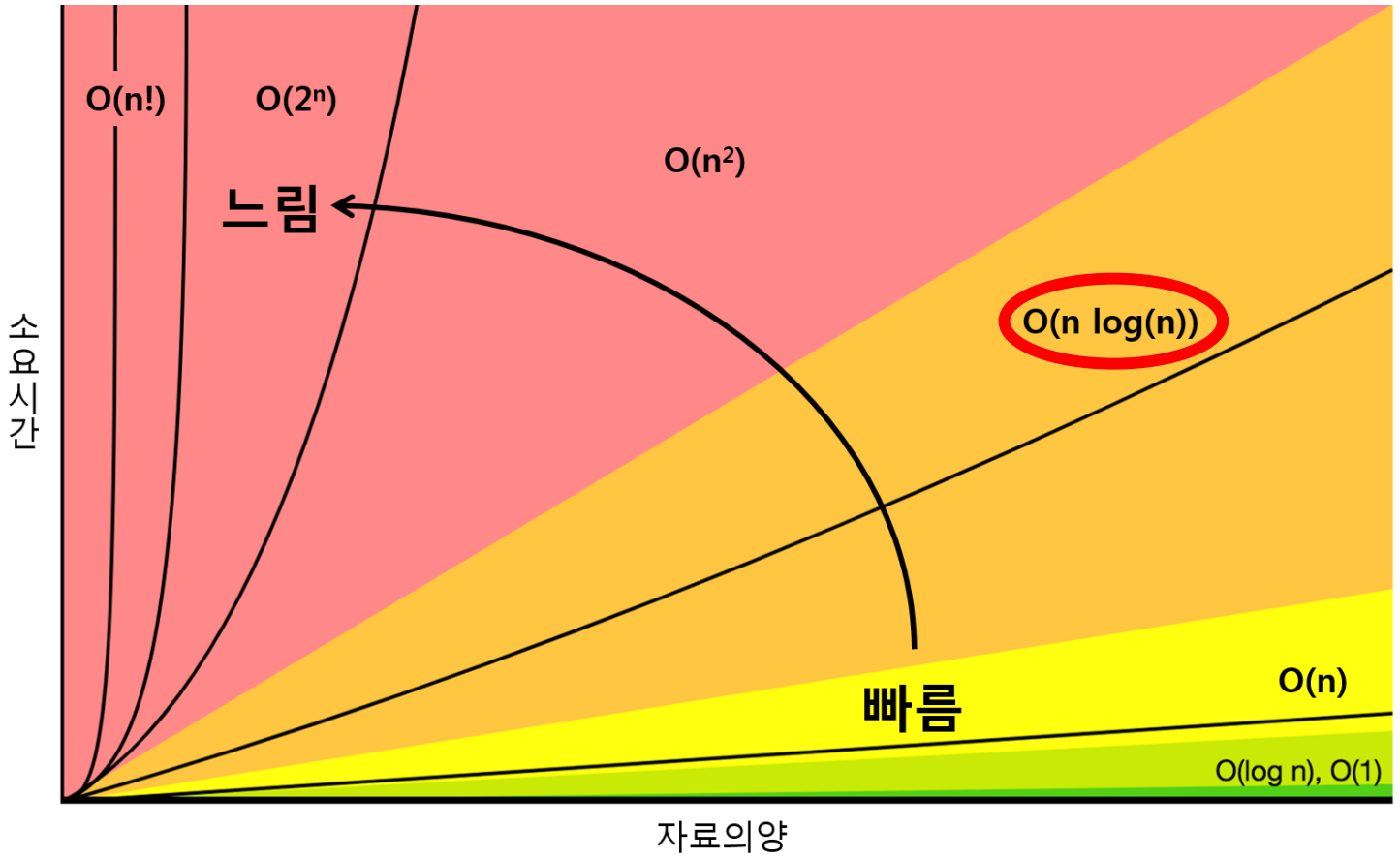
정렬 알고리즘	정의
버블bubble	데이터의 인접 요소끼리 비교하고, swap 연산을 수행하며 정렬하는 방식
선택selection	대상에서 가장 크거나 작은 데이터를 찾아가 선택을 반복하면서 정렬하는 방식
삽입insertion	대상을 선택해 정렬된 영역에서 선택 데이터의 적절한 위치를 찾아 삽입하면서 정렬하는 방식
퀵quick	pivot 값을 선정해 해당 값을 기준으로 정렬하는 방식
병합merge	이미 정렬된 부분 집합들을 효율적으로 병합해 전체를 정렬하는 방식
기수radix	데이터의 자릿수를 바탕으로 비교해 데이터를 정렬하는 방식



정렬 알고리즘과 평균 성능

03-1 정렬 알고리즘 시간 복잡도 그래프

정렬 알고리즘	최악 시간복잡도
버블정렬	$O(n^2)$
선택정렬	$O(n^2)$
삽입정렬	$O(n^2)$
퀵정렬	$O(n^2)$
병합정렬	O(n log(n))
기수정렬	O(wn)



03-2 정렬

1. 나누어 떨어지는 숫자 배열

<문제 설명>

array의 각 element 중 divisor로 나누어 떨어지는 값을 오름차순으로 정렬한 배열을 반환하는 함수, solution을 작성해주세요.

divisor로 나누어 떨어지는 element가 하나도 없다면 배열에 -1을 담아 반환하세요.

<제한 조건>

- arr은 자연수를 담은 배열입니다.
- 정수 i, j에 대해 i ≠ j 이면 arr[i] ≠ arr[j] 입니다.
- divisor는 자연수입니다.
- array는 길이 1 이상인 배열입니다.

<입출력 예>

arr	divisor	return
[5, 9, 7, 10]	5	[5, 10]
[2, 36, 1, 3]	1	[1, 2, 3, 36]
[3,2,6]	10	[-1]

입출력 예#1
arr의 원소 중 5로 나누어 떨어지는 원소는 5와 10입니다. 따라서 [5, 10]을 리턴합니다.
입출력 예#2
arr의 모든 원소는 1으로 나누어 떨어집니다. 원소를 오름차순으로 정렬해 [1, 2, 3, 36]을 리턴합니다.
입출력 예#3
3, 2, 6은 10으로 나누어 떨어지지 않습니다. 나누어 떨어지는 원소가 없으므로 [-1]을 리턴합니다.

03-2 정렬

2. 예산(1)

<문제 설명>

S사에서는 각 부서에 필요한 물품을 지원해 주기 위해 부서별로 물품을 구매하는데 필요한 금액을 조사했습니다. 그러나, 전체 예산이 정해져 있기 때문에 모든 부서의 물품을 구매해 줄 수는 없습니다. 그래서 최대한 많은 부서의 물품을 구매해 줄 수 있도록 하려고 합니다.
물품을 구매해 줄 때는 각 부서가 신청한 금액만큼을 모두 지원해 줘야 합니다. 예를 들어 1,000원을 신청한 부서에는 정확히 1,000원을 지원해야 하며, 1,000원보다 적은 금액을 지원해 줄 수는 없습니다.
부서별로 신청한 금액이 들어있는 배열 d와 예산 budget이 매개변수로 주어질 때, 최대 몇 개의 부서에 물품을 지원할 수 있는지 return 하도록 solution 함수를 완성해주세요.

<제한 조건>

- d는 부서별로 신청한 금액이 들어있는 배열이며, 길이(전체 부서의 개수)는 1 이상 100 이하입니다.
- d의 각 원소는 부서별로 신청한 금액을 나타내며, 부서별 신청 금액은 1 이상 100,000 이하의 자연수입니다.
- budget은 예산을 나타내며, 1 이상 10,000,000 이하의 자연수입니다.

<입출력 예>

d	budget	result
[1,3,2,5,4]	9	3
[2,2,3,3]	10	4

03-2 정렬

2. 예산(2)

입출력 예#1

각 부서에서 [1원, 3원, 2원, 5원, 4원]만큼의 금액을 신청했습니다. 만약에, 1원, 2원, 4원을 신청한 부서의 물품을 구매해주면 예산 9원에서 7원이 소비되어 2원이 남습니다. 항상 정확히 신청한 금액만큼 지원해 줘야 하므로 남은 2원으로 나머지 부서를 지원해 주지 않습니다. 위 방법 외에 3개 부서를 지원해 줄 방법들은 다음과 같습니다.

1원, 2원, 3원을 신청한 부서의 물품을 구매해주려면 6원 필요
1원, 2원, 5원을 신청한 부서의 물품을 구매해주려면 8원 필요
1원, 3원, 4원을 신청한 부서의 물품을 구매해주려면 8원 필요
1원, 3원, 5원을 신청한 부서의 물품을 구매해주려면 9원 필요
3개 부서보다 더 많은 부서의 물품을 구매해 줄 수는 없으므로 최대 3개 부서의 물품을 구매해 줄 수 있습니다.

입출력 예#2

모든 부서의 물품을 구매해주면 10원이 됩니다. 따라서 최대 4개 부서의 물품을 구매해 줄 수 있습니다.

03-2 정렬

3. K번째수

<문제 설명>

배열 array의 i번째 숫자부터 j번째 숫자까지 자르고 정렬했을 때, k번째에 있는 수를 구하려 합니다.

예를 들어 array가 [1, 5, 2, 6, 3, 7, 4], i = 2, j = 5, k = 3이라면

- array의 2번째부터 5번째까지 자르면 [5, 2, 6, 3]
- 1에서 나온 배열을 정렬하면 [2, 3, 5, 6]
- 2에서 나온 배열의 3번째 숫자는 5

배열 array, [i, j, k]를 원소로 가진 2차원 배열 commands가 매개변수로 주어질 때, commands의 모든 원소에 대해 앞서 설명한 연산을 적용했을 때 나온 결과를 배열에 담아 return 하도록 solution 함수를 작성해주세요.

<제한 조건>

- array의 길이는 1 이상 100 이하입니다.
- array의 각 원소는 1 이상 100 이하입니다.
- commands의 길이는 1 이상 50 이하입니다.
- commands의 각 원소는 길이가 3입니다.

<입출력 예>

array	commands	return
[1, 5, 2, 6, 3, 7, 4]	[[2, 5, 3], [4, 4, 1], [1, 7, 3]]	[5, 6, 3]

[1, 5, 2, 6, 3, 7, 4]를 2번째부터 5번째까지 자른 후 정렬합니다. [2, 3, 5, 6]의 세 번째 숫자는 5입니다.
[1, 5, 2, 6, 3, 7, 4]를 4번째부터 4번째까지 자른 후 정렬합니다. [6]의 첫 번째 숫자는 6입니다.
[1, 5, 2, 6, 3, 7, 4]를 1번째부터 7번째까지 자릅니다. [1, 2, 3, 4, 5, 6, 7]의 세 번째 숫자는 3입니다.

오늘도 수고하셨습니다.