KOSA
Korea Software Industry Association

알고리즘으로 배우는 Python 과정

2021. 12. 13 ~ 16

정 준 수 Ph.D

과정 목표

프로그래밍 언어 Python를 활용하여 SW 알고리즘의 구현기술 습득

1. Python 언어특성과 기본 문법 익히기
2. 머신러닝에 필요한 Data Type 변환
3. 객체지향 프로그램 구현과 함수형 언어
4. Python 라이브러리를 이용한 고급 알고리즘 구현
5. Tree Algorithm와 Graph 구조 등을 이용한 다양한 머신러닝 알고리즘 응용 사례 구현

Python 자료형이란?

- 일반적으로 프로그램은 연산을 수행하며, 연산을 하기 위해서는 값들을 임시로 혹은 영구적으로 어딘가에 저장해 놔야 함
- 이러한 값들을 저장하는 저장소를 변수라고 함
- 변수의 유형을 자료형이라고 함

Python 변수명

- 문자, 숫자, 밑줄(_)로 구성
  숫자는 처음에 나올 수 없음
- 대소문자 구분
- 예약어 사용 불가

```
>>> friend = 10
>>> Friend = 1
>>> friend
10
>>> Friend
1
```

and, as, assert, break, class, continue, def, del, elif, else, except, is, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, yield

Python 자료의 종류

- 수치 - int float complex 등
- 문자형
- 리스트 – 리스트는 쉽게 값들의 나열
- 세트 – 집합의 개념으로 순서가 없음
- 튜플 – tuple은 리스트와 유사하나, 읽기 전용임
- 딕셔러리 – 키와 값의 쌍으로 이루어져 있음
- 부울(bool) – 참(True)와 거짓(False)를 나타내는 자료형

함수의 정의

- 함수의 선언은 def로 시작하고 콜론(:)으로 끝남
- 함수의 시작과 끝은 코드의 들여쓰기로 구분
- 시작과 끝을 명시해 줄 필요가 없음

함수 선언 문법

```
def <함수명>(인수1, 인수2, … 인수N):
    <구문>
    return<반환값>
```

간단한 함수 선언해 보기

입력 받은 2개의 인수를 서로 곱한 값을 리턴함

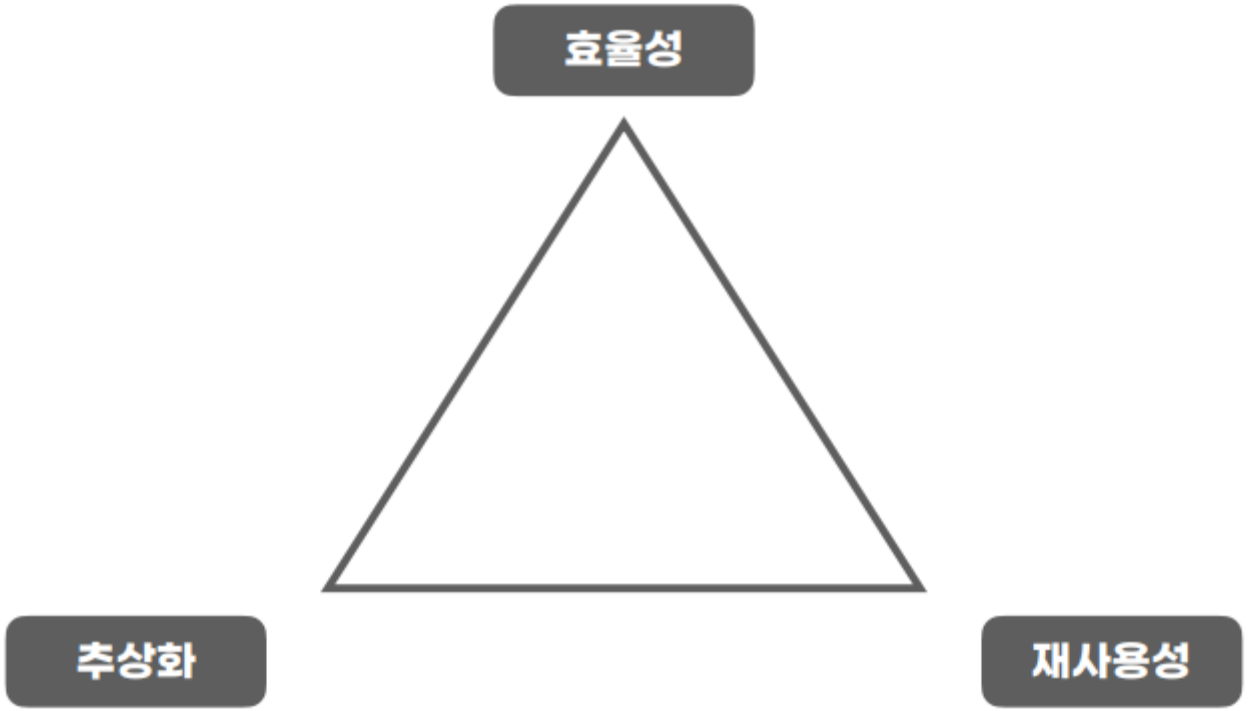
```
def Times(a, b):
    return a*b

>>> Times(10, 10)
100
```

<Python의 유용한 함수 예제>

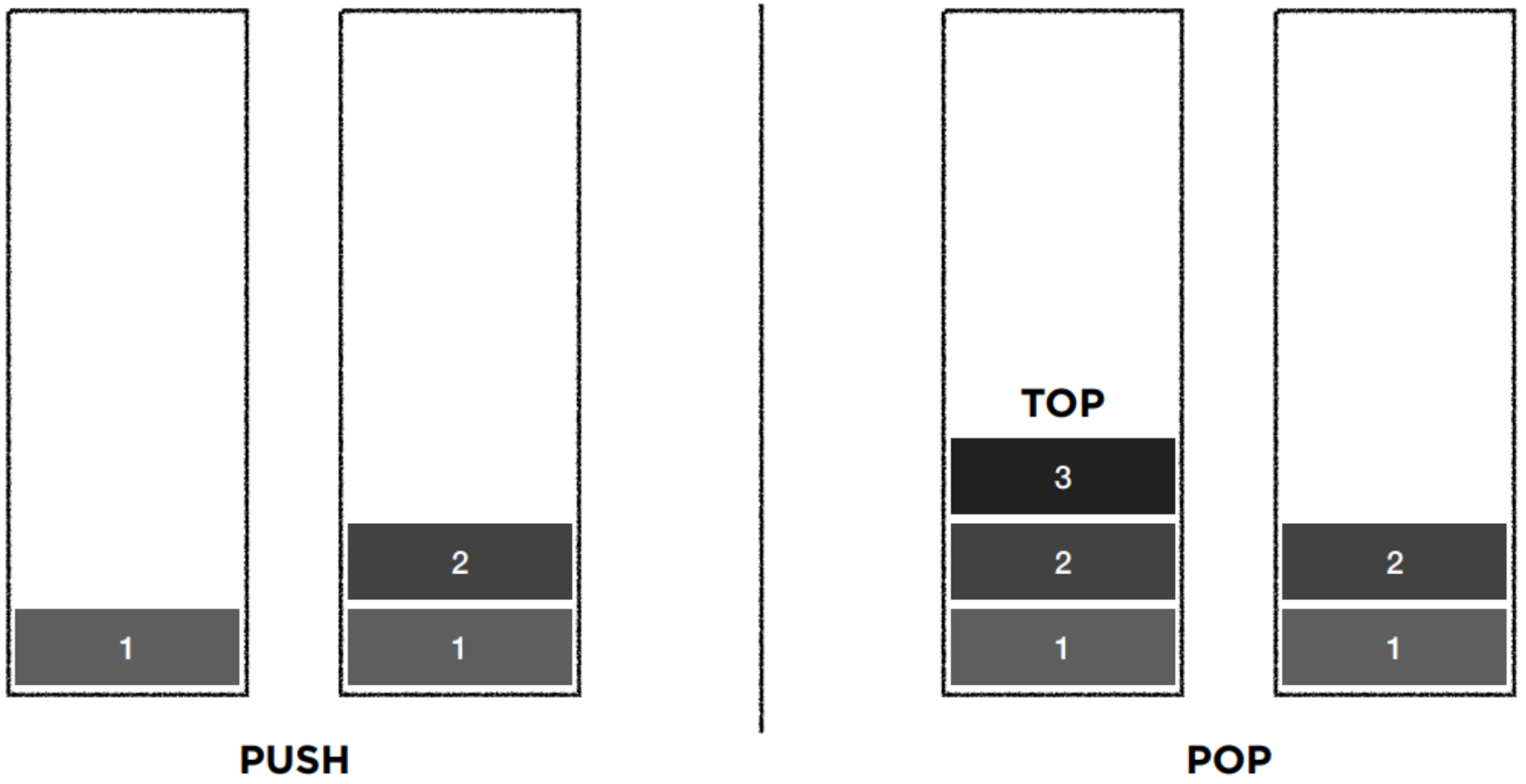
https://github.com/JSJeong-me/KOSA-Python-Advance

자료구조와 알고리즘

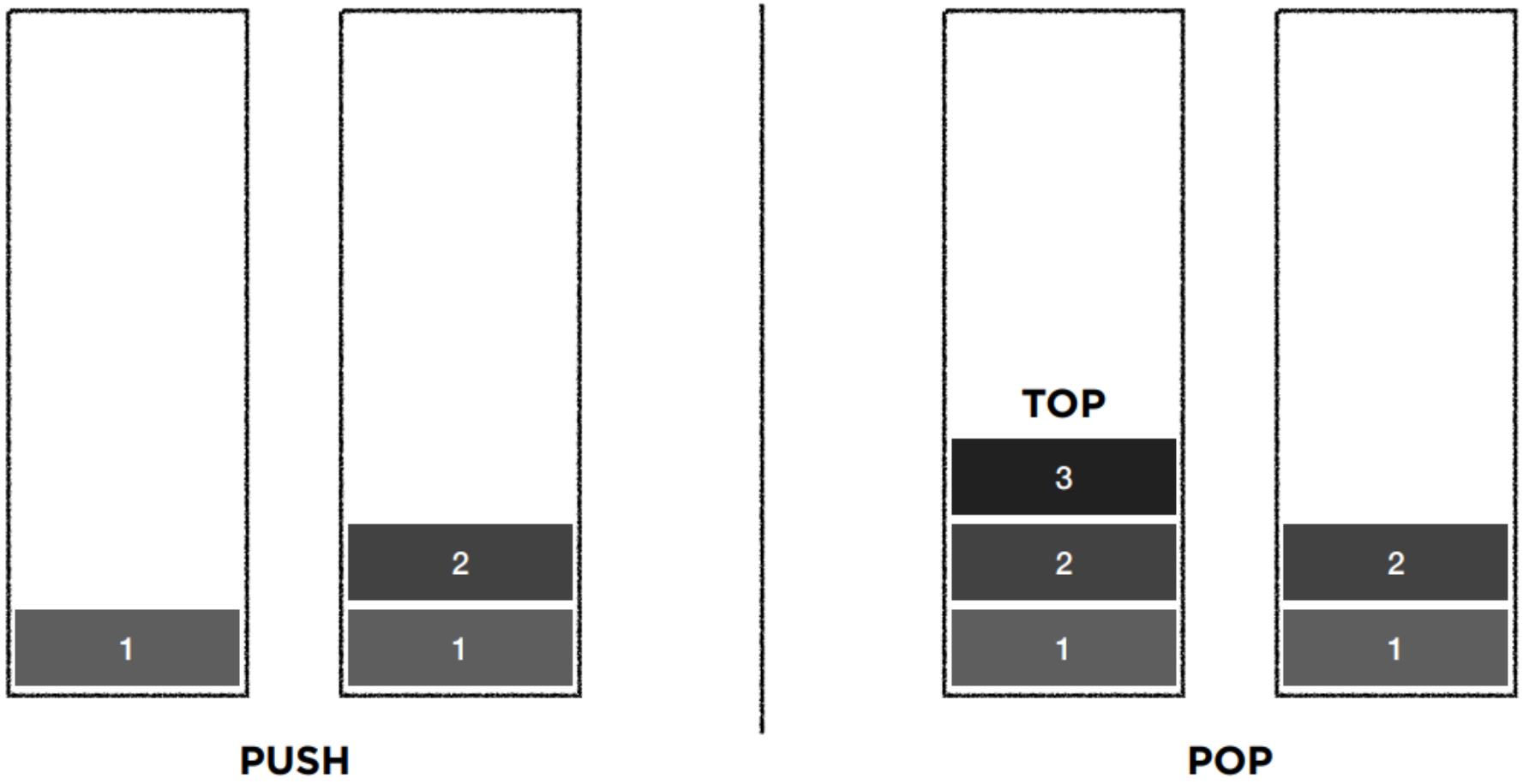


자료를 저장하는 방법론, 규칙

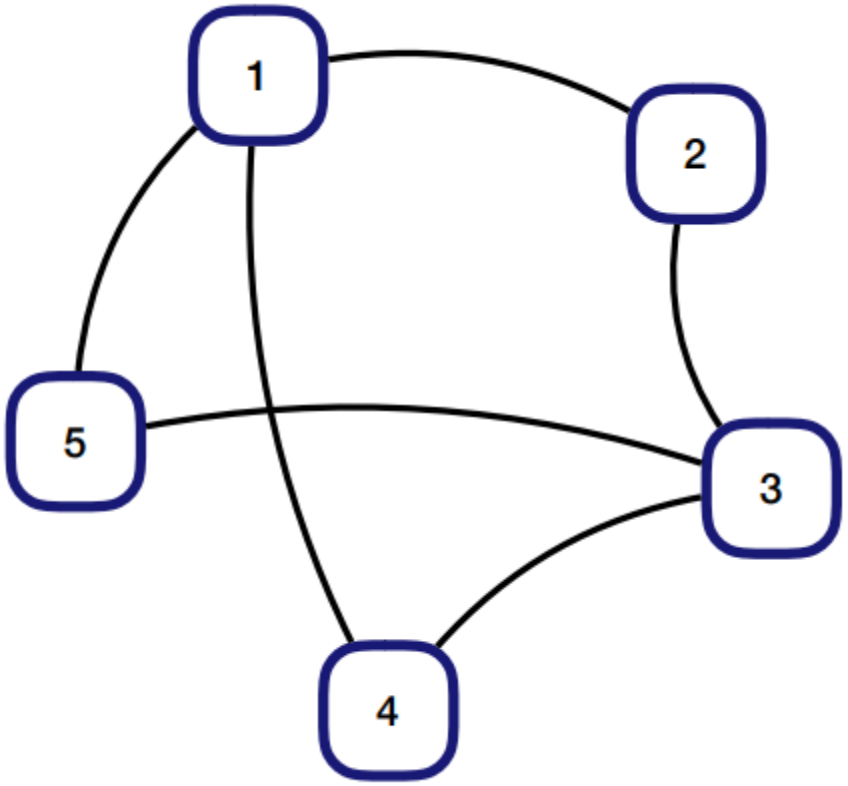
Stack: FILO(First In, Last Out)



Queue: FIFO(First In, Fast Out)



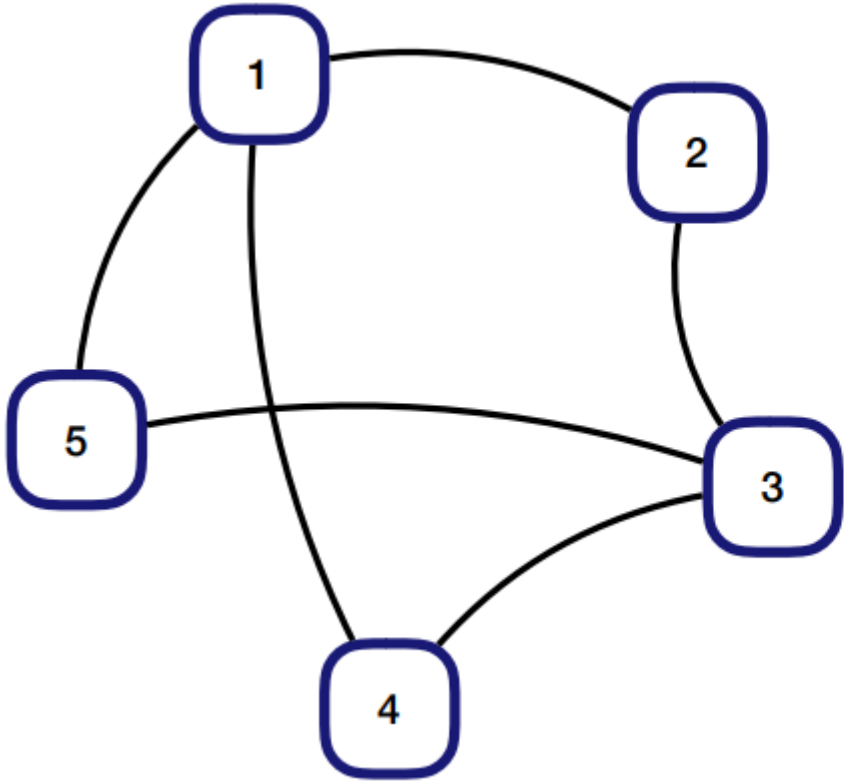
Graph: 관계를 위한 자료구조



노드(Node)와 간선(Edge)으로 구성
너와 나의 연결고리

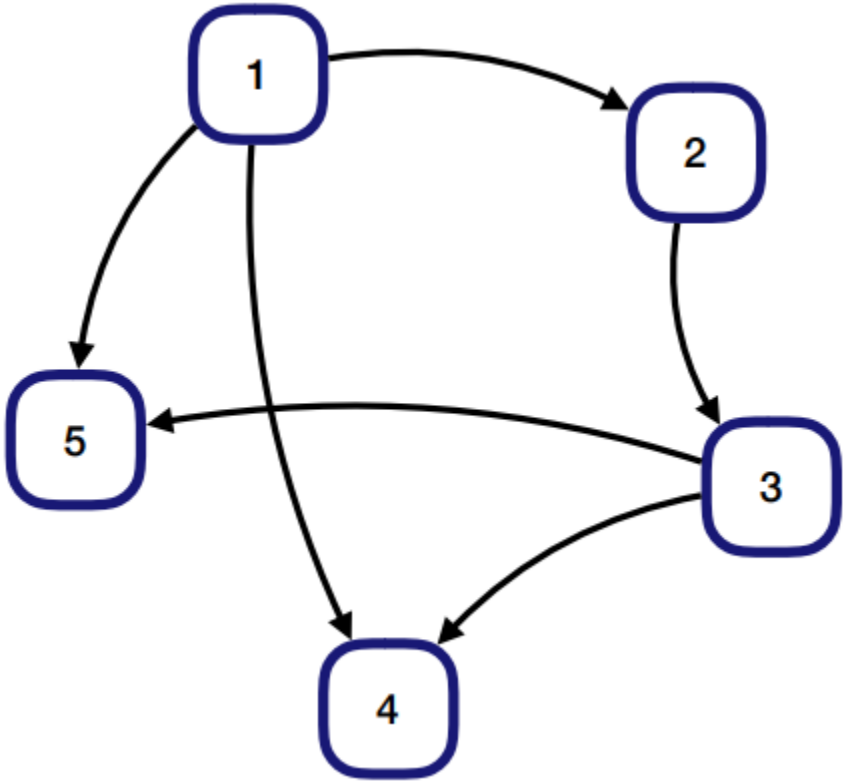
들어오는 간선 수를 **indegree**
나가는 간선 수를 **outdegree**

Graph: 저장하는 방법 / 인접 행렬



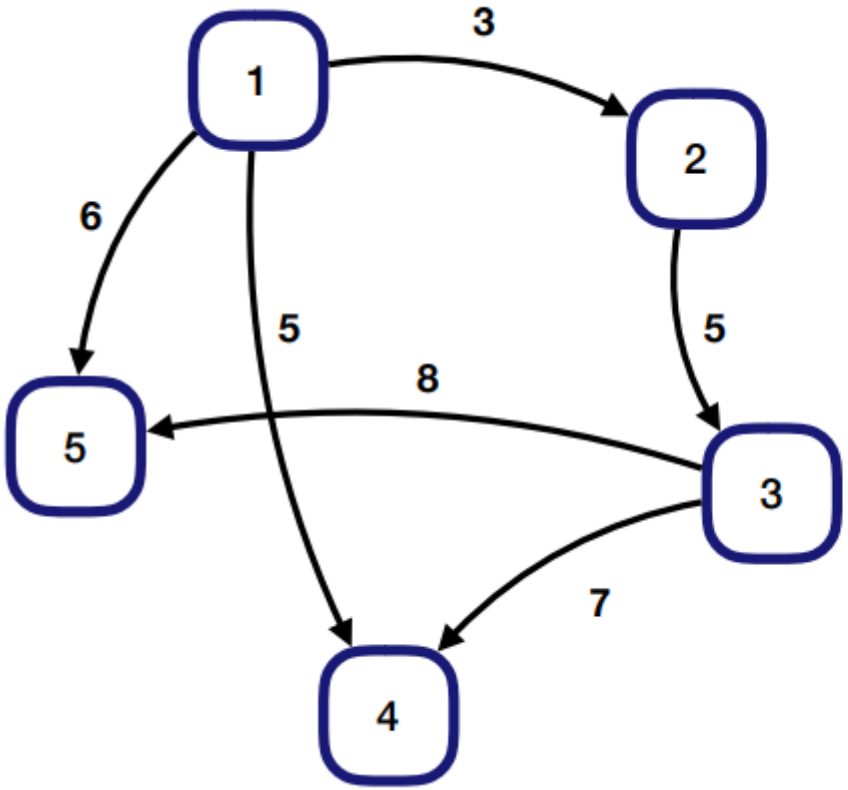
	1	2	3	4	5
1	0	1	0	1	1
2	1	0	1	0	0
3	0	1	0	1	1
4	1	0	1	0	0
5	1	0	1	0	0

Graph: 방향을 추가하면?



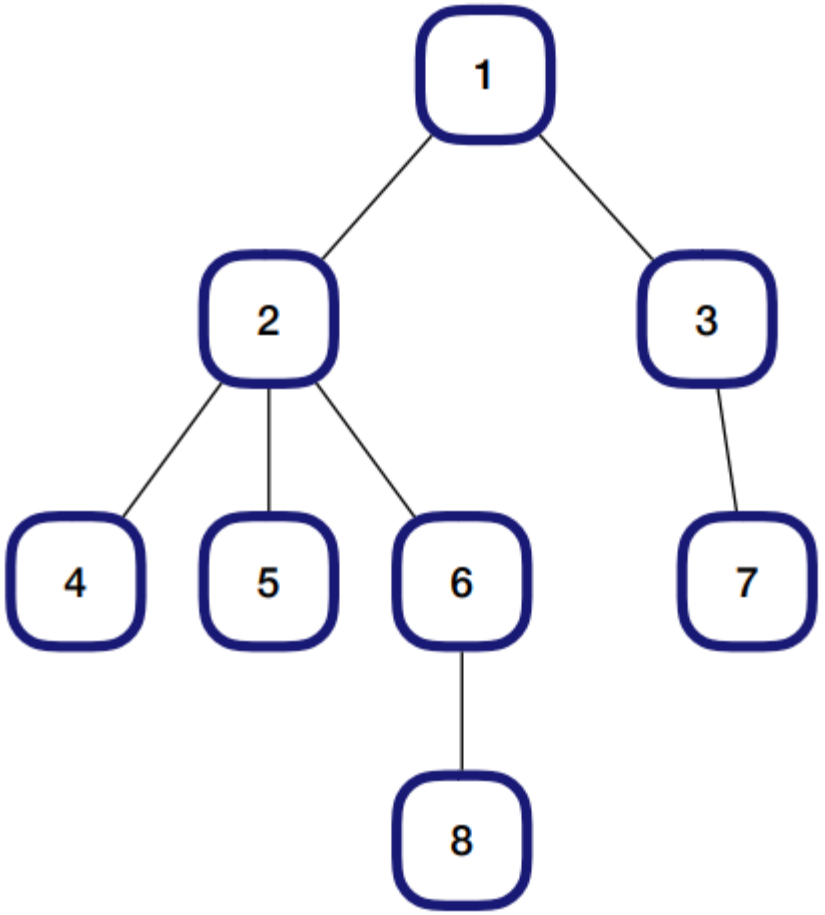
	1	2	3	4	5
1	0	1	0	1	1
2	0	0	1	0	0
3	0	0	0	1	1
4	0	0	0	0	0
5	0	0	0	0	0

Graph: 가중치를 추가하면



	1	2	3	4	5
1	0	3	0	5	6
2	0	0	5	0	0
3	0	0	0	7	8
4	0	0	0	0	0
5	0	0	0	0	0

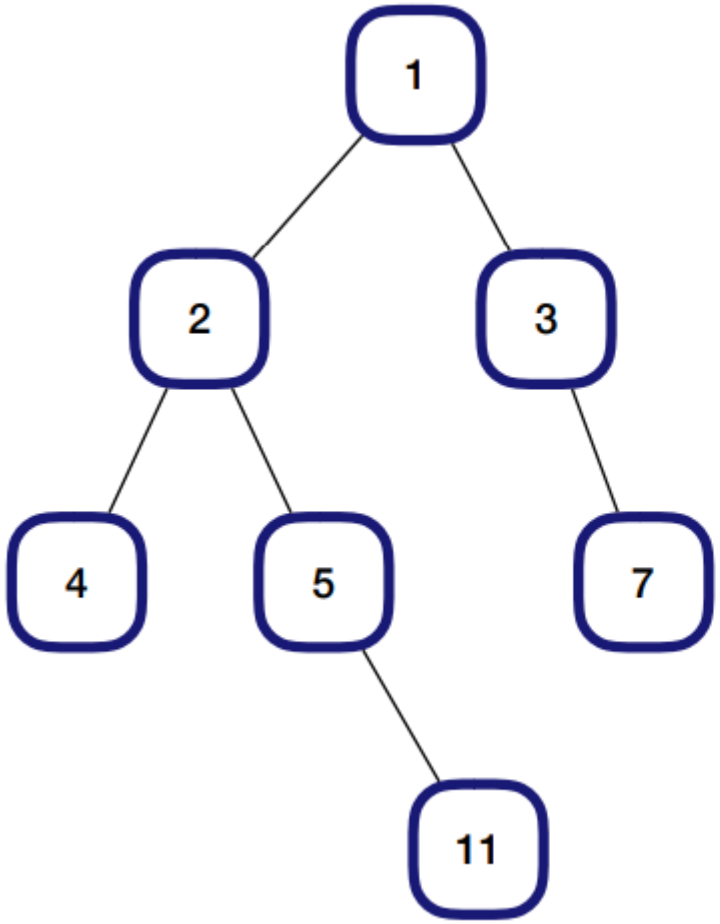
Tree: 특수한 구조를 가진 그래프



그래프이기에 똑같이 **노드(Node)와 간선(Edge)**으로 구성

상위/하위 관계를 나눌 수 있음

Binary Tree: 이진 트리



부모 노드가 X이면, 왼쪽 노드가 2X 오른쪽 노드가 2X+1
Heap과 BST에 대해 알아봅시다.

<실습 예제>

https://github.com/JSJeong-me/KOSA-ML-BigData-Analysis/tree/main/decision_tree

시간 복잡도: Time Complexity

연산에 따라 속도는 모두 같을까? + 연산과 * 연산은 같을까?

모든 연산을 Counting 할 수 있을까? break 등의 생략은?

최악의 경우와 최선의 경우? 수열의 정렬에서 최악과 최선?

Counting에 따라 실행시간은? 서버와 컴퓨터의 속도 차?

시간 복잡도: Big-O Notation

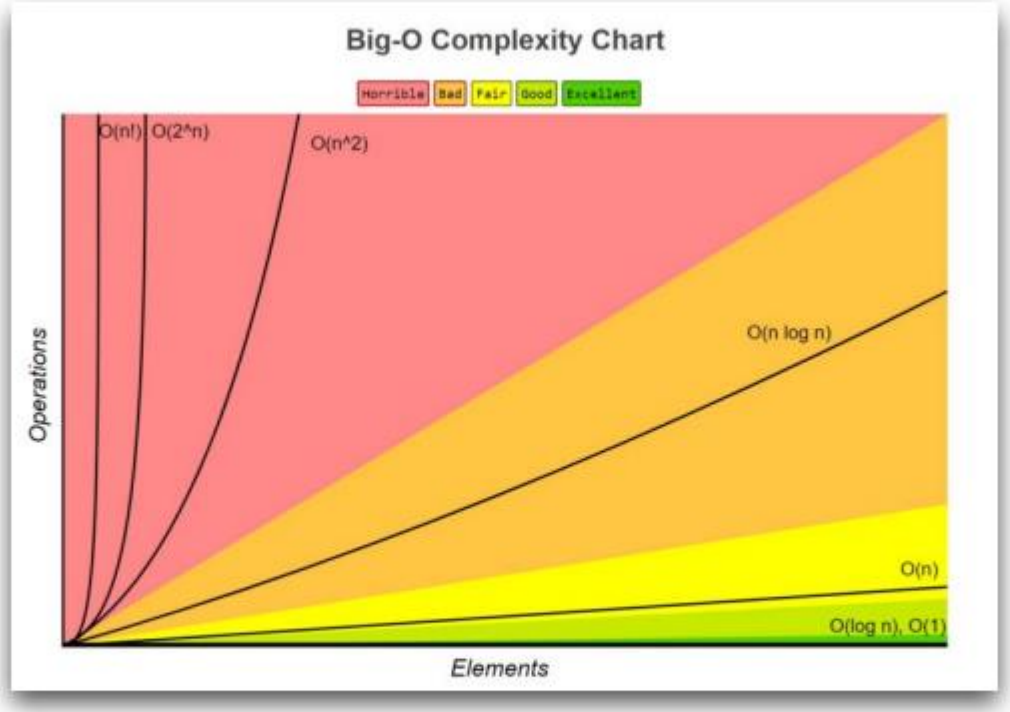
대충 계산하자 => 점근적 표기법

입력이 N일때, 연산 횟수가 최악이 2N^2 + 4N 이라면??

N이 무한대로 커질 때, 증가에 미치는 영향은 가장 큰 항만 필요하다!

O(N^2)

시간 복잡도: Big-O Notation

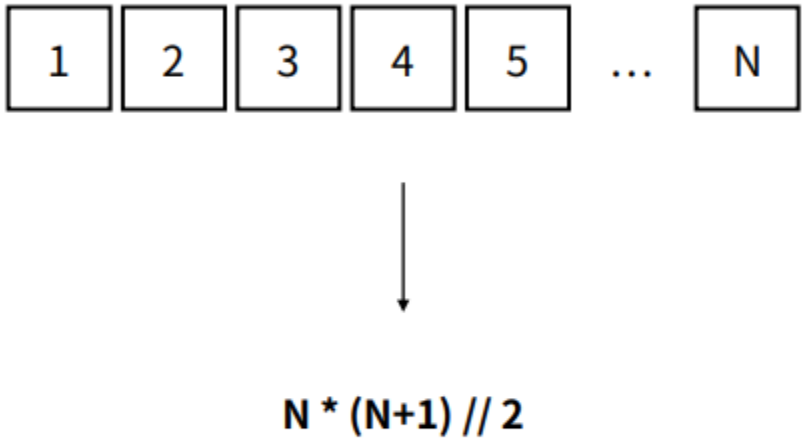


$$O(n!) > O(2^N) > O(N^2) > O(NlogN) > O(N) > O(\sqrt{N}) > O(logN) > O(1)$$

시간 복잡도: O(1) 예시

Q. 1부터 N까지 합을 구하시오.

```
def sum_N(N):
    return N*(N+1)//2
```

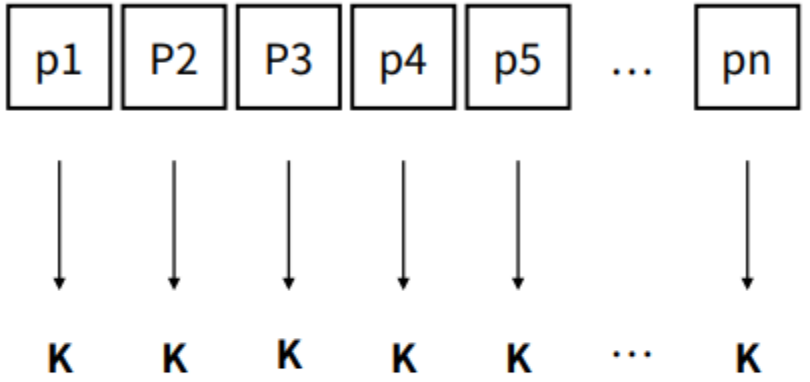


단순한 수식으로 표현 가능

시간 복잡도: O(N) 예시

Q. 길이 N 수열에서 수 K 찾기 (sum, min, max도 유사)

```
def search(lst, N, K):
    for i in lst:
        if i == lst : return True
    return False
```



최악의 경우 N번을 돌려봐야함
확률적으로는 N/2번

시간 복잡도: O(logN) 예시

계속 길이가 절반으로 줄어듬 $N/2/\cdots/2 \simeq 1 \Rightarrow N \simeq 2^k \Rightarrow k = log_2 N$

2라는 상수를 때면… $O(log_2 N) = O(logN)$

```
def binary_search(lst, N, K):
    lo, hi = 0, N-1
    while lo <= hi :
        mid = (lo + hi) // 2
        if lst[mid] == K: return True
        if lst[mid] > K: hi = mid-1
        else : lo = mid+1
    return False
```

유클리드호제법: 최대공약수와 최소공배수

최대공약수란 공통적인 약수 중 최댓값
최대공약수가 1이면 **서로소**

최소공배수는 다음 공식으로 구할 수 있음
공통된 배수 중 최솟값

$$LCM(A, B) = A \times B / GCD(A, B)$$

우리는 GCD만 잘 구하면 LCM은 O(1)에 구할 수 있다

유클리드호제법: 최대공약수와 최소공배수

혹시 유클리드 호제법 아시는 분??

$$GCD(A, B) = GCD(B, A \,\%\, B)$$

```
def gcd(a, b):
    return b if a%b==0 else gcd(b, a%b)
```

<교안 및 실습 예제 프로그램>

https://github.com/JSJeong-me/KOSA-Python_Algorithm

강사 소개

정 준 수 / Ph.D (heinem@naver.com)

- 前) 삼성전자 연구원
- 前) 삼성의료원 (삼성생명과학연구소)
- 前) 삼성SDS (정보기술연구소)
- 現) (사)한국인공지능협회, AI, 머신러닝 강의
- 現) 한국소프트웨어산업협회, AI, 머신러닝 강의
- 現) 서울디지털재단, AI 자문위원
- 現) 한성대학교 교수(겸)
- 전문분야: 시각 모델링, 머신러닝(ML), RPA
- https://github.com/JSJeong-me/