Day 02

[Stack]

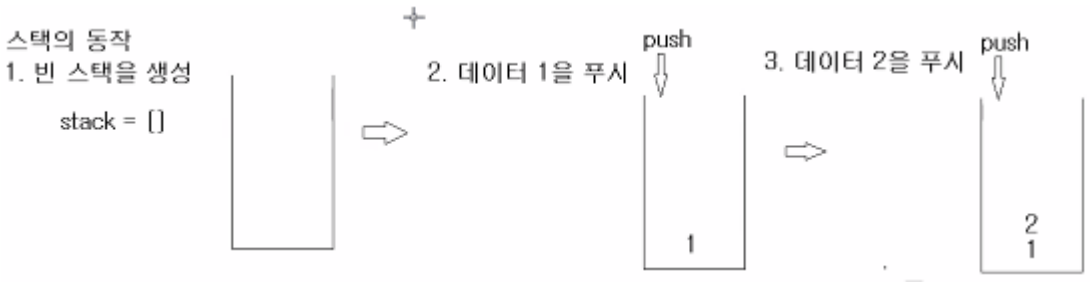
Stack

- “쌓는다” 라는 의미
- 임시 저장 장소
    - 데이터를 변경하겠다는 의미
- **대표적으로 임시 저장장소의 의미는 list, stack, queue**
    - 배열
        1. 정적 배열
            - 한번 선언시 끝
        2. 동적 배열 (파이썬에서는 대표적으로 사용)
            - 유동적 (사용자가 배열의 크기를 조절가능)
        - 배열의 크기(길이값)

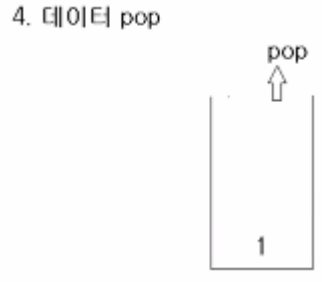
- 제일 먼저 데이터가 놓여질 장소를 선언(비어있는 스택을 생성)
- (FILO, LIFO)
    - 데이터 삽입하는 연산 => push()
    - 데이터 꺼내는 연산 => pop()

스택의 동작

1. 빈 스택을 생성 stack = []
2. 데이터 1을 푸시 push(stack, 1)
3. 데이터 2를 푸시 push(stack, 2)



4. 데이터 pop pop(stack, 2)



스택의 ADT (추상 자료형), abstract data type

- 특정 함수를 정의해서 사용자가 원하는 형식으로 이루어진다.
- 인터페이스만 존재하고 실제 구현은 되지 않은 자료형을 의미한다.
- ex)
    - play()
    - 플레이가 무엇을 한다고 정의하지 않았기 때문에 ADT다.
    - 모든 언어에 따라 표준 라이브러리에서 “스택” 제공 여부는 다르다.
    - 파이썬에서 스택을 제공하지 않는다.
    - 그러다보니 대안으로 리스트 메서드인 append() , push() 메서드로 스택을 대체한다.

*****덱(deque) -양방향에서 데이터를 삽입, 삭제**

- 시험문제에서는 100% 이걸 이용한다. (코테)
- Collection이 가지고 있다.
- 덱은 스택과 다르지만 댁을 스택처럼 이용할수 있다.
    - 약간의 응용이 필요하다.
- 어느 한쪽방향으로만 데이터를 삽입, 삭제하는 스택과 다르게 양쪽 방향에서 데이터를 삽입, 삭제 할수 있는 자료구조를 말한다.

스택의 연산

1. (void) push(dataType, date)
    a. 스택에 데이터 푸시
2. (dataType) pop
    a. 스택에서 최근에 푸시한 데이터를 제거하고, 데이터를 반환한다.
3. (boolean) isFull(가득 찼는지 확인)
    a. 스택에 들어있는 데이터의 개수가 max인지 확인
    b. 가득 찼다면 True, 그렇지 않다면 False
4. (boolean) isEmpty(비어 있는지 확인)
    a. 스택에 들어 있는 데이터가 하나도 없는지 확인
    b. 데이터가 하나라도 있으면 False, 그렇지 않으면 True

스택의 상태

1. int top(기본적으로 -1) (스택에서 최근에 삽입한 데이터의 위치를 저장하는 변수)
    a. 스택에서 최근에 푸시한 데이터의 위치를 기록
2. dataType[maxsize]
    a. 스택의 데이터를 관리하는 배열

```
stack = []
maxSize = 10

def isFull(stack):
    return len(stack) == maxSize

def isEmpty(stack):
    return len(stack) == 0

def push(stack, item):
    if isFull(stack):
        print("가득 찼습니다.")
    else:
        print("데이터 추가완료")

def pop(stack):
    if isEmpty(stack):
        print("비어있습니다.")
        return None
    else:
        return stack.pop()
```

리스트는 동적으로 사용하기 때문에 maxSize, isFull(), isEmpty() 메서드를 실제로는 구현하지 않는다.

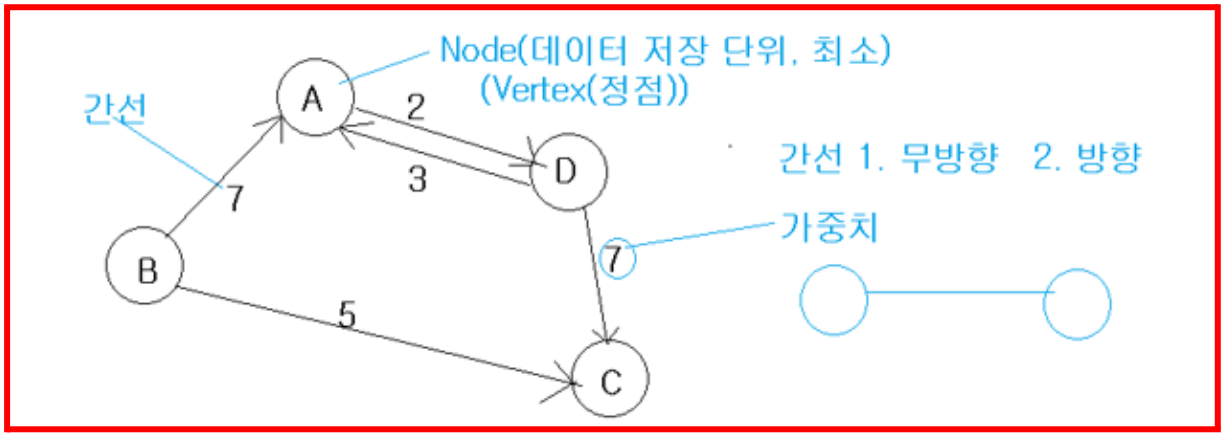
stack.append(item)

```
stack = []

def push(stack, item):
    stack.append(item)
    print("데이터 추가완료")

def pop(stack):
    if len(stack) == 0:
        print("비어있습니다.")
        return None
    else:
        return stack.pop()
```

비선형 구조 (Non-Linear Structure)

비선형 리스트 (나가는 길이 직진이 아니라 순서가 정해저 있지 않다)

- **트리(Tree), 그래프(Graph)**
  - 데이터를 탐색 하기 위한 하나의 작업
  - 모든 그래프는 다 2진 트리로 되어있다.

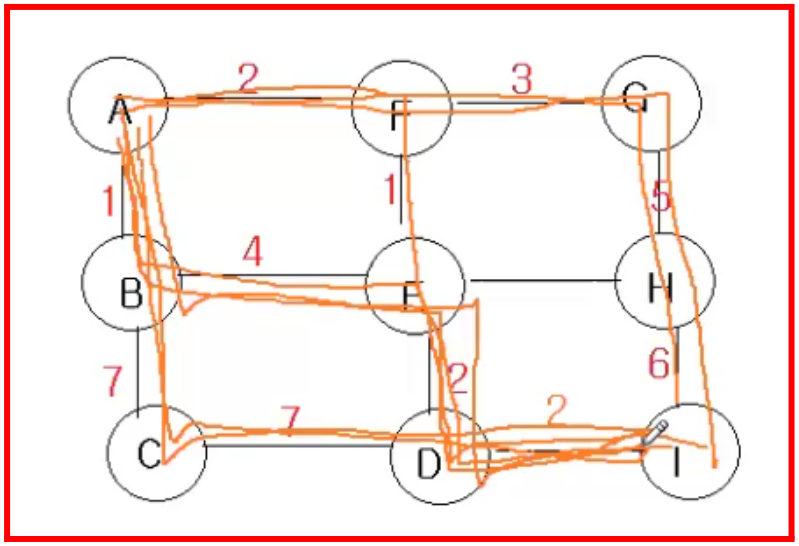


간선

- 노드와 노드를 연결하는 것
- 두가지 종류
  1. 무방향 (왕복가능)
     a. 화살표 X
  2. 방향 (한방향)
     a. 화살표로 표시

가중치

- 가장 효율적인 Path를 찾는데 사용된다:
  - 시험문제는 100% 이게 나온다.



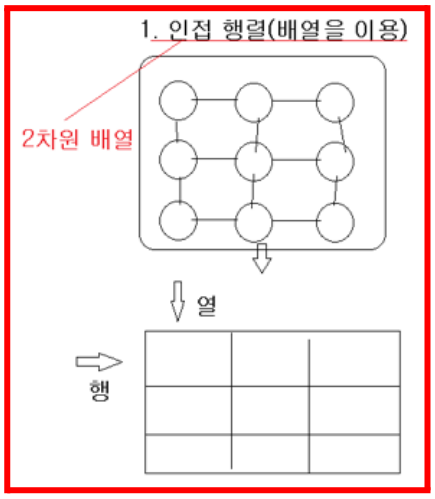
그래프 (Graph)

- **그래프는 순환이 가능하다.**
  - 순환(O) 이 되야하는 문제
  - 순환(X) 이 되지 않아야하는 문제
    - 문제 푸는 패턴이 다르다.

- 노드에서 나가는 방향의 개수에따라 **"진출 차수"**라 부른다.
- 노드로 들어오는 개수에 따라 **"진입 차수"** 라고 한다.

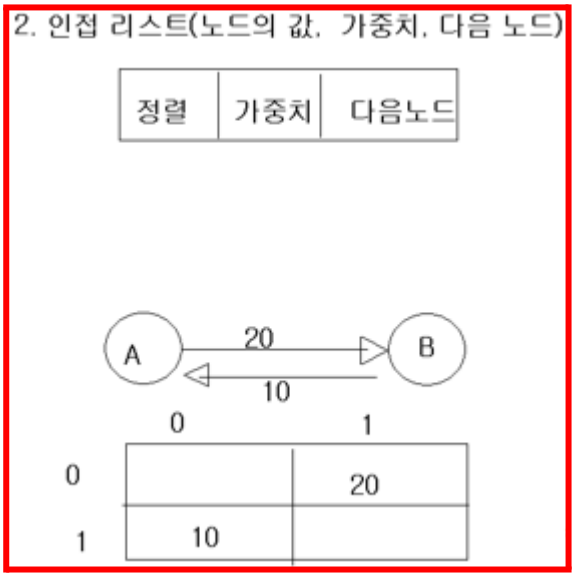
인접 노드

- 노드와 가까운 노드를 **"인접노드"** 라고 이야기 한다.
- **인접 노드에는 두 가지로 표현한다**
  1. **인접 행렬 (Adjacency matrix)**
     a. 배열을 이용 (스택에서 사용했던)



     b. 2차원 배열 (행과 열로 구성되어 있다)

  2. **인접 리스트 (Adjacency list)**
     a. 노드의 값, 가중치, 다음 노드
     b. 이 세가지를 묶어서 관리



2진 트리 구조 (Binary Tree Structure)

- 모든 그래프는 다 이진트리로 되어있다.