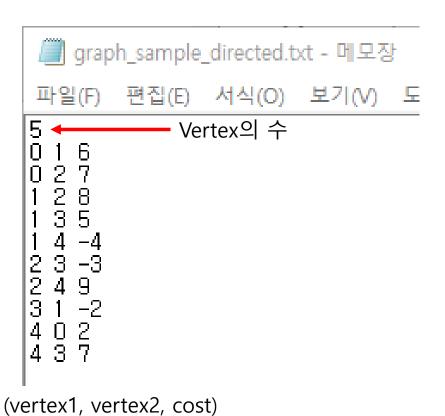
# Algorithm

Practice 08

#### Homework(Programming)

- 입력은 파일입출력 사용
- C file 제출
  - 함수 형식으로 구현하여 main함수에서는 해당 함수 call만 하는 형식
  - 주석 철저(함수 파라미터 및 변수 설명, 코드 설명 등등)
- 보고서
  - 컴파일 방법
  - 알고리즘에 대한 설명
  - 결과에 대한 사진 및 설명
- 제출 기한 : 11월 22일 화요일 수업시간 이전까지 이러닝으로 제출 (딜레이는 24일 23시 59분까지만 받습니다.)
- TA e-mail: <u>lunch5537@gmail.com</u>(컴퓨터비전 연구실 5527 석사과정 김선민)

### Homework #8-1 Bellman-Ford Algorithm



- 첫번째 줄 : 그래프 내의 총 Vertex의 수
- 두번째 줄부터 각각 (vertex1, vertex2, cost)를 의미: directed graph로 구성

## Homework #8-1 Bellman-Ford Algorithm

```
BELLMAN-FORD (G, w, s)

1 INITIALIZE-SINGLE-SOURCE (G, s)

2 for i = 1 to |G.V| - 1

3 for each edge (u, v) \in G.E

4 RELAX (u, v, w)

5 for each edge (u, v) \in G.E

6 if v.d > u.d + w(u, v)

7 return FALSE

8 return TRUE
```

#### Homework #8-1 Bellman-Ford Algorithm

- Input : graph\_sample\_bellman.txt (source : Vertex 0)
- Output : 그래프 내 모든 Vertex의 Shortest Path 및 Cost
- graph의 Shortest Path들을 간단히 손으로 그려보고, 프로그램 실행 결과의 path와 cost를 비교(캡쳐사진 보고서에 첨부)

#### Homework #8-2 Shortest paths in Directed Acyclic Graphs

```
DAG-SHORTEST-PATHS (G, w, s)

1 topologically sort the vertices of G

2 INITIALIZE-SINGLE-SOURCE (G, s)

3 for each vertex u, taken in topologically sorted order

4 for each vertex v \in G.Adj[u]

5 RELAX (u, v, w)
```

#### Homework #8-2 Shortest paths in Directed Acyclic Graphs

- Input : graph\_sample\_DAG.txt (source : Vertex 1)
- Output : 그래프 내 모든 Vertex의 Shortest Path 및 Cost
- graph의 Shortest Path들을 간단히 손으로 그려보고, 프로그램 실행 결과의 path와 cost를 비교(캡쳐사진 보고서에 첨부)

#### Homework #8-3 Dijkstra's algorithm

```
DIJKSTRA(G, w, s)

1 INITIALIZE-SINGLE-SOURCE(G, s)

2 S = \emptyset

3 Q = G.V

4 while Q \neq \emptyset

5 u = \text{EXTRACT-MIN}(Q)

6 S = S \cup \{u\}

7 for each vertex v \in G.Adj[u]

8 RELAX(u, v, w)
```

#### Homework #8-3 Dijkstra's algorithm

- Input : graph\_sample\_dijkstra.txt (source : Vertex 0)
- Output : 그래프 내 모든 Vertex의 Shortest Path 및 Cost
- graph의 Shortest Path들을 간단히 손으로 그려보고, 프로그램 실행 결과의 path와 cost를 비교(캡쳐사진 보고서에 첨부)