Homework # 10

데이타구조론 (CSE2003-02)

HW#10 과제 안내

- 일정
  - 게시 : 5/27(목) 13:00
  - 제출 마감 : 6/10(목) 11:00 (delay 없음)
  - 채점 결과 확인 : 6/14(월)
  - 이의신청 마감 : 6/18(금) (이의신청 이메일 : greenlife124@yonsei.ac.kr, [데이타구조론HW#10 이의신청])

HW#10 과제 안내

- 설명
  - 모든 코드의 핵심 부분에는 comment를 달아 설명 할 것 (not option!!)
  - Compiler는 visual studio 2019 이상을 사용하여, HW#10_학번_이름 하나의 파일로 압축하여 제출 할 것
- HW#10_학번_이름
  - HW#10_1 > **ArrayGraph.c,** ArrayGraph.h, ArrayStack.c, ArrayStack.h, GraphMain.c

HW#10_1 최단 거리 알고리즘 구현

- 아래와 같이 실행되도록 **ArrayGraph.c** 작성
  - **djkstra(), bellmanFord(), floyd(), printShortestPath()** 작성
- ArrayGraph.h, ArrayStack.c, ArrayStack.h GraphMain.c제공
- 주의 사항
  - HW#10_1 폴더 안에, 주어진 5개의 파일 모두 존재 해야 함

HW#10_1 출력 화면

```
G1 가중치 행렬
 0      10       5      INF     INF
INF      0       2       1      INF
INF      3       0       9       2
INF     INF     INF      0       4
 7      INF     INF      6       0

G1: 시작 정점 0, Dijkstra Algorithm 결과
A -> B 최단 경로: A     C       B
A -> B 최단 경로 비용: 8
A -> C 최단 경로: A     C
A -> C 최단 경로 비용: 5
A -> D 최단 경로: A     C       B       D
A -> D 최단 경로 비용: 9
A -> E 최단 경로: A     C       E
A -> E 최단 경로 비용: 7

G2 가중치 행렬
 0       6      INF      7      INF
INF      0       5       8      -4
INF     -2       0      INF     INF
INF     INF     -3       0       9
INF     INF      7      INF      0

G2: 시작 정점 0, Bellman-Ford Algorithm 결과
A -> B 최단 경로: A     D       C       B
A -> B 최단 경로 비용: 2
A -> C 최단 경로: A     D       C
A -> C 최단 경로 비용: 4
A -> D 최단 경로: A     D
A -> D 최단 경로 비용: 7
A -> E 최단 경로: A     D       C       B       E
A -> E 최단 경로 비용: -2

G1: 모든 정점 쌍의 최단 경로 경비(Flyod-Warshall Algorithm 결과)
 0       8       5       9       7
11       0       2       1       4
 9       3       0       4       2
11      19      16       0       4
 7      15      12       6       0
```

HW#10_1 참고 자료

Algorithm	void dijkstra(Graph* G, int v)

```
dijkstra(G, v)
  S ← {v}
  for i←0 to |V(G)|-1 do
    dist[i] ← cost[v][i]
    if cost[v][i] ≠ ∞ then
      pred[i] ← v
    else
      pred[i] ← NULL
  for(i←0; S!= V(G); i←i+1) do
    u ← S에 속하지 않은 정점 중에서 dist[]가 최소인 정점
    S ← S U {u}
    for(w←0; w∈V(G); w←w+1) do
      if(w∉S and dist[u]+cost[u][w]<dist[w]) then
        dist[w] ← dist[u]+cost[u][w]
        pred[w] ← u
end dijkstra()
```

Algorithm	void bellmanFord(Graph* G, int v)

```
bellmanFord(G, v)
  n ← G.n
  for i←0 to |V(G)|-1 do
    for each edge(u, v) ∈ E(G) do
      RELAX(u, v)
  for each edge(u, v) ∈ E(G) do
    if(dist[v] > dist[u]+cost[u][v]) then
        return FALSE
  return TRUE
end bellmanFord()
```

```
RELAX(u, v)
  if dist[v] > dist[u] + cost[u][v]
    dist[v] = dist[u] + cost[u][v]
```

Algorithm	void floyd(Graph* G)

```
floyd(G)
  n ← G.n
  for i←0 to |V(G)|-1 do
    for j←0 to |V(G|-1 do
      A[i][j] = cost[i][j]
  for k←0 to |V(G)|-1 do
    for i←0 to |V(G)|-1 do
      for j←0 to |V(G)|-1 do
        if (A[i][k] + A[k][j] < A[i][j]) then
          A[i][j] = A[i][k] + A[k][j]
end floyd()
```