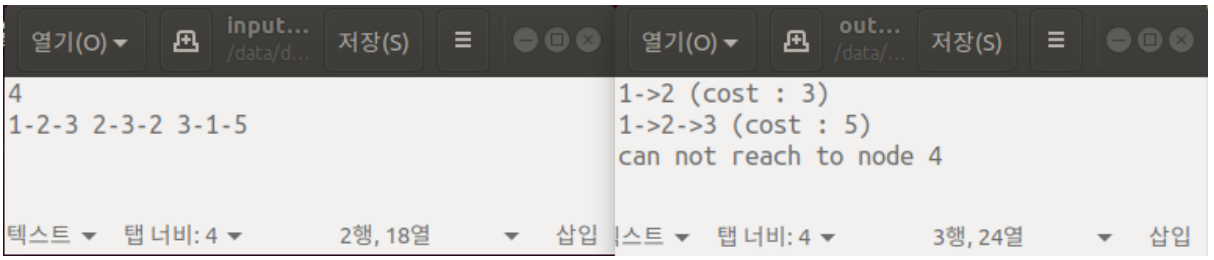
2023 Data Structure (CSE2010 12282)

Lab 12 (6/1) Dijkstra Algorithm

- arranged by TA Sehyun Cha

Due : ~2023.06.07.(Wed) 23:59, Late Submission : ~2023.06.08.(Thu) 23:59

- **Implement the Dijkstra Algorithm to find the Shortest Path.**
- 아래의 사진과 같이 **[input].txt** 파일을 입력받아 **Dijkstra Algorithm**을 수행하고 결과값을 **[output].txt** 파일에 출력하여 저장한다**.**
- **Implement Dijkstra Algorithm that finds the shortest path by receiving the [input].txt file as input commands as below, and the output result is stored in the [output].txt file.**



- **<input> :**
  첫째줄: Graph에서 Node의 개수.
  둘째줄: Edge Set 정보. (공백을 사이에 두고 **SourceNode-TargetNode-Weight** 의 형식, ex: **“2-3-2”** : **(Node:2)**-------~~[weight:2]~~------->**(Node:3)** 의미)
  1st line: the number of Nodes in the graph.
  2nd line: information of the edge set. (**SourceNode-TargetNode-Weight** format with “one space”, ex: **“2-3-2”** : **(Node:2)**-------~~[weight:2]~~------->**(Node:3)**)

- **<output> :**
  Source Node(key : 1)에서부터 나머지 모든 **Node**에 대한 **Shortest Path** 와 **Distance (weight**합**)** 출력**.**
  Shortest Distance가 같은 **Path**가 여러개 존재하는 경우 하나의 **Path**만 출력**.**
  All the shortest paths from the Source Node(key : 1) to other Nodes and the Distance (sum of weights) for each path should be printed.
  If there exist more than one Shortest Paths which have the same distance, print only one path.

<createGraph>

- 그래프 메모리를 할당하고 초기화합니다.
- **size**: 노드의 개수
- **vertices**: 엣지의 weight. weight가 없다면 0
- **nodes**: 현재 꼭지점(vertex), distance, 이전 노드 꼭지점을 저장
- Allocate memory and initialize it.
- **size**: the number of nodes
- **vertices**: weight of edges. 0 if there is no edge between two nodes
- **nodes**: current vertex, it’s distance, and previous vertex

<deleteGraph>

- 그래프의 메모리를 해제합니다.

- Deallocate Graph's memory

<dijkstra>
- 입력받은 그래프에 Dijkstra 알고리즘을 적용합니다.
- Heap에 Node를 삽입할 때, 업데이트될 거리가 현재 거리보다 작은 경우에만 수행되어야 합니다**(**거리가 같은 경우**,** 삽입 **x)**.
- 업데이트는 vertex 번호가 작은 순서부터 진행하면 됩니다.
- Apply Dijkstra Algorithm to a given Graph.
- When inserting a Node into the Heap, it should only be performed if the distance to be updated is smaller than the current distance(no insertion occurs if the distances are equal).
- The updates should be carried out in the order of ascending vertex numbers.

<shortestPath>
- 입력받은 지점까지 최단경로를 반환합니다.
- 배열의 크기는 노드의 개수이고, 경로의 반대 순서로 저장하면 됩니다.
- 나머지 부분은 0으로 채워주면 됩니다.
- **ex) size: 5 & path: 1->2->3 => [3, 2, 1, 0, 0]**
- Return the shortest path from source to the given vertex.
- The size of the array is the same as the number of nodes, and its order is opposite to the order of the path.
- rest of the array should be filled to 0.

<createMinHeap>
- min Heap을 생성합니다.
- Create a minimum Heap.

<deleteMinHeap>
- min Heap의 메모리를 해제합니다.
- Deallocate Heap's memory.

<InsertToMinHeap>
- 입력받은 vertex, distance인 Node를 min Heap에 삽입합니다.
- Insert Node with given vertex and distance.

<deleteMin>
- min Heap에서 distance가 가장 작은 노드를 제거하고 반환합니다.
- Pop the Node with minimum distance.

<Structure & Function Format>

Structure

```
typedef struct Node {
    int vertex;
    int dist;
    int prev;
}Node;
```

```
typedef struct Graph {
    int size;
    int** vertices;
    Node* nodes;
}Graph;
```

```
typedef struct Heap {
    int Capacity;
    int Size;
    Node* Element;
}Heap;
```

Function

```
Graph* createGraph(int size);
void deleteGraph(Graph* g);
void dijkstra(Graph* g);
int* shortestPath(Graph* g, int dest);
Heap* createMinHeap(int heapSize);
void deleteMinHeap(Heap* minHeap);
void insertToMinHeap(Heap* minHeap,
                     int vertex, int distance);
Node deleteMin(Heap* minHeap);
```

- 위 사진과 같은 Struct 구조체를 사용하셔야 합니다.
- 위 사진과 같은 함수들을 형식에 맞게 구현해주시면 됩니다.
- Struct format above should be used for implementation
- Functions should be implemented in appropriate format as above.
-

<Execution>

- gcc 20XXXXXXXX.c -o 20XXXXXXXX
- ./20XXXXXXXX [input_file_name] [output_file_name]
- !!! 꼭 제공되는 testcase로 실행시켜보시기 바랍니다.!!!!!,
- !!! Run your solution code with the provided test case above and check whether it works properly !!!
-

<Issue>

- 코드 작성시 주석을 적어주시기 바랍니다. 주석이 없는경우 Cheating으로 간주될 수 있습니다.
- 제공된 testcase는 채점 case에 포함됩니다. 모두 알맞게 나오는지 확인해보시기 바랍니다.
- 파일 입출력은 *argv[]* 를 사용하여 구현해주시기 바랍니다.
- 제출 마감 시간 이전의 가장 최신 버전의 commit을 기준으로 채점할 예정입니다.
- 제출 파일과, 폴더 naming 은 꼭 지정된 형식으로 해주셔야 합니다.
- Please write down the detailed comments when writing the code. If there is no comment, it might be considered cheating.
- Provided test case is included in the test case for grading. Please check to see if it makes a proper result.
- Do not use a fixed file name when inputting and outputting files, but implement it using argv[] as in skeleton code.
- Scoring will be based on the latest version of commit before the deadline.
- The names of the .c file and directory should be named in proper format.

- 출력시 꼭 정해진 형식에 맞게 출력해주셔야 합니다. 모든 공백은 띄어쓰기 한칸입니다.
- **Graph**에서 **Source Node** 는 **Key** 값이 **1**인 **Node**입니다. **Node**의 개수가 **5**개라면, 각 **Node**의 **Key** 값은 **1, 2, 3, 4, 5** 입니다.

- 결과 출력은 **Source Node(key=1)** 를 제외한 모든 **node**에 관해서 **Path**와 **Distance**를 모두 출력하시면 됩니다**. ex) "source->...PATH...->target (cost : cost값)"**
- 만약 **Source Node**에서 갈 수 있는 **Path**가 없을 경우 위 사진처럼 **"can not reach to node key값"**를 출력하시면 됩니다**.**
- **Distance**가 같은 **Shortest Path**가 여러개 존재하는 경우**,** 아무 경로나 하나만 출력하시면 됩니다**.**
- All the messages must be printed out according to the appropriate format as shown in the example above. **All the spaces are "one space".**
- **The Source Node of the graph is the node which has Key value "1". If the number of nodes in the graph is 5, each Node's Key value is 1, 2, 3, 4, 5.**
- **For printing the Path result, paths and its distance to all nodes except the Source Node(key=1) should be printed. ex) "source->...PATH...->target (cost : costValue)"**
- **If the path does not exist, "can not reach to node keyValue" should be printed.**
- **If there are multiple Shortest Paths, any of them can be printed.**

<Directory Format>

- 아래와 같이 git 프로젝트 폴더에 "lab12" 폴더 생성후, "lab12" 폴더 안에 "20XXXXXXXX.c" 파일을 위치시키시면 됩니다.
- After creating the "lab12" directory in the git-project-directory as below, place the "20XXXXXXXXXX.c" file in the "lab12" directory.

>>

2023_CSE2010_20XXXXXXXX/ (GitLab project directory)

---...

---lab11/

-------2022XXXXXX.c

---lab12/

-------2022XXXXXX.c

>> QnA : 2023ds12282@gmail.com