Lab 12

Data Structure

Lab12 (due on the day before the next Lab Session)

1. Do p12.c

Evaluation criteria

Category	Evaluation	
p12	100	
Total	100	

- *Use GCC 4.8 version or GCC 5.4 version.*
- *No score will be given if the gcc version is different.*

Lab12 – Dijkstra's algorithm

- Dijkstra's algorithm for finding the shortest path. Use a priority queue (heap) to find the node with the smallest distance from the source node
- For p12 you can submit on **portal site (assignment)**
- Folder name : Lab12
- code name: p12
- No score, if the folder, code names are wrong.
- No score, if it does not use FILE I/O
- Each code will be tested by 5 different input files.
- 20 score for each input, if you don't get the answer you get 0 score.

Lab12 – Dijkstra's algorithm

Graph createGraph(int size) Create a graph with nodes.

void printShortestPath(Graph g) Print the shortest path for the given path

Heap* createMinHeap(int heapSize) Create min heap

Void insertToMinHeap(Heap* minheap, int vertex, int distance) insert a new vertex to heap

Node deleteMin(Heap* minHeap) delete the smallest distance node for calculation

Lab12 – Dijkstra's algorithm – file input

- **Line1:** number of vertices
- **Line2:** edge information
- The result is the shortest path (and cost) from vertex 1 to every other vertices

Lab12 – Dijkstra's algorithm

- Structure

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

```
Struct Node{
    int vertex;
    int dist; //distance
    int prev;
}
```

```
Struct Heap {
    int Capacity;
    int Size;
    Node* Element;
} Heap;
```

Lab12. Dijkstra’s Algorithm

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct Node{
		int vertex;
		int dist; //distance
		int prev;
}Node;
typedef struct Graph{
		int size;
		int** vertices;
		Node* nodes;
}Graph;
typedef struct Heap{
		int Capacity;
		int Size;
		Node* Element;
}Heap;

Graph CreateGraph(int size);
void printShortestPath(Graph g);
```

```
Heap* createMinHeap(int heapSize);
void insertToMinHeap(Heap* minHeap, int vertex, int distance);
Node deleteMin(Heap* minHeap);

void main(int argc, char* argv[])
{
		FILE *fi = fopen(argv[1], "r");
		Graph g;

		int size;
		fscanf(fi, "%d₩n",&size);
		g = CreateGraph(size+1);
		char temp = 0;
		while( temp != '₩n' )
		{
				int node1, node2, weight;
				fscanf(fi,"%d-%d-%d",&node1,&node2,&weight);
				g.vertices[node1][node2] = weight;
				temp = fgetc(fi);
		}
		printShortestPath(g);
}
```

Lab12. Dijkstra's Algorithm

- program name : p12.c
- input : A file containing information about the number of vertex, edges connecting vertices, edge weights
- output : the all vertices' shortest path and cost.

Lab12. Dijkstra's Algorithm

- input file : Lab12_input.txt

```
6
1-2-7 1-3-9 1-6-14 2-4-15 2-3-10 3-6-2 3-4-11 6-5-9 4-5-6
```

- Result

```
1->2 (cost : 7)
1->3 (cost : 9)
1->3->4 (cost : 20)
1->3->6->5 (cost : 20)
1->3->6 (cost : 11)
```