Data Structures & Algorithms

Lab Exercise 4 – Graph Project

Due: to be demonstrated by Thursday 27th November

Learning outcomes
At the end of this lab you should be able to:

- Implement the Uniform Cost Search algorithm as a member function of class Graph.

Q1.

(i) Construct a graph with the same topology as illustrated on the fantasy map of Dor (you will need to create two text files to describe the vertices and their connections). Some sample entries might look like:

(From	**to**	**weight)**
Baldor	Caldor	40
Caldor	Baldor	40
Baldor	Aldor	60
Aldor	Baldor	60
Etc…		

Note that we will need to store two pieces of information at each vertex.

1) The name of the vertex (a string)
2) The cost of getting to that vertex from some arbitrary vertex (an integer)

Thus, we might use an STL pair for this purpose. To construct the Graph in function main():

```
…
#include <utility>  // for STL pair
using std::pair;

int main() {

       // Meaning of template arguments below:
       // pair<string, int> is the data we are storing at each node
       // int is the arc type (the data stored at each edge or arc)

       Graph< pair<string, int>, int > myGraph(6);
```

(ii) Implement the Uniform-Cost Search (UCS) algorithm described in Chapter 4: Lecture 20 (slides are available on blackboard). This should be added as a new member function to the **Graph** class as follows:

```
void ucs( Node* pStart, Node* pDest, void (*pVisitFunc)(Node*),
std ::vector<Node *>& path );
```

where:

`pStart` is a Graph node indicating the origin of the search
`pDest` is the goal vertex indicating the destination of the search
`void (*pVisitFunc)(Node*)` is a function that outputs the node currently being expanded (i.e. the node at the top of the priority queue).
`path` is a container (a vector) which holds the best path generated by the algorithm's completion

Test the UCS algorithm on your newly constructed graph.