CS302

Assignment 7: Finding Shade

Description

This is for all the brave students that survived the brutal summer at UNLV. Of course the only buildings that are open in the summer are: Thomas Beam Engineering Complex (TBE) and the Student Union (SU). You want to reach the student union from the engineering complex. To make matters worse, there's construction happening on campus as well which limits the paths you can take from TBE to the SU. Fortunately there are buildings you can sit behind to get some shade to avoid passing out. You are given a stamina amount and each edge takes you from one building to another, the longer edge uses up more stamina. You must find a path (if exists) that leads you from TBE to SU such that when you arrive at the SU you have at least one unit of stamina left that gives you energy to buy a passion fruit ice tea at Starbucks.

This will be an example of a graph problem using a DFS-like traversal to explore the graph. Each node is a building and each edge denotes a path between two buildings, each edge is weighted, the weight denotes a unit of distance. Each unit of distance uses up a unit of stamina. Thus, you need to update the stamina when arriving at a node. And there can be more than 1 path that leads to a node (you may need to re-explore a node if a better path is discovered). The recommended data structures needed would be

- `unordered_map< string, list<vertexType> > adjList` - adjacency list, a building name is mapped to a linked list of all its neighbors and the distances (`vertexType` would be a struct type you would define that stores the destination node and length of the neighbor edge)
- `unordered_map<string, string> predecessor` - a predecessors map that helps construct the paths

You would also need a "visited"-like map that helps you determine whether you should visit a node or not. You'll need to figure this out. You would probably need to write a recursive function (that traverses the graph in a DFS-like manner), that returns a `bool` type (`true` implying a path was found and `false` implies a path was not found).

Input

The first line of the input file contains the start node, end node, and the initial amount of stamina. The remaining lines contain the edge data. Each edge contains a from vertex, to vertex, and distance of the edge. You would need to insert each edge individually into the adjacency list using the following

```
adjList[from].push_back( {to, distance} );
```

Output

For each input, if a path exists you output `"Time to cool off!"` and then output `"Path: "` followed by the path that leads from the start node to the end node. If no valid path exists, then output
`"This is going to be a long summer..."`

Specifications

- You must use a recursive DFS-Like function to traverse the graph
- Use a block comment to document the recursive function and any functions you write in your main

Example Run

```
% g++ main.cpp
% ./a.out < input01.txt

Time to cool off!
Path: TBE CEB CBC SU

% ./a.out < input02.txt

Time to cool off!
Path: TBE WHI BHS SU

% ./a.out < input03.txt

Time to cool off!
Path: TBE SEB LFG WRI BEH SU

% ./a.out < input04.txt

This is going to be a long summer...

% ./a.out < input05.txt

This is going to be a long summer...

% ./a.out < input06.txt

This is going to be a long summer...
```

Submission

Submit the source file to code grade by the deadline

References

- Supplemental Video https://youtu.be/m6LX13SSx44
- Link to the top image can be found at https://clipart-library.com/clip-art/sunglasses-clipart-transparent-5.htm