PROJECT 1
DUE: 10/23/2017, 11:59 PM

DESCRIPTION

Write a program which implements the A* algorithm on an arbitrary map described through "connections" file and "locations" file. The algorithm should allow the exclusion of 1 or more cities in the locations file.

Two possible A* algorithm heuristics are required: the "Straight Line Distance" and the "Fewest Links". Note: The "fewest links" is the path in which the fewest number of connections are needed to get from start city to end city.

INPUT

The user input at runtime consists of the following:

- The connections file and the locations file
  - Allow the user to specify the full path of each file, for example: "c:\myDocs\myTestFiles\locations.txt"
- The start city
- The end city
- The city name(s) that should not be included in the solution path; let the user select 1 or more cities that are to be excluded.
- The heuristic to use
  - Straight line distance, or
  - Fewest links
- Whether to show just the optimal path found or a "step-by-step" path to the solution. The step-by-step path to the solution would allow the user to control each step by pressing the "enter" key

OUTPUT

For example if the start city is A1 and the end city is C3 and the user selects to show only the optimal path found , the output might look like ---

Optimal Path:
A1->B1->B2->C3

However if the user selects the "step-by-step" path to the solution, and the heuristic is the straight line distance, the output might look like --

Current Path: A1
Distance traveled: 0
Best move is to: B1

Current Path: A1-> B1
Distance traveled: 10
Best move is to: A2

Current Path: A1-> A2
Distance traveled: 9
Best move is to: D1

Current Path: A1->B1->D1
Distance traveled: 18
Best move is to: B2
.
.
.
Optimal Path:
A1->B1->B2->C3
Distance traveled: 38

SPECIFICATIONS

- City names are given as string labels that can be from 1 to 80 characters long.
- There will be no more than 200 cities.
- There will be no more than 10 direct roads from one city to the next city.
- There can be one-way roads. For example the connections file can show that A1 is connected to B2 but the connection from B2 does not include A1.
- All city locations are given in (x, y), where x and y are positive integers in the range from 0 to 800.
- (0, 0) is the top left of the coordinate space. X is the horizontal position. Y is the vertical position.

SUBMISSION

Through a link in Canvas. For those working in pairs, just upload one file.

DELIVERABLES

1- ZIP file including source file, executable file, and a readme text file with short instructions about how to compile and execute the program.
2- Project report (worth 10% of the project grade) in which you should describe:
   - Algorithms
   - Functions description and program design
   - Results
   - Performance
   - Task management: list the tasks accomplished by each group member by name.

GRADING CRITERIA

The TA will schedule demo sessions where the student(s) will demo their projects. Only the code that has been uploaded to canvas by the due date can be used.

- **Late submissions will lose points as stated on the syllabus – 10% for each day that is late.**
- **A program that does not compile will result in a zero!**
- A compilation warning will result in a 5% deduction
- A run-time error will result in a 5% deduction.
- Commenting and style is worth a total of 10%. The major purpose of programming style guidelines is to make programs easy to read and understand. Good programming style helps make it possible for a person knowledgeable in the application area to quickly read a program and understand how it works. Your program should begin with a comment that briefly summarizes what it does. This comment should also include your name and your partner's name. In most cases, a function should have a brief comment above its definition describing what it does. Other than that, comments should be written only when needed in order for a reader to understand what is happening.
- Report 10%
- Functionality 70%:
    - 10% - Program extracts data from "locations" and "connections" files correctly.
    - 30% - Program performs correctly while searching for the final path.
    - 30% - Program finds the correct final path.

PLAGIARISM !

Your program must be your original work, as stated and described in the syllabus. If you are unsure about whether some open source code can be used, contact the TA.