Assignment 3 Testcases

Submitted by:

Sakib Sadman

ID: B00934956

Submission Date: March 11, 2023

MapPlanner

depotLocation

Input validation

- Location's street is not on the map

Boundary case

- Set location of depot on the first street on the map
- Set location of depot on the last street on the map

Control flow

- none

Data flow

- call depotLocation without adding any streets on the map with addStreet.

addStreet

Input validation

- Start point and end point are the same
- Given streetID already exists in the map
- Another street with the same starting and ending point already exists in the map
- Another street's starting point matches with this street's ending point, and ending point matches with this street's starting point (starting and ending point reversed)
-

Boundary case

- Add street with either starting or ending point at the center (0,0)
- Add street with either starting or ending point at positive coordinates
- Add street with either starting or ending point at negative coordinates

Control flow

- Add street with no common intersections
- Add street with one common intersection
- Add street with two common intersections

Data flow

- Call addStreet after building a route with routeNoLeftTurn

routeNoLeftTurn

Input validation

- Destination is not on the map
- Depot location and destination location are the same

Boundary case

- Destination is on the other side of street (same street different side of street)
- Destination is one street away

Control flow

- Given destination is reachable with taking only right or straight turns
- Route leads to destination street, but on the opposide side of the street where that destination street is dead-end so u-turn is possible
- Route leads to destination street, but on the opposide side of the street where that destination street is not a dead-end so u-turn isn't possible
- When choosing leg for next turn, the only connected street requires a left turn
- When choosing leg for next turn, left turn street would result in lower distance than taking right or straight turn
- Destination is not connected with the map. In figure 1, if depot is A, and destination is G.

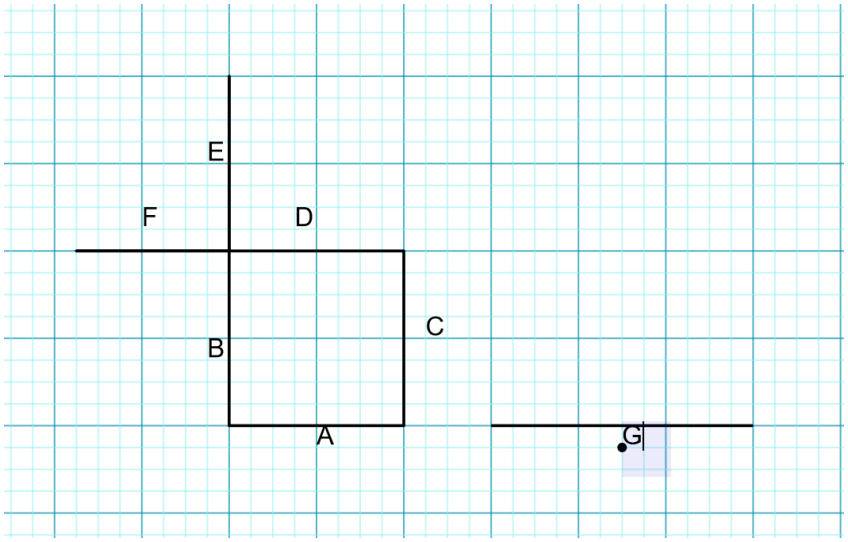

Figure 1 streets in map

Data flow

- Calling routeNoLeftTurn before setting a depot
- Calling routeNoLeftTurn before adding any streets

furthestStreet

Input validation

- none

Boundary case

- Call furthestStreet with only one street on the map
- Call furthestStreet with many connected streets on the map

Control flow

- Furthest street requires only right and straight turn to reach
- Furthest street requires combination of right, straight and left turn to reach
- On a map of many connected streets, a street directly connected to depot is of length 2147483647 or some absurdly high number, where the distance to it's centre would be higher than the cumulative distance of all legs of the second highest distant street.

Data flow

- Calling furthestStreet before setting a depot location
- Calling furthestStreet before adding any streets

Route

appendTurn

Input validation

- none

Boundary case

- none

Control flow

- none

Data flow

- none

turnOnto

Input validation

- legNumber is 0
- legNumber is greater than total legs on the route

Boundary case

- legNumber is 1
- legNumber is the last leg in the route

Control flow

- none

Data flow

- Call turnOnto before adding any legs with appendTurn.

turnDirection

Input validation

- legNumber is 0
- legNumber is greater than total legs on the route

Boundary case

- legNumber is 1
- legNumber is the last leg in the route

Control flow

- none

Data flow

- Call turnDirection before adding any legs with appendTurn.

legs

Input validation

- none

Boundary case

- Call legs with 0 legs in the route
- Call legs with only one leg in the route
- Call legs with two legs in the route

Control flow

- none

Data flow

- Call legs without appending any turns in the route with appendTurn

Length

Input validation

- none

Boundary case

- Call length with one leg in the route
- Call length with many legs in the route

Control flow

- Call length with two legs in the route
- Call length with zero legs in the route

Data flow

- Call length before appending any legs in the route

loops

Input validation

- none

Boundary case

- Call loops on a route with only one leg

- Call loops on a route with many legs

Control flow

- Call loops on a route containing no loops
- Call loops on a route containing one loops
- Call loops on route containing many loops
- Call loops on a route where inside of a loop, there exists one or many loops

Data flow

- Call loops before adding any legs with appendTurn

Simplify

Input validation

- none

Boundary case

- Call Simplify on a route with only one leg
- Call Simplify on a route with many legs

Control flow

- Call simplify on a route that contains only straight turns
- Call simplify on a route that contains no straight turns
- Call simplify on a route that contains a combination of straight, left, right and u-turns

Data flow

- Call loops before adding any legs with appendTurn

SubRoute

SubRoute

Input validation

- startLeg is greater than endLeg
- startLeg is less than 1

Boundary case

- Walk with one leg
- Walk with many legs

Control flow

- startLeg and endLeg are the same on a route with one leg
- startLeg and endLeg are same on a route with many legs
- startLeg is not in the route
- endLeg is not in the route

Data flow

- none

extractRoute

Input validation

- none

Boundary case

- route has one leg
- route has many leg

Control flow

- Call extractRoute on a subRoute where the startLeg isn't the first leg in the route
- Call extractRoute on a subRoute where the endLeg isn't the last leg in the route
- Call extractRoute on a subRoute where the startLeg is the first leg in the route and endLeg is the last leg in the route

Data flow

- none