# Test Description

**Test Name or ID**: BlackBoxUnit-calculateMinDistance

**Test Type**: Black box

**Description**: Ensure the “calculateMinDistance” function accurately calculates the minimum diversion distance from the route to the destination. This is important for efficient delivery planning.

**Setup:** Provide the following parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Member Variables | Description |
| const struct Truck\* | char routeSymbol; | the route symbol of the truck |
|  | int weight; | the current weight held by the truck |
|  | double volume; | the current volume held by the truck |
|  | double minDistance; | the shortest distance needed to travel to the destination |
|  | int isReady; | whether the truck is ready for delivery (neither full nor unreachable to destination) |
| const struct Point\* | char row; | the row number |
|  | char col; | the column number |

**Test Function**:

|  |  |
| --- | --- |
| Name | Description |
| calculateMinDistance | This function returns the minimum diversion distance from the route to deliver a package to a destination.  First, it should get the route the truck runs on from the parameter “truck”.  Then it should call the “getClosestPoint” function by inputting the “truck” and “dest” values to find the nearest point on the route to diver.  After that, it should call the “shortestPath” function to calculate the minimum distance to divert from the route to the destination.  Finally, return the diversion distance. |

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| valid truck route and destination | truck :   * &bluerRoute   point:   * {3,1} | 1 |  |  |
| empty truck route | truck :   * {{0}}   point:   * {3,1} | Error |  |  |
| null truck route | truck :   * NULL   point:  {3,1} | Error |  |  |
| null destination | truck :   * {{0}}   point:   * {3,1} | Error |  |  |
| destination on the route | truck :   * &bluerRoute   point:   * {4,1} | 0 |  |  |
| destination as starting point of the truck route | truck :   * &bluerRoute   point:   * {0,0} | 0 |  |  |
| destination as ending point of the truck route | truck :   * &bluerRoute   point:   * {17,24} | 0 |  |  |
| destination outside the map boundaries | truck :   * &bluerRoute   point:   * {2,25} | error |  |  |
| destination on the map boundaries | truck :   * &bluerRoute   point:   * {5,0} | 1 |  |  |
| destination on the map boundaries corner | truck :   * &bluerRoute   point:  - {0,24} | 15.52 |  |  |
| destination on a straight line path to route | truck :   * &bluerRoute   point:  - {4,15} | 6 |  |  |
| destination on a curved path to route | truck :   * &bluerRoute   point:  - {2,12} | 3.6 |  |  |
| multiple equidistant route points | truck :   * &bluerRoute   point:  - {6,7} | 2 |  |  |
| destination with special character | truck :   * &bluerRoute   point:  - {\*,’\n’} | error |  |  |
| destination with letter | truck :   * &bluerRoute   point:  - {6,G} | error |  |  |
| destination with empty character | truck :   * &bluerRoute   point:  - {6, } | error |  |  |

**Bugs Found**:

Description of each bug found above and how to reproduce it.