# Test Description

**Test Name or ID**: ValidatePackageBox

**Test Type**: White Box

**Description**: This test validates that the validatePackageBox function correctly identifies valid and invalid package sizes. The function should return 1 for valid sizes (1, 3, 5) and 0 for any other size.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: validatePackageBox

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Valid package size 1 | 1 | 1 | 1 | PASS |
| Valid package size 3 | 3 | 1 | 1 | PASS |
| Valid package size 5 | 5 | 1 | 1 | PASS |
| Invalid package size 2 | 2 | 0 | 0 | PASS |
| Invalid package size 4 | 4 | 0 | 0 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.

Test Description

**Test Name or ID**: ReturnInt

**Test Type**: White Box

**Description**: This test verifies that the returnInt function accurately converts alphabetic characters to their corresponding integer values based on their position in the alphabet.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: returnInt

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Convert 'A' to 0 | A | 0 | 0 | PASS |
| Convert 'B' to 1 | **B** | 1 | 1 | PASS |
| Invalid character 'Z' | Z | -1 | -1 | PASS |
| Convert 'Y' to 24 | Y | 24 | 24 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.

Test Description

**Test Name or ID**: Distance

**Test Type**: White Box

**Description**: This test checks the accuracy of the distance function, which calculates the Euclidean distance between two points.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: distance

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Distance between (0,0) and (3,4) | (0,0), (3,4) | 5.0 | 5.0 | PASS |
| Distance between (1,1) and (4,5) | (1,1), (4,5) | 5.0 | 5.0 | PASS |
| Distance between (0,0) and (0,0) | (0,0), (0,0) | 0.0 | 0.0 | PASS |
| Distance between (-1,-1) and (2,2) | (-1,-1), (2,2) | 4.2426 | 4.2426 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.

Test Description

**Test Name or ID**: PopulateMap

**Test Type**: White Box

**Description**: This test validates the populateMap function, which initializes a Map structure with predefined values for buildings and open spaces. The function should correctly set up the map with the specific number of rows, columns, and the layout of buildings.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: populateMap

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Correct number of rows and columns | N/A | |  | | --- | |  |  |  | | --- | | map.numRows equals MAP\_ROWS and map.numCols equals MAP\_COLS | | |  | | --- | |  |  |  | | --- | | map.numRows equals MAP\_ROWS and map.numCols equals MAP\_COLS | | PASS |
| First element in the map | N/A | map.squares[0][0] equals 0 | map.squares[0][0] equals 0 | PASS |
| Specific building placement (1,1) | N/A | map.squares[1][1] equals 1 | map.squares[1][1] equals 1 | PASS |
| Specific open space placement (2,0) | N/A | map.squares[2][0] equals 0 | map.squares[2][0] equals 0 | PASS |
| Last element in the map | N/A | map.squares[24][24] equals 0 | map.squares[24][24] equals 0 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.

Test Description

**Test Name or ID**: EqPt

**Test Type**: White Box

**Description**: This test verifies the eqPt function, which checks if two points are equal.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: eqPt

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Distance between (0,0) and (3,4) | (0,0), (3,4) | 5.0 | 5.0 | PASS |
| Distance between (1,1) and (4,5) | (1,1), (4,5) | 5.0 | 5.0 | PASS |
| Distance between (0,0) and (0,0) | (0,0), (0,0) | 0.0 | 0.0 | PASS |
| Distance between (-1,-1) and (2,2) | (-1,-1), (2,2) | 4.2426 | 4.2426 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.

Test Description

**Test Name or ID**: GetClosestPoint

**Test Type**: White Box

**Description**: This test ensures that the getClosestPoint function correctly identifies the closest point in a route to a given point.

**Setup:** 1. Ensure the development environment is set up with all necessary files (Truck.c, Truck.h, mapping.c, mapping.h).

2. Include the test file WhiteBoxTest.cpp in the Visual Studio project.

3. Configure the project for unit testing using CppUnitTestFramework.

**Test Function**: getClosestPoint

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| Closest point to (4,5) in route | Route: { (0,0), (3,4), (6,8) }, Point: (4,5) | 1 | 1 | PASS |
| Closest point to (7,9) in route | Route: { (0,0), (3,4), (6,8) }, Point: (7,9) | 2 | 2 | PASS |
| Closest point to (0,0) in route | Route: { (0,0), (3,4), (6,8) }, Point: (0,0) | 0 | 0 | PASS |
| Closest point to (3,3) in route | Route: { (0,0), (3,4), (6,8) }, Point: (3,3) | 1 | 1 | PASS |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Bugs Found**:

No bugs found in this function.