# Test Description

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

**Test Type**: Black box

**Description**: To ensure “printMap” function correctly displays the map with the appropriate symbols for buildings, routes, overlaps, and shortest paths. The purpose of this test is to verify that the printed map accurately represents the map's elements and helps users understand the map layout and routes.

**Setup:** Provide the following parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Member Variables | Description |
| const struct Map\* | int squares[MAP\_ROWS][MAP\_COLS]; | represents the squares that are buildings and the squares that are not |
| int numRows; | represents the number of rows of the map |
| int numCols; | represents the number of columns of the map |
| const int Base1 | NA | indicates whether the function should print row indices from 1 up (true) or 0 up (false) |
| const int alphaCols | NA | indicates whether the function should print col headers as letters (true) or as numbers (false) |

**Test Function**:

|  |  |
| --- | --- |
| Name | Description |
| printMap | This function prints the “map” variable inputted using the following symbols:  • space = open space  • X = building  • B = blue route  • G = green route  • Y = yellow route  • . = B & G routes overlap  • - = B & Y routes overlap  • \* = G & Y routes overlap  • + = B & G & Y routes overlap  • P = a shortest path or route from one point to another  The function should first set the maximum of rows + base1 to a local variable “rowMax”. Then, depending on the value of “alphaCols”, print the col headers as letters (if true) or as numbers (if false). Then, it should loop through all the rows and columns of the “map” variable and print the correct symbol for each point on the map. |

**Test Scenarios:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| empty map | map  - squares: [], numRows: 0, numCols: 0 | Error |  |  |
| map is NULL | map: NULL | Error |  |  |
| map containing buildings | &map with buildings | map printed with buildings |  |  |
| map containing blue routes | &map with blue routes | map printed with buildings + blue route |  |  |
| map containing green routes | &map with green routes | map printed with buildings + green route |  |  |
| map containing yellow routes | &map with yellow routes | map printed with buildings + yellow route |  |  |
| map containing overlapping building, all routes, overlap route and shortest path. | &map with overlapping routes and buildings | whole map printed |  |  |
| base1 set to true | Base1 =1 | Row indices are printed from 1 up |  |  |
| base1 set to false | Base1 =0 | Row indices are printed from 0 up |  |  |
| base1 set to MAX\_INT-1(true) | Base1 = MAX\_INT-1 | Row indices are printed from 1 up |  |  |
| base1 set to MAX\_INT(true) | Base1 = MAX\_INT | Row indices are printed from 1 up |  |  |
| base1 set to MAX\_INT+1(true) | Base1 = MAX\_INT+1 | Row indices are printed from 1 up |  |  |
| alphaCols set to true | alphaCols = 1 | Column headers are printed as letters |  |  |
| alphaCols set to false | alphaCols = 0 | Column headers are printed as numbers |  |  |
| alphaCols set to MAX\_INT-1(true) | alphaCols = MAX\_INT-1 | Column headers are printed as letters |  |  |
| alphaCols set to MAX\_INT(true) | alphaCols = MAX\_INT | Column headers are printed as letters |  |  |
| alphaCols set to MAX\_INT+1(true) | alphaCols = MAX\_INT+1 | Column headers are printed as letters |  |  |
| map numRows not matching the squares data | map: { squares: [ [0, 0], [0, 0] ], numRows: 3, numCols: 2 } | error |  |  |
| map numCols not matching the squares data | map: { squares: [ [0, 0], [0, 0] ], numRows: 3, numCols: 2 } | error |  |  |

**Bugs Found**:

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