*A brief description of notable obstacles you overcame.*

**First function**

* Took a long time to come up with strategy of *excluding* the situations (exclude strings with more than two digits together, strings with characters other than letters and digits etc.)

**Second function**

* Wasn’t that hard after doing the warm-up function
* Basically used the warm-up function four times
* Just had to make sure that off-grid and wall condition in if statements were correct for each direction
* Problems with “non-void function”, inserted “return -1” at the end to counter this, should not occur if all scenarios are covered in function

**Third function**

* Most difficult
* Had to think about how to process route and move robot while at the same time also keeping count of nsteps
* Wanted to make separate functions to check if route was valid (doesn’t go into wall or off the grid) and if route resulted in end position, but then didn’t want to write more code to count steps
* Had to make sure that I didn’t try to access characters out of the string, using inequalities (e.g. k+1 <= route.size()-1) and to make sure these conditions would come before trying to access the character
* Difficulty trying to convert part of the string into an integer, at first tried to convert individual characters to integers, and then using (10\*(first digit) + (second digit)), but then decided to use the atoi function instead to be faster, less complicated
* Many loops and if/else statements in return 3 loop, was very confusing and took a long time to figure out

*A description of the design of your program. You should use pseudocode in this description where it clarifies the presentation.*

isDirection(char direction)

* Return true if direction input is N/E/S/W/n/e/s/w
* Return false if not

bool isValidGridPosition(int r, int c)

* Return false if r or c is bigger than number of rows or columns in grid, or if integer input for r or c is negative, or if (r,c) is a wall
* Return true otherwise (i.e. it is on the grid, not a wall, not negative)

void executeSegment(int& sr, int& sc, char dir, int numSteps)

* moves robot as route is processed by third function
* sr or sc changes in accordance with direction and numSteps

bool isRouteWellFormed(string route)

* Return true for empty string
* Return false for routes that do not start with direction letter
* First for loop goes through each letter in route string to check if all are digits and valid direction letters (isDirection function), returns false if not digit and not letter
* Second for loop goes through digits in string and returns false if they are followed by two numbers (i.e. more than two digits are together in the route string), condition was k+2 < route.size()to make sure that loop doesn't try to retrieve character beyond route string

int navigateSegment(int r, int c, char dir, int maxSteps)

* return -1 if grid position is not valid (isValidGridPosition function), direction is not valid (isDirection function), or maxSteps is negative
* for loop for each direction
* condition of loop: next position is in the grid and not a wall
* if satisfy condition, add a step
* when reach wall or edge of grid, compare maxSteps input and steps
* if steps < maxSteps, wall or edge of grid blocking robot from moving maxSteps
* change value of maxSteps to steps (maxSteps = steps;)
* return new/unchanged value of maxSteps
* repeat routine for each direction, changing for loop condition for wall and edge of grid

int navigateRoute(int sr, int sc, int er, int ec, string route, int& nsteps)

* return 2 if starting position or ending position is not valid (isValidGridPosition function), or if route is not well formed
* nsteps = 0 to start counting steps moved

(empty string (assuming valid start and end grid positions) will have nsteps set to 0, but will not pass through loop below, so will return 1 with nsteps = 0)

* for loop to go through each character in route string, identify each segment, change (sr,sc) and nsteps accordingly (refer to code comments for more detailed information of code design)
* in the loop, return 3 if segment execution goes into wall or off the grid
* if finish loop without returning 3
  + return 0 if modified (sr,sc) = (er,ec)
  + else return 1 because it reached a position that is not (er,ec) but did not go into wall or off the grid

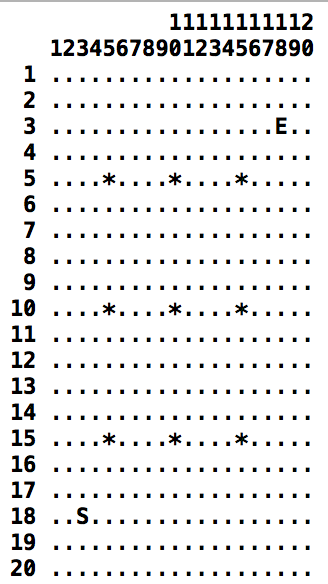
*A list of the test data that could be used to thoroughly test your program, along with the reason for and the expected result of each test.*

bool isRouteWellFormed(string route)

|  |  |  |
| --- | --- | --- |
| route | Reason | Expected result |
| "" | Empty string | T |
| "nNn01EN05n2E2e4enNn" | Double digits, single digits, upper and lower case letters, letters stringed together | T |
| "NnNnssnnNnnnNnn" | Only letters | T |
| "N14+e15" | Non-letters/numbers | F |
| "North14east15" | Not valid direction letters | F |
| "N014e15" | More than two digits together | F |

int navigateSegment(int r, int c, char dir, int maxSteps)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| r | c | dir | maxSteps | Reason | Expected result |
| 21 | 21 | N | 5 | Invalid grid position | -1 |
| 18 | 3 | x | 5 | Invalid direction | -1 |
| 18 | 3 | n | 5 | Random direction to valid grid position | 5 |
| 15 | 3 | e | 5 | Random direction with wall | 1 |
| 18 | 3 | s | 3 | Random direction to off-grid position | 2 |

int navigateRoute(int sr, int sc, int er, int ec, string route, int& nsteps)

setSize(20,20);

setWall(5,5);

setWall(10,5);

setWall(15,5);

setWall(5,10);

setWall(10,10);

setWall(15,10);

setWall(5,15);

setWall(10,15);

setWall(15,15);

len = -999;

|  |  |  |  |
| --- | --- | --- | --- |
| sr,sc | er,ec | route  Reason | Expected return value, len |
| 18,3 | 18,3 | ""  check empty string starting in end position | 0,0 |
| 18,3 | 3,18 | ""  check empty string starting in non-end position | 1,0 |
| 18,3 | 3,18 | "nNn01EN05n2E2e4eEn4W02w2S1s01e10nNn"  check random route string ending in end position; random route contains double digits, single digits, upper and lower case letters, letters stringed together; random route ends with letter | 0,42 |
| 18,3 | 3,18 | "nNn01EN05n2E2e4eEn4W02w2S1s01e10nN"  check random route string ending in non-end position; random route contains double digits, single digits, upper and lower case letters, letters stringed together; random route ends with letter | 1,41 |
| 18,3 | 3,18 | "NnNnssnnNnnnNnnNnnNeeeeEwweeeeeeeeEeee"  check random route string with only letters; ending in end position | 0,38 |
| 18,3 | 3,18 | "NnNnssnnNnnnNnnNnnNeeeeEwweeeeeeeeEee"  check random route string with only letters; ending in non-end position | 1,37 |
| 18,3 | 3,18 | "N15e15"  check random route string ending in end position; random route contains double digits, single digits, upper and lower case letters, letters stringed together; random route ends with number | 0,30 |
| 18,3 | 3,18 | "N14e15"  check random route string ending in non-end position; random route contains double digits, single digits, upper and lower case letters, letters stringed together; random route ends with number | 1,29 |
| 15,5 | 3,18 | ""  start position is wall | 2,-999 |
| 18,3 | 5,15 | ""  end position is wall | 2,-999 |
| 21,21 | 3,18 | ""  start position is off the grid | 2,-999 |
| 18,3 | 21,21 | ""  end position is off the grid | 2,-999 |
| 18,3 | 3,18 | "N14+e15"  route is not syntactically valid, contains non-letter/number | 2,-999 |
| 18,3 | 3,18 | "North14east15"  route is not syntactically valid, contains non-direction letter | 2,-999 |
| 18,3 | 3,18 | "N014e15"  route is not syntactically valid, contains more than two numbers in a row | 2,-999 |
| 18,3 | 3,18 | "N3E3N12E12"  route moves to cell containing a wall (but without wall would have reached end position) | 3,4 |
| 18,3 | 3,18 | "N3E3"  route moves to cell containing a wall (but without wall would not have reached end position) | 3,4 |
| 18,3 | 3,18 | "N18"  route moves to cell off the grid (but with extra row on top would not have reached end position) | 3,17 |
| 18,3 | 18,3 | "N3S3"  start and end positions are the same; route moves away from start/end position and back to start/end position (same as start position) | 0,6 |
| 18,3 | 18,3 | "N3S2"  start and end positions are the same; route moves away from start/end position and does not move back to start/end positionposition) | 1,5 |