Amy Yu

905138432

Project 3 Report

1. One obstacle that I had was figuring how to alter the columns and rows of the current position after each course segment. However, I realized that the number of columns change if the direction is east or west, and the number of rows change if the direction of north or south. I also had difficulty having the program add one step to the total number of steps if there were no digits after the direction character. I was able to fix this by using another if statement and incrementing by one.

bool isCourseWellFormed(string course)

set all letters to uppercase

if course is empty string

return true

if course only contains one valid letter

return true

if the first character is a valid letter

repeatedly:

check if there are 3 digits in a row

return false

check if there are 2 digits after letter

continue

check if there are no digits after letter

continue

return true if all tests are passed and false otherwise

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

change dir to uppercase

if dir is not valid or maxSteps is negative

return -1

else if dir is N

if at edge of grid

return 0

else

repeatedly:

if a wall is hit, return one less step

else continue through loop

if maxSteps is less than number or rows

return maxSteps

else

return steps required to reach end of grid

repeat for dir equals E, S, and W

else

return -1

int driveCourse(int sr, int sc, int er, int ec, string course, int& nsteps)

change every character in string to uppercase

if isCourseWellFormed is not true

return 2

if sr and sc are out of the grid or at a wall

return 2

if first character is letter

repeatedly:

if there are two digits in a row, set steps to that number

else if there is only one digit, set steps to that number

else if there are no digits after the letter, set steps to one

if the maximum steps is equal to the steps actually taken

add max steps to the total distance

change current row or column according to direction

set nsteps to total distance

continue

else if max steps is less than total steps taken

set nsteps to total distance plus max steps

return 3

if the current row and column is equal to the ending row and column

set nsteps equal to total distance

return 0

else

set nsteps equal to total distance

return 1

1. Test data

// tests if number of steps taken is 0, should be true because ran into wall

assert(driveSegment(3, 1, 'E', 3) == 0);

// tests of course is valid, should be true

assert(isCourseWellFormed("n2e1");

// tests if program correctly interprets no digits as one step, should be true

assert(isCourseWellFormed("neswese"));

// course is not valid because of x

assert(!isCourseWellFormed("e1x"));

// course is not valid because 3 digits in a row

assert(!isCourseWellFormed("e154"));

// tests if the number of steps taken is 2, should be true

assert(driveSegment(3, 1, 'N', 2) == 2);

int len;

len = -999; // so we can detect whether driveCourse sets len

// tests if the course is valid and ends at the ending position and if nsteps is set to 7, should be true

assert(driveCourse(3,1, 3,4, "N2eE01n0s2e1", len) == 0 && len == 7);

len = -999; // so we can detect whether driveCourse sets len

// tests if driveCourse returns 2 and leaves len unchanged, should be true because the course is not valid

assert(driveCourse(3,1, 3,4, "e1x", len) == 2 && len == -999);

len = -999; // so we can detect whether driveCourse sets len

// tests if driveCourse returns 3 and sets len to 1, should be true because only one step can be taken before there is a wall

assert(driveCourse(2,4, 1,1, "w3n1", len) == 3 && len == 1);

// should return 2 because there is a wall at the starting position

len = -999;

assert(driveCourse(3, 2, 3, 2, "n4", len) == 2 && len == -999);