***Project 3 - Due November 1, 2012***

***Obstacles***

There were few obstacles while writing the code. The first large obstacle was during the traverseSegment method. While trying to find the number of steps possible to be taken in a certain direction, it was difficult to figure out a way in which to check the next position and make sure it was not an invalid position of the maze, while accounting for the direction. I ended up writing a switch statement to account for each direction. The other problem I ran into was recalling how to transfer the numbers in the string into an integer. After rereading the characters and strings page a few times, I was able to write the code to change the string into an integer variable.

***Description - Psuedocode***

for isPathWellFormed

repeatedly:

if not a digit

return false

while it is a digit (up to two digits)

go to the next character in the string

if end of string

return false

based on current character

next character or return false

if it reaches end of string, return true

for traverseSegment

if r or c is out of bounds

return -1

if (r,c) is a wall

return -1

if maxSteps is negative

return -1

repeatedly:

based on direction

if next position is not a wall

move to the position

if wall or out of bounds

set proceed to false and end loop

for traversePath

if starting position or ending position is a wall

return 3

if the path is not syntactically correct

return 3

repeatedly:

find the next number in the string

change string containing number into integer

increment position of string

find possible steps in direction given

if possible steps does not equal steps requested

return 2

based on direction:

add or subtract number of steps from row or column

increment position of string

if path ends at (er,ec)

return 0

otherwise

return 1

***Test Data***

(Using given maze)

setSize(3,4);

setWall(1,4);

setWall(2,2);

if (!isWall(3,2))

setWall(3,2);

draw(3,1, 3,4);

*for isPathWellFormed*

correct segment ("2N1e01E0n2e1e") = true

two letters in a row ("4se") = false

ending on number ("3s2") = false

spaces ("3s 2n") = false

other characters ("2w+3n") = false

zero path segments ("") = true

three digit number ("144N") = false

invalid direction ("2N1f") = false;

*for traverseSegment*

north (3,1,'n',2) = 2

east (1,1,'e',2) = 2

south (2,4,'s',1) = 1

west (3,4,'w',1) = 1

limited steps (3,1,'n',1) = 1

too many steps (2,4,'w',3) = 1

invalid grid position (0,0,'n',1) = -1

wall at starting point (3,2,'s',1) = -1

invalid direction character (1,1,'q',1) = -1

maxSteps negative (1,1,'w',-3) = -1

*for traversePath*

correct ending (3,1,4,3,"2n2E2S1e", numsteps) = 0 ... numsteps = 7

incorrect ending (3,1,4,3,"2n2E2S", numsteps) = 1 ... numsteps = 6

invalid path (3,1,4,3,"32q", numsteps) = 3 ... numsteps = 0

could not finish path (3,1,4,3,"2n2E3w", numsteps) = 2 ... numsteps = 6

invalid starting grid position (0,0,4,3,"2n2E2S1e", numsteps) = 3 ... numsteps = 0

invalid ending grid position (3,1,1,4,"2n2E2S1e", numsteps) = 3 ... numsteps = 0