***Report***

1. One of the biggest obstacles I faced during the coding process was that the program didn’t work when I sent integers as parameters to the function navigateSegment using the string (route[i]). On outputting these sent values, I discovered that it was sending ASCII values of these integers and that’s why the program didn’t work. Another one of the initial problems I faced was that for navigateRoute, when I called navigateSegment to navigate in one direction, I forgot to keep updating sr and sc. I did that now, by creating my own new function.
2. function row

check direction (n,e,w,s)

find max steps in that direction

update sr or sc

check for if off the grid or wall on the next/previous position

update sr/sc again to that position

function isRouteWellFormed

check first letter is n,e,w,s

if not

return false

otherwise

*repeatedly*

check if there are no characters other than alphabets and digits

if not

return false

check if digit

increment cur

check if alphabet(n,e,w,s)

set max to cur if its less

set cur to 0

If all false

Return false

If max greater or equal 3

False

If max less than 3(only 2 simultaneous digits)

Return true

Function naviagteSegment

If wall at row,col entered or maxsteps negative or direction is not n,e,w,s

Return -1

If direction is north, east, west, south

*While you don’t reach the top/bottom/right end/ left end of the grid*

Increment a variable trial until no walls encountered

If wall

Break

If steps given by user less than maximum steps possible

Return steps by user

Otherwise return maximum steps possible

Function navigateRoute

If route syntactically wrong or a wall is encountered at start or end positions

Return 2

*Repeatedly*

If either n, e, w, s

If next place holds a digit

If the place next to the next place also holds a digit

Increment steps by calling navigateRoute and sending the first digit\*10+second digit as the value for maxsteps

Call row to update new row and column position

If wall at new row and column position or off the grid

Set nsteps to value of steps

Return 3

If the place next to the next place doesn’t hold a digit

Increment steps by calling navigateRoute and sending the digit as the value for maxsteps

Call row to update new row and column position

If wall at new row and column position or off the grid

Set nsteps to value of steps

Return 3

If next place holds another direction alphabet

Increment steps by calling navigateRoute and sending 1 as the value for maxsteps

Call row to update new row and column position

If wall at new row and column position or off the grid

Set nsteps to value of steps

Return 3

If updated rows and columns equal destination row and column

Return 0

Else

Return 1

Int main()

Varies…

c) For a grid of setSize(3,4) and certain walls (and ignoring starting and end point in this condition),

**1234**

**1 ...\***

**2 .\*..**

**3 S\*.E**

Input of route and start and end row and column positions

(1N2Ews, 2,1,3,4) –syntactically wrong route (starts with a digit)

(n1xe23w, 2,1,3,4) –syntactically wrong route (has x)

(n+2e01, 2,1,3,4) – syntactically wrong route (character +)

(n12e123w1s, 2, 1, 3 ,4) –syntactically wrong route (has more than 2 simultaneous digits)

(n12ews1, 2,3,3,4) – returns 2 because wall at start point(2,3)

(n3e1, 3,1,1,2)- returns 3 because runs off the grid and nsteps=2(in this case)

(n2ese01, 3,1,2,3) –returns 3 because wall at (2,2) and nsteps=3

(n2e2se01, 3,1,3,4)- returns 1 because doesn’t reach destination nsteps=6

(n2e2se01s,3,1,3,4)- returns 0 and nsteps=7

(e02,3,1,3,3)- returns 3 because wall at (3,2) and nsteps=0