//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//TITLE: Turtle Drawing Simulator

//

//This program is a simlation drawing of the user inserting a text file to give the turtle

//direction with commands that are defaulty made in the project. The turtle can either

//move with it pen up/down, left/right, and forward with #'s of steps. With the program

//being executed you are able to print the drawing with a single command.

//

//Programmer: Justin Tran

//Class:CSCI 1106

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

#include <iostream>

#include <fstream>

#include <iomanip>

#include <string>

using namespace std;

void arrayGrid();

void penUp(char &penStatus);

void penDown(char &penStatus);

void turnLeft(int &currentX, int &currentY, int &nextX, int &nextY, char &currentFace); //current x and y co-ordinates of the turtle which is set 0,0

void turnRight(int &currentX, int &currentY, int &nextX, int &nextY, char &currentFace); //as program read the file the turtle will move in the fourth

void forward (int num, int &currentX, int &currentY, int &nextX, int &nextY, char &penStatus, char &currentFace); //quadrant with x being postive and y beinf also postive increasing

void printArray();

void quit();

const int SIZE = 40; //size of the board

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GLOBAL VARIABLES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int grid[SIZE][SIZE]; //40 x 40 grid for the turtle to draw

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main()

{

ifstream infile;

infile.open("turtle.txt");

char commands; //character to read from the file

int n; //steps for function foward()

int cY = 0, cX=0, nX=0, nY=0; //c=current, n=next

char pen = 'U'; //default status of the turtle direction is east

char face = 'E'; //default status of the pen is up

while (!infile.eof())

{

infile >> commands;

switch (commands) //call appropriate functions based on the read character

{

case 'U':

penUp(pen);

break;

case 'D':

penDown(pen);

break;

case 'L':

turnLeft(cY,cX,nX,nY, face);

break;

case 'R':

turnRight(cY, cX, nX, nY, face);

break;

case 'F':

infile >> n;

forward(n, cY, cX, nX, nY, pen, face);

break;

case 'P':

printArray();

break;

case 'Q':

quit();

break;

default: cout << "Error: Invalid character in file" << endl;

}

}

infile.close();

return 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: arrayGrid

//pre: none

//post: setting the array values all set to 0

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void arrayGrid()

{

for (int i = 0; i<SIZE; i++)

for (int j = 0; j<SIZE; j++)

grid[i][j] = 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: penUp

//pre: none

//post: setting pen status to up

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void penUp(char &penStatus)

{

penStatus = 'U';

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: penDown

//pre: none

//post: setting pen status to down

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void penDown( char &penStatus)

{

penStatus = 'D';

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: turnLeft

//

//pre: switch in main() is reading all the direction that is left

//on the test file then goes to the function turnLeft() in the

//switches of E,W,N,S

//

//post: each direction in the switch which turn the current

//direction to the left side of the turtle then is set as next

// direction, etc.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void turnLeft(int &currentX, int &currentY, int &nextX, int &nextY, char &currentFace)

{

char newdir;

switch (currentFace)

{

case 'E': //current direction facing east

newdir = 'N'; //turn north

if (currentX>0)

nextY = currentY - 1; //move up

nextX = currentX;

break;

case 'W': //current direction facing west

newdir = 'S'; //turn south

if (currentX<SIZE - 1)

nextY = currentY - 1; //move down

nextX = currentX;

break;

case 'N': //current direction facing north

newdir = 'W'; //turn west

if (currentY>0)

nextX = currentX - 1; //move left

nextY = currentY;

break;

case 'S': //current direction facing south

newdir = 'E'; //turn east

if (currentY<SIZE - 1)

nextX = currentX + 1; //move right

nextY = currentY;

break;

}

currentFace = newdir; //face the new direction

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: turnRight

//

//pre: switch in main() is reading all the direction that is left

//on the test file then goes to the function turnRight() in the

//switches of E,W,N,S

//

//post: each direction in the switch which turn the current

//direction to the right side of the turtle then is set as next

// direction, etc.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void turnRight(int &currentX, int &currentY, int &nextX, int &nextY, char &currentFace)

{

char newdir;

switch (currentFace)

{

case 'E':

newdir = 'S';

if (currentX<SIZE - 1)

nextX = currentX + 1;

nextX = currentX;

break;

case 'W':

newdir = 'N';

if (currentX>0)

nextY = currentY - 1;

nextX = currentX;

break;

case 'N':

newdir = 'E';

if (currentY<SIZE - 1)

nextX = currentX + 1;

nextY = currentY;

break;

case 'S':

newdir = 'W';

if (currentY>0)

nextX = currentX - 1;

nextY = currentY;

break;

}

currentFace = newdir; //face the new direction

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: forward

//

//pre: gets the current direction of the turtle from global variable

//and also get the status of the pen aswell from global variable

//

//post: whenever it is called in a direction S,W,E,N the function

// will determin whether the pen is up, if the pen is up

// then it will move the turtle to how steps it reads on .txt

//if pen is down then it will draw '\*' in how many steps it is.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void forward(int num, int &currentX, int &currentY, int &nextX, int &nextY, char &penStatus, char &currentFace)

{

switch (currentFace) //changing direction per current facing direction

{

case 'E': //current direction facing east

while (currentY<SIZE && num>0) //moving forward means moving to the right

{

if (penStatus == 'D')

grid[currentX][currentY] = 1;

currentY = nextY++; //move one step to the right

num--; //decrement one step to the left

}

break;

case 'W': //current direction facing west

//moving forward means moving to the left

while (currentY>0 && num>0)

{

if (penStatus == 'D')

grid[currentX][currentY] = 1;

currentY = nextY--; //move one step to the left

num--; //decrement one step to the left

}

break;

case 'N': //current direction facing west

//moving forward means moving up

while (currentX>0 && num>0)

{

if (penStatus == 'D')

grid[currentX][currentY] = 1;

currentX = nextX--; //move one step up

num--; //decrement one step to the left

}

break;

case 'S': //current direction facing south

//moving forward means moving down

while (currentX<SIZE && num>0)

{

if (penStatus == 'D')

grid[currentX][currentY] = 1;

currentX = nextX++; //move one step down

num--; //decrement one step to the left

}

}

if (penStatus == 'D') //last direction

grid[currentX][currentY] = 1;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: printArray

//pre: print array with '\*' is the turtle on the grid is set with 1

//

//post: print array '\*'

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void printArray()

{

for (int i = 0; i<SIZE; i++)

{

for (int j = 0; j<SIZE; j++)

if (grid[i][j] == 1)

cout << '\*'; // printing the \* for the 1 recorded from the turtle

else

cout << ' ';

cout << endl;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//name: quit

//pre: switch in main() is reading 'Q' in .txt

//post: program quits with exit (1106)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

void quit()

{

cout << "Exiting the program..." << endl;

exit(1106);

}



