Design:

**Classes:**

Ant

Board

Space

**Functions:**

main()

menu()

runSimulation()

**Program Description:**

class Ant that describes the ant on the board

integer m\_antRow to store the row that the ant is in.

integer m\_antCol to store the column that the ant is in.

enum antOrientation

has 4 values: Left, Right, Up, Down that describe the direction the ant is facing.

void rotate(char direction) rotates the ant left or right depending on the passed direction.

bool move(int rowSize, int colSize) – moves ant one space in direction it is currently facing.

if the space the ant is moving to is white turn it right 90° and change the space to black. if it is black, turn it left 90° and change the space to white using toggle color and rotate. If the ant would be moving off of the board it should rotate right 90° and not move.

int getRow()

int getCol()

void setRow(int row)

void setCol(int col)

class Space that describes a single space on the board

enum SpaceColor – describes possible colors of a space.

has 2 values: White and Black

SpaceColor m\_colorOfSpace – the current color of the space.

character displayedSpace – the character that is printed for the space.

bool occupied: set to true if ant is in the space and false otherwise.

Space(bool occupied) constructor that initializes the spaceColor to white and the displayedSpace to ‘ ‘ and m\_occupied to the passed value.

updateDisplayedSpace(): changes the displayedSpace character to reflect the color and ant status of the square.

void toggleColor(): change space to the opposite color

getSpaceColor()

setSpaceColor()

getDisplayedSpace()

setDisplayedSpace()

class Board

constructor that creates the dynamic array of Space objects and places the ant.

int m\_numRows – number of rows on board

int m\_Cols – number of columns on board

2D array of Space objects representing the board: m\_board

Ant member object: m\_ant

antLeave(int row, int col)

changes the status of the space.m\_occupied at board[row][col] to false and calls

space.updateDisplayChar() function to change the character of the space

antEnter(int row, int col)

changes status of the space.m\_occupied variable at board[row][col] to true and calls space.updateDisplayChar() to change the character of the space.

void moveAnt()

int currentRow and int currentColumn will storethe ant’s current position

if ant.move() is successful

antLeave(current row and column of ant)

update new space to hold the ant with antEnter(new location of ant)

printBoard()

void runSimulation(int rows, int cols, int steps, int antRow, int antCol)

create board object with passed number of rows, and columns and place the ant in the passed row and column.

for the desired number of steps:

antLeave() – change space currently occupied by the ant to reflect its absence

moveAnt() – move the ant to the next space

antEnter() – change the new space to reflect occupation by the ant