Arjun Kahlon
kahlona@oregonstate.edu

Design:

Test Case	Input Values	Function	Expected	Observed
Input 3 in menu	Input = 3	menuLangton()	Do while loop woud renter	The do while loop correctly renetered as expected
Input 2	Input = 2	menuLangton()	Simulation loop would not start and game will output “Program will now exit.”	The program outputted “Program will now exit” and program successfully exited.

Inputting char For int variables	Input = ‘g’	main()
		Testing cin extraction by using:
		cin.good()
			Expected my if(cin.good()) to correctly check for char and return user back to beginning of loop.	Returned the user back into the loop but caused an endless loop. I could not find a solution to this so I left the user input type validation out of my final code.

Inputting large int values for rows and columns	Input for rows = 50 Input for columns = 50	main() Initialized Ant Object through Ant Constructor	Ant Object would be created and a board with the correct number of rows and columns would be created and initialized	The board was successfully created with the correct number of rows and columns.

Placed cout statement to ensure correct variable values	cout << x_coord;
	cout << y_coord;
		Board.cpp
		Print Board
			Would print the correct values for the x and y coordinates of the ant’s location	Printed out incorrect values at first but aided me in fixing the problem. This test eventually printed the correct values for x and y.

Placed cout to ensure correct color being stored	cout << getCurrentColor()	Ant.cpp	Would correctly print the correct color being stored	Correctly printed the correct char value stored for color

Program Design:

Create objects for Ant and Board.

Create member variables for both the Ant Object as well as the Board object.

Ant	Board
Private: Direction drct, Board*, int x_temp, int y_temp	Private char** board, int total_rows, int total_columns, int x_coord, int y_coord
Public: Ant(int, int, int, int). Void setDirection(Direction), Direction get Direction, void move(int, int), Board* getBoard()	Public: Board(int, int, int, int), void setXY(int, int), void printBoard(), int getXCoord(), int getYCoord(), char getCurrentColor(), void setCurrentColor(), void deleteBoard()

I created a Board object in the Ant class. This board object gets created and initialized when I declare the Ant Object in main.

The program starts by welcoming the user to the Langton Simulation. It then runs my menuLangton() function, which prompts the user to choose 1 to start simulation or choose 2 to exit. This function returns type bool. If menuLangton() returns true, then the Langton’ simulation begins. If menuLangton() returns false, then the program outputs “Program will now exit” and exits. When menuLangton() returns true, I prompt the user for rows, columns, starting x coordinate, starting y coordinate, and total_steps. My program runs a for-loop using the total_steps as the loop control variable. I use the move() function for the Ant class as well as the printBoard() function for the Board

Class within this for loop. The output is pushed to the screen and prints a board for every individual step until the loop runs to completion. The move() functions contains conditionals that determine the direction of the ant as well as the current color of the x,y coordinate.

Reflection:

Overall, I found this lab extremely difficult. It took me a while to brainstorm how exactly I would create different object and use member functions for them to interact. I really struggled with testing user input in the main. As I showed in my test case above, I tried different methods to test user input. This include testing successful cin extraction for each user input. I was succesful in accomplishing this various int types but was unnsuccesful when the user entered the wrong type (such as a char instead of an int.). The program would not clear the input buffer and would get stuck in an endless loop. Thus I expect to be marked off for not having a strong user input validation process. I will continue to research different methods for testing user input and will be prepared for my next lab/project. I also struggled with declaring pointers and spent hours changing the syntax to get everything to compile correctly. I had multiple segmentation faults during compilation, so I had to completely break down my code to ensure memory was correctly being created and deleted. The conditionals that determined ant movement, direction, and color of coordinate were also very troublesome for me and took hours of debugging. I wasn't sure how to provide a correct outline of my board. Overall, I felt this assignment really challenged me as a programmer and pushed me out of my comfort zone. While I know I didn't perfect this assignment, I do take pride in what I was able to accomplish, given my trepidation when first attempting this assignment. I know this is a challenging course. I'm ready to tackle the hurdles that this class presents.