CS 121 — **Programming Assignment #2**

Bruce Bolden — **20 Points**

February 2, 2023 — **Due:** February 10, 2023

Objective: Become familiar with dynamically allocated two dimensional arrays and the basics of software design. Recall from CS 120 that top-down design is a problem-solving method in which the programmer breaks a problem up into its major subproblems and then solves the subproblems using techniques such as stepwise-refinement to derive the solution to the original problem. A structure chart is a documentation tool that shows the relationships among the subproblems of a problem. All subproblems should translate *directly* into functions used by the final program.

Program Description: A robot, R2D2/a Roomba, has a malfunction that causes it to wander aimlessly within a large rectangular enclosure whose dimensions are m by n. Assuming that it is at position (i, j) within the enclosure, its next step (randomly chosen) is equally likely to bring it to any one of eight nearest-neighbor cells, $(i-1, j)$, $(i-1, j+1)$ etc., except that it may not go outside the boundary of the enclosure. Thus, the robot will eventually visit every cell.

Write a program which allows the robot to start at some location (row,col) in a 2D array of size n by m. The input is the values for m, n, row, and col (described above), and k, a value that limits the number of moves that can be made (in order to guard against excessive running time). The program should terminate when all the cells have been visited or the iteration limit k is reached, whichever comes first. Print the total number of steps taken and a grid showing the number of visits to each cell. Your program should also print out all the cells that were visited the maximum number of times and what the maximum value. For example, if the maximum number of times a cell was visited is 25, display all the cells that were visited 25 times.

Your program should handle a maximum grid size of 50×50. For testing purposes you should start with smaller values, e.g., $n = 4$, $m = 5$, and $k = 100$. When successful you should try more realistic sizes such as $n = 50$, $m = 50$, and $k = 50,000$ or more.

When you output a larger grid, wrap-around cannot be avoided on the display. Use formatting so that the smaller grids are lined up (e.g., grids 10×10 and smaller).

Requirements: Your program (modular) must perform the following operations:

- Read the grid size, starting position, and maximum iterations.
- Allocate an array (the grid) based on the grid size.
- Move the robot around the grid.
- Display the contents of the array when finished.

Deliverables:

- A program design—A description or structure chart of all functions needed to implement your program.
- Programming Log:
  - Record the time required to design and implement your program.
  - Record of things you encountered/learned while implementing your program.
- Output—proof that your program worked:
  - nRows = 15, nCols = 10, row = 7, col = 4, k = 250
  - nRows = 10, nCols = 15, row = 1, col = 1, k = 2500
  - nRows = 55, nCols = 50, row = 1, col = 1, k = 2500

  and some small test-case of your own choosing.
- Program—fully documented.

If you have any questions regarding this assignment, do not hesitate to contact me (please use email). Start working on this assignment as soon as possible.

For more information:
`bruceb@uidaho.edu` or
`http://www.cs.uidaho.edu/~bruceb`