EEEN10036
2018-19
C Programming Mini Assignment

Please read carefully - this assignment is assessed and is worth up to 10% of the overall course unit mark. You may choose between two versions which are worth different percentages of the available marks: "Basic" (worth 70%) or "Advanced" (worth 100%). A 'skeleton' program is provided on Blackboard to help you get started.

1 Assignment Overview

The objective of this assignment is to write a "C" program which estimates the probability that an explorer will safely escape from a dinosaur- and volcano-infested island ("Jurassic Island") by taking random walks across it. As well as calculating the probability of escape, the mean and standard deviation of the path lengths should be determined for each starting cell.

2 Jurassic Island Map

The island map is stored as a 9*9 array:

B	W	W	B	B	W	B	W	W
B	L	L	V	L	L	L	L	B
W	L	L	L	L	D	L	L	B
B	L	L	D	L	L	L	L	W
B	L	D	L	L	L	L	L	W
W	L	L	L	L	L	V	L	B
W	V	L	L	L	L	L	L	W
W	L	L	L	D	L	L	L	W
B	B	W	B	W	B	B	W	B

where each 'cell' is labelled as follows:

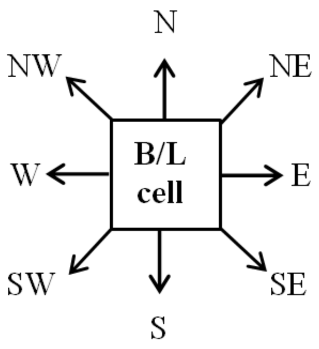
- **L**: Land, the explorer can safely step on this cell.
- **B**: Bridge, the explorer can safely step on this cell and use it to get off the island.
- **W**: Water, the explorer will die if this cell is stepped on.
- **D**: Dinosaur, the explorer will die if this cell is stepped on.
- **V**: Volcano, the explorer will die if this cell is stepped on.

The island should be represented in your program as a 2-D array. The only way to escape from the island is for the explorer to step onto a Bridge (**B**) cell *(this includes the case when **B** is a starting cell)*. Stepping onto a Land (**L**) cell is safe, however if the explorer steps on a Dinosaur (**D**), Volcano (**V**) or Water (**W**) cell, the walk is terminated and the next random walk should be attempted.

Each cell in the map is to be used as a starting point for walks, beginning, for example, with the **B** cell at the top-left corner (i.e. [0][0] in the 2-D array).

3 Random Exploration

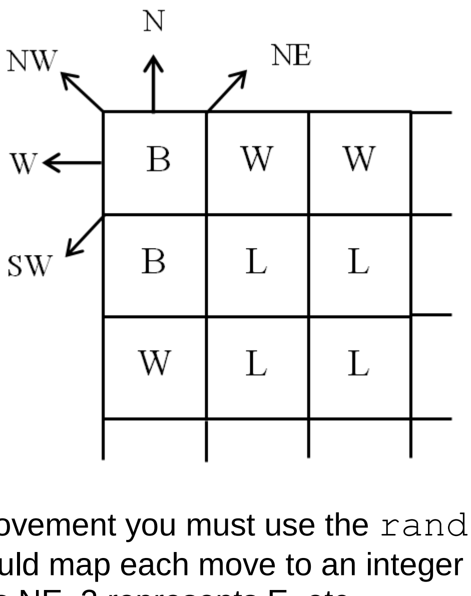
For each valid starting cell (i.e. a cell marked **B** or **L** on the map), the program should perform 1,000 random walks where the explorer attempts to get off the island. Each step on a walk is randomly chosen from one of eight directions, i.e. North, Northeast, East, Southeast, South, Southwest, West, or Northwest, with equal probability:



For each cell, the program should check to see if the walk:

- is **successful** by stepping off the island from a Bridge cell;
- has **failed** by stepping onto a Dinosaur, Volcano or Water cell;
- **continues** by stepping onto a Land or Bridge cell.

For example, assuming that the explorer is on the top-left corner cell (**B**), if the next step is to the NE, N, NW, W, or SW then they will have stepped off the island successfully; if the next step is to the S or SE the walk continues (lands on a **B** or a **L** cell); however, if the next step is to E then the explorer lands on a **W** cell and the walk terminates unsuccessfully:



To perform the random movement you must use the `rand()` function (explained in the Appendix). You should map each move to an integer value as follows: 0 represents N, 1 represents NE, 2 represents E, etc.

4 Results and Example Output

For every valid starting cell your program should compute to two decimal places

- the probability of successfully escaping from the island;
- the mean path length (i.e. number of steps) of successful walks;
- the standard deviation of path lengths of successful walks.

(Invalid starting cells have 0.00 for the above items)

The values should be displayed on the screen when the program terminates. Use of gcc is expected, using either the C89 or C99 "C" standard. An example output is shown below:

```
Percentage (probability) of getting off Jurassic Island
71.40  0.00    0.00   46.40   47.00    0.00   44.90    0.00    0.00
53.40 20.70   12.10    0.00   17.40   19.70   14.60   25.70   49.40
 0.00 22.50   12.70   10.60    7.40    0.00   13.10   24.30   51.50
50.30 21.60   12.40    0.00    5.40    7.30   10.30   14.30    0.00
51.80 16.10    0.00    5.60    5.20    6.60   11.30   10.40    0.00
 0.00 10.70    7.30    7.70    7.80    7.10    0.00   12.40   39.70
 0.00  0.00   10.00    8.90    8.20   10.40   14.00   13.70    0.00
 0.00 17.30   16.30   10.30    0.00   18.00   23.70   20.90    0.00
69.50 49.60    0.00   42.00    0.00   48.90   52.30    0.00   63.80

Mean path length when escaping
1.28  0.00    0.00    1.53    1.63    0.00    1.67    0.00    0.00
1.51  3.13    4.31    0.00    3.22    3.69    4.26    3.30    1.56
0.00  3.60    4.83    5.68    5.15    0.00    5.12    3.94    1.73
1.76  3.47    5.26    0.00    8.63    7.73    6.25    5.31    0.00
1.58  3.66    0.00    8.64    10.02   8.74    7.08    4.09    0.00
0.00  3.90    6.62    7.69    8.47    8.41    0.00    4.08    1.67
0.00  0.00    5.54    6.56    6.84    6.64    5.31    4.12    0.00
0.00  2.80    3.14    4.63    0.00    3.83    3.70    3.25    0.00
1.21  1.47    0.00    1.21    0.00    1.63    1.68    0.00    1.13

Standard deviation of path length when escaping
1.00  0.00    0.00    1.82    1.63    0.00    1.72    0.00    0.00
1.08  1.72    3.49    0.00    2.26    2.49    2.68    2.34    1.64
0.00  1.90    2.19    2.70    2.89    0.00    2.63    2.65    1.75
1.69  2.26    3.68    0.00    3.59    3.43    3.28    4.12    0.00
1.32  2.58    0.00    4.19    4.77    4.57    4.69    3.16    0.00
0.00  3.22    4.03    4.24    4.15    3.81    0.00    2.60    2.18
0.00  0.00    4.05    4.08    4.12    3.90    2.80    3.12    0.00
0.00  1.63    2.06    3.07    0.00    2.81    2.32    2.93    0.00
0.72  0.98    0.00    1.12    0.00    1.96    1.68    0.00    0.61
```

5 "Basic Version"

The basic version consists of implementing the program using only the **main** function to calculate everything and print out the results.

6 "Advanced" Version

For additional marks, the basic version can be extended by restructuring the program by implementing and using the following (incomplete) function prototypes*:*

```
void randomStep(…);
int status(…);
void printResults(…);
```

where

- `randomStep` calculates the next random step.
- `status` returns the status of the next step (0, 1 or 2, where 0 = "successfully made it off the island", 1 = "failure, terminate walk" and 2 = "continue walk").
- `printResults` prints out the results to the screen. `printResults` should be called three times, each time printing out the following:
    - the probability of successfully escaping from the island;
    - the mean path length (i.e. number of steps) of successful walks;
    - the standard deviation of the path lengths of successful walks.

The functions are incomplete in that you must specify the parameters needed; however you may not change their return types or names.

7 Deadline and Notes

The assessment will take place in week 9 (i.e. week beginning Monday 25th March 2019), and only in that week. *In the event that the assessment is not completed by the end of your first lab, the second lab that week will be used as an 'overspill'.*

Important:

- The assignment is **individual** and will be conducted under exam conditions. Plagiarism or collusion will result in zero marks and the matter will be reported for academic malpractice.
- The assessment will consist of demonstrating your solution to an examiner. Afterwards you will be asked to upload your solution to Blackboard and then answer one or more previously unseen questions.
- You are expected to have completed the assignment before you arrive. You must be able demonstrate your program when asked. **You are not permitted to use the lab time to work on the assignment.**
- This is a **closed-book** assessment, taken under examination conditions.
- You must implement the program as described. If not, marks may be deducted. See the marking scheme below.
- You must also answer one or more previously unseen questions, which are collectively worth 2 marks.
- You must provide evidence to the examiner that your program produces the expected results with a different map *(the map can be of your choosing, but must contain a mix of **B**, **L**, **D**, **V** and **W** cells).*

continued..

- On the day there will also be the usual set of lab tasks which you should aim to complete. Please note that there may be some waiting on the day due to the limited number of examiners, and your patience will be appreciated.

8 Marking Scheme ("Basic" Version)

Item	**Mark**
Using test data supplied by the examiner, the program correctly prints out the data as expected, i.e. the probability, the mean path length, and the standard deviation of the path lengths of all successful walks.	**3**
Able to explain how the program works to the examiner's satisfaction.	**1**
Evidence that testing has been carried out using a different map.	**1**
Answer previously unseen questions.	**2**

Marking Scheme ("Advanced" Version)

Item	**Mark**
Using test data supplied by the examiner, the program correctly prints out the data as expected, i.e. the probability, the mean path length, and the standard deviation of the path lengths of all successful walks.	**3**
Able to explain how the program works to the examiner's satisfaction.	**1**
Evidence that testing has been carried out using a different map.	**1**
The program correctly implements and uses the `randomStep` function.	**1**
The program correctly implements and uses the `status` function.	**1**
The program correctly implements and uses the `printResults` function.	**1**
Answer previously unseen questions.	**2**

Marks will be ***deducted*** for both "Basic" and "Advanced" versions as follows:

Item	**Mark**
One or more compiler warnings.	**1**
The appearance of the program is poor, for example is poorly structured, or is not indented, or has very few comments.	**1**
The program uses one or more global variables.	**2**

Appendix

A. Calculating Random Values

You are required to use the `rand()` function to randomly move in one of eight possible directions. `rand()`, which is part of the C standard library `stdlib.h`, returns a pseudorandom integer uniformly distributed within the interval given by [0, RAND_MAX], where RAND_MAX typically has a value of 32767. Note that `rand()`must be "seeded" with a value before it is first called*;* for this purpose use the `srand()` function with a suitable parameter: for example '`srand(time(NULL));`', where `time(NULL)` returns the number of seconds elapsed since 00:00:00 1st January 1970. Note that `srand()` must only be called once, e.g. at the beginning of the main function*.* To use the `time()` function, `time.h` should be included; to use the `rand()` and `srand()` functions, `stdlib.h` should be included.

B. Mean & Standard Deviation

The mean path length m for each cell is given by

$$m = (\Sigma\ st_i) / sum\_sw$$

where st_i is number of steps taken for the ith successful walk, and sum_sw is the total number of successful walks from that cell.

One way to calculate the standard deviation sd is as follows:

$$sd = sqrt\ (\ (\Sigma\ (st_i - m)^2\ ) / sum\_sw)$$

However this may not be the most efficient method of calculating sd, and you are encouraged to research alternatives. *To use the sqrt function include math.h.*