Problem Set 3 Exercise #28: Traversing a Topographical Map

Reference: Lecture 9 notes

Learning objectives: Two-dimensional array; Algorithm design

Estimated completion time: 70 minutes

Problem statement:

[CS1010 AY2017/18 Semester 1 PE2, Ex2]

A topographical (or contour) map is one that shows the elevations of an area of surface or terrain, indicating the hills and valleys, steepness or gentleness of slopes. Points with the same elevation are joined using a contour line. Figure 1 gives an example of a topographical map, showing several contour lines, and represents a hill that peaks at 280m. A contour line is one that joins points on a map with the same elevation. A topographical map is useful when one wants to navigate over a terrain. Usually you would want to avoid high areas (peaks) and low areas (valleys).

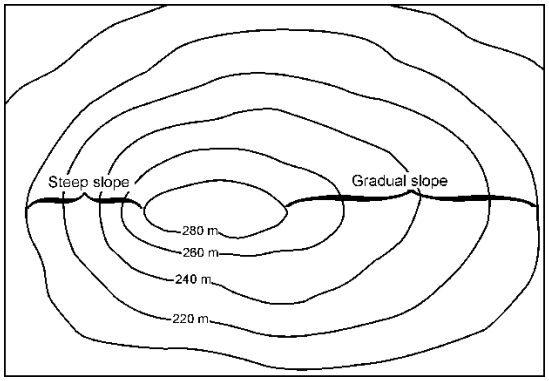

Figure 1: A topographical map showing several contour lines

A topographical map can be represented by a 2D array, where the elements of the array represent the elevation at those points in the terrain. Figure 2 (on next page) shows an example 2D array (6 rows and 8 columns) representing a particular terrain. In the figure, an example of a contour line of elevation 39 is shown in *italics*. **A peak** is defined as an element for which all eight neighbours are lower than that element. **A valley** is defined as an element for which all eight neighbours are higher than that element. In the figure, peaks are shown in bold and valleys are underlined. Note that no element on the perimeter may be considered a peak or valley, because they have fewer than eight neighbours.

25 58 53 23 21 34 21 50

32 45 43 40 **41** 32 30 27

34 40 *39* *39* *39* 28 30 35

40 *39* 42 **48** *39* 34 29 32

39 *39* *39* *39* *39* **49** 27 30

31 31 31 32 32 33 44 35

Figure 2: 2D array representation of a topographical map

In this exercise, you are to read in a 2D array that represents a topographical map of a terrain, count the number of peaks and valleys and show their locations. You may assume the array has a maximum of 12 rows and 12 columns.

For the example above, the following 2D array (**`peaksandvalleys[6][8]`**) shows the locations of the peaks (represented by 1) and valleys (represented by 2) as shown in Figure 3 below.

0 0 0 0 0 0 0 0

0 0 0 0 **1** 0 0 0

0 0 0 0 0 **2** 0 0

0 0 0 **1** 0 0 0 0

0 0 0 0 0 **1** **2** 0

0 0 0 0 0 0 0 0

Figure 3: peaks and valleys map

If you were planning to hike across this terrain, you would be constrained by the differences in elevation between neighbouring entries. A person can traverse between 2 adjacent locations if their elevations differ by no more than two. Adjacency refers to just the four basic compass directions, i.e. North, South, East, and West. Therefore, a point on the map is considered reachable if it is traversable from a starting point of **`map[0][0]`** (where **`map[row][col]`** is the topographical map) through any valid sequence of adjacent entries. In this exercise, you are required to compute all of the reachable locations from a starting point of **`map[0][0]`**. The output will be another 2D array **`hike[row][col]`** with values of 0s and 1s (1 means reachable, 0 unreachable).

For example, suppose a map is represented by **`map[10][10]`** as shown in Figure 4 below.

50 51 54 58 60 60 60 63 68 71

48 52 51 59 60 60 63 63 69 70

44 48 52 55 58 61 64 64 66 69

44 46 53 52 57 60 60 61 65 68

42 45 50 54 59 61 63 63 66 70

38 42 46 56 56 63 64 61 64 62

36 40 44 50 58 60 66 65 62 61

```
34 32 40 49 56 62 67 66 65 60
30 36 40 47 50 64 64 63 62 60
50 50 50 50 50 50 50 50 50 50
```

Figure 4: 2D array representation of a map

Both East and South are traversable from **`map[0][0]`** so reachable entries would be **`map[0][1]`** and **`map[1][0]`**. The reachable points for the map are shown in bold in Figure 5 below.

50 51 54 58 60 60 60 **63** 68 71
48 52 51 59 60 60 **63 63** 69 70
44 48 **52** 55 58 61 **64 64 66** 69
44 46 **53 52** 57 60 60 61 **65** 68
42 45 50 **54** 59 61 63 63 **66** 70
38 42 46 **56 56** 63 64 61 **64 62**
36 40 44 50 **58 60** 66 65 **62 61**
34 32 40 49 **56 62** 67 66 65 **60**
30 36 40 47 50 **64 64 63 62 60**
50 50 50 50 50 50 50 50 50 50

Figure 5: Reachable points shown in bold

And the resulting array **`hike[10][10]`** is as shown in Figure 6 below:

1 1 0 0 0 0 0 **1** 0 0
1 1 1 0 0 0 **1 1** 0 0
0 0 **1** 0 0 0 **1 1 1** 0
0 0 **1 1** 0 0 0 0 **1** 0
0 0 0 **1** 0 0 0 0 **1** 0
0 0 0 **1 1** 0 0 0 **1 1**
0 0 0 0 **1 1** 0 0 **1 1**
0 0 0 0 **1 1** 0 0 0 **1**
0 0 0 0 0 **1 1 1 1 1**
0 0 0 0 0 0 0 0 0 0

Figure 6: Corresponding hiking map for the map in Figure 4

Write in the skeleton file **contour.c** given to you. Functions to read and print the 2D array have already been provided. You are supposed to complete the following two functions.

- ```
  void getPeaksValleys(int map[][MAX_COL],
                       int peaksandvalleys[][MAX_COL],
                       int row, int col,
                       int *nopeaks, int *novalleys)
  ```

This function takes in the 2D map array, counts the number of peaks and valleys, and also produces the peaks and valleys map.

- ```
  void getHikeTrail(int map[][MAX_COL], int hike[][MAX_COL],
                    int row, int col)
  ```

This function takes in the 2D map array and produces the required hiking map.

You may define additional functions as needed. Check sample runs for input and output format.

Sample run #1:

```
Enter the number of rows and cols: 6 8
Enter the data:
25 58 53 23 21 34 21 50
32 45 43 40 41 32 30 27
34 40 39 39 39 28 30 35
40 39 42 48 39 34 29 32
39 39 39 39 39 49 27 30
31 31 31 32 32 33 44 35
Number of peaks = 3
Number of valleys = 2
Peaks and Valleys map
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 2 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 2 0
0 0 0 0 0 0 0 0
Hiking map
1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
```

Sample run #2:

```
Enter the number of rows and cols: 10 10
Enter the data:
50 51 54 58 60 60 60 63 68 71
48 52 51 59 60 60 63 63 69 70
44 48 52 55 58 61 64 64 66 69
44 46 53 52 57 60 60 61 65 68
42 45 50 54 59 61 63 63 66 70
38 42 46 56 56 63 64 61 64 62
36 40 44 50 58 60 66 65 62 61
34 32 40 49 56 62 67 66 65 60
30 36 40 47 50 64 64 63 62 60
50 50 50 50 50 50 50 50 50 50
Number of peaks = 1
Number of valleys = 1
Peaks and Valleys map
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 2 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
Hiking map
1 1 0 0 0 0 0 1 0 0
1 1 1 0 0 0 1 1 0 0
0 0 1 0 0 0 1 1 1 0
0 0 1 1 0 0 0 0 1 0
0 0 0 1 0 0 0 0 1 0
0 0 0 1 1 0 0 0 1 1
0 0 0 0 1 1 0 0 1 1
0 0 0 0 1 1 0 0 0 1
0 0 0 0 0 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0
```

Sample run #3:

```
Enter the number of rows and cols: 2 4
Enter the data:
20 20 21 21
21 21 22 22
Number of peaks = 0
Number of valleys = 0
Peaks and Valleys map
0 0 0 0
0 0 0 0
Hiking map
1 1 1 1
1 1 1 1
```