CS-M12 2014

Software Concepts and Efficiency

(Attempt 2 questions out of 3)

Question 1: Algorithm and Complexity

Suppose `arr` is a given array of non-negative integers of length `N > 5`. Consider the following function:

```
public static int something (int arr[], int N)
{
      int temp = Integer.MAX_VALUE;
      for (int i = 0; i < N-1; i++) {
            for (int j = i + 1; j < N; j++) {
                  if (arr[i] + arr[j] < temp) {
                        temp = arr[i] + arr[j];
                  }
            }
      }
      return temp;
}
```

a) What is the output of the function? (Describe the value of the variable `temp` when the function returns. Do not explain how the function executes the task.)

[5 Marks]

b) What is the time complexity of the function in Big-O notation [5 marks]?
Detail how you arrive at the time complexity [5 marks].

[10 Marks]

c) Write a snippet of code or pseudo-code which performs the same task but which is ***asymptotically faster*** (not a constant factor faster) in time complexity. Analyze the complexity of your algorithm in Big-O notation.

[10 Marks]

Question 2: Binary Tree and Binary Search Tree

a) Explain succinctly what a binary tree and binary search tree are.

[7 Marks]

b) Given a pointer to the root of a binary search tree, write **two** separate snippets of code or pseudo-code that find 1) the minimum key, and 2) the maximum key stored in the tree.

You should not use any library functions but can assume that the key at a tree node can be obtained using the `t.data` property and the children of a node `t` using the `t.left` and `t.right` properties. If there is no child, `t.left` and `t.right` store `null` value.

```
public class Node {
      public Node left;
      public Node right;
      public int data;
      Node(int newData) {
            left = null;
            right = null;
            data = newData;
      }
}
```

Use the following template:

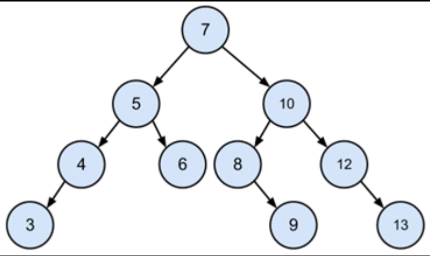
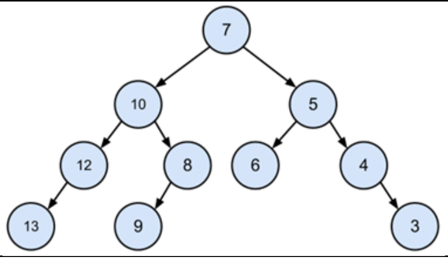
```
public int max(Node t) {
      //... to find the max value in the tree ...
}

public int min(Node t) {
      //... to find the min value in the tree ...
}
```

[8 marks]

---- Question continues on the next page ----

c) Given the root node of a binary tree, write a snippet of code or pseudo-code that returns a mirror of the binary tree. The mirror of a binary tree T is another binary tree M(T) with left and right children of all non-leaf nodes interchanged. For example:

Input:	Output:
	

You should not use any library functions but can assume that the child nodes of a tree node `t` can be accessed using the `t.left` and `t.right` properties. If there is no child, `t.left` and `t.right` store `null` value.

Use the following template:

```
public void mirror (Node t) {
     //... mirror a binary tree ...
}
```

so that the following method call will mirror a binary tree, where `root` is the root node of the tree:
`mirror(root);`

Hint: use recursion.

[10 marks]

Question 3: Graph and traversal

a) Explain succinctly what a directed graph is.

[5 marks]

b) Explain succinctly what a breadth first search (BFS) graph traversal algorithm is and what it is good for, *in plain English*.

Hint: check that your answers address the following:
1) what is the main purpose of a traversal algorithm,
2) a high-level description of how BFS works, and
3) what problem can BFS solve whilst depth first search (DFS) cannot?

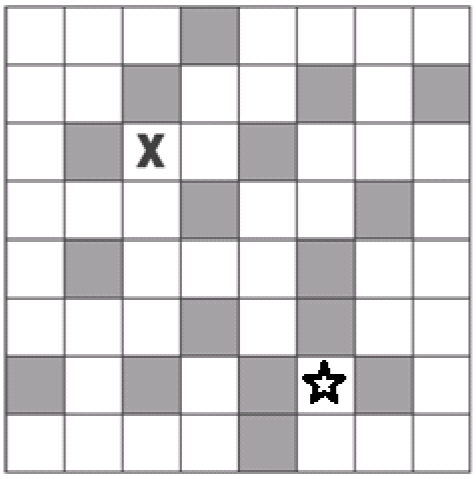
[5 marks]

c) Dungeon

A token (marked 'X' in the diagram) is in a maze. You may move the token around according to the following rule.

RULE: in each move the token may travel to ***any*** adjacent white square ***horizontally*** or ***vertically***, but it cannot pass over or stop on a shaded square.

For example, from its starting position the token could travel either one square right or one square down in a single move. To reach any other square would require more than one move.



0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

Suppose all the squares are labelled 0-63 (as shown on the right), what is the shortest path for the token 'X' (square 18) to reach the 'star' exit (square 53) from its starting position?

Write down the shortest path in terms of a sequence of the square labels e.g., 18, w_1, w_2, ... , 53. (Note, if the path is not unique, only one solution is required.)

[3 marks]

---- Question continues on the next page ----

d) The above problem can be solved using a graph traversal BFS algorithm. Write a snippet of code or pseudo-code by modifying the BFS algorithm to compute the shortest path from the token to the star exit. You can assume the following are provided:

Input:
`map[0:n^2-1]` - a 1D integer array indexed from 0 to n^2-1. Each element in the array represents a square which contains `0` (a shaded square) or `1` (a white square). For example, according to the rule and map in part (c), a token on square `map[18]` can be moved to `map[19]`, but not `map[17]` because `map[19]==1` and `map[17]==0`.
`src` - an integer, position of the token (e.g. `src == 18` in part c).
`dst` - an integer, location of the star exit (e.g. `dst == 53` in part c).

Output:
`pathTo` - an array that stores the move. For example, a move to the white square 53 from 61 is stored as `pathTo[53]=61`. A move to the white square 61 from 62 is stored as `pathTo[61]=62`. And finally `pathTo[18]=-1`.

Hint: use the following template.

```
public void bfs() {
      int N = n*n;     // N - number of total squares
      marked = new boolean[N];
      pathTo = new int[N];

      for (int v = 0; v < N; v++) {
            marked[v] = false;
            //... initialization
      }

      Queue q = new Queue();
      q.enqueue(new Integer(src));
      marked[src] = true;

      while (!q.isEmpty()) {
            int v = (int)q.dequeue();

            // for each adjacent white square w of v
            for (...) {
                  //...

                  if (!marked[w]) {
                        marked[w] = true;
                        //...

                        q.enqueue(w);
                  }
            }
      }
}
```

[8 marks]

---- Question continues on the next page ----

e) Write a snippet of code or pseudo-code to print out the path **from** the token 'X' **to** the star exit **(i.e. solution of part c)**. Use the computed `pathTo` array in part d. Use the following template where input `dst` is the location of the star exit.

```
public void printPathTo(int dst) {
      ...
}
```

[4 marks]

---- END ----

Additional mazes for Question 3c