SER 222 **Practice** Exam 3

Updated 11/23/2021b

Last Name: _____
First Name: _____
Last 4 digits of ASU ID: _____

Exam Instructions

The exam is open one note card (3x5 inches). No electronic items are allowed. Write legibly. Please use a pen (instead of a pencil) if you have one. There are 110 points available and the exam must be completed in 60 minutes. This exam has three types of questions:

Multiple choice questions: There are 35 points of multiple choice questions. An answer is selecting one option among the choices given. Each multiple choice is worth 2 to 5 points.

Short answer questions: There are 30 points of short answer questions. A typical answer is one or two sentences. Each short answer question is worth 5 or 10 points.

Programing questions: The programming questions are given near the end of the paper. They must be answered on the question paper. There are 40 points of write-in programming questions.

Topic	**Earned**	**Possible**
MC/SA: Binary Search Trees		20
MC/SA: Hashtables		20
MC/SA: Undirected Graphs		10
MC/SA: Directed Graphs		15
Prog: Binary Search Trees		30
Prog: Undirected Graphs		10
Prog: Directed Graphs		0
Total:		105

Short Answer: Symbol Tables & Binary Search Trees

1. [Acuña] Consider the following scenario: you need to store a record for each of the top 1000 teams that participated in a machine learning competition, in order to be able to look up which team placed in a specific place. Would it make more sense to use an array or a symbol table for this problem? Explain. [5 points]

   (a) An array will probably be better. It will be faster and more efficient in storage.

   (b) An array will probably be better. It will allow the team record type to be stored properly.

   (c) A symbol table will probably be better. It will provide more flexibility.

   (d) A symbol table will probably be better. It will be more efficient in storage.

2. Consider the following method discussed in lecture for deleting a node from a BST. What functionality is provided by the indicated line? [5 points]

```
private Node delete(Node x, Key key) {
    if (x == null) return null;
    int cmp = key.compareTo(x.key);
    if (cmp < 0) x.left = delete(x.left, key);
    else if (cmp > 0) x.right = delete(x.right, key);
    else {
        if (x.right == null) return x.left;
        if (x.left == null) return x.right;
        Node t = x;
        x = min(t.right);
        x.right = deleteMin(t.right);
        x.left = t.left;  //THIS LINE HERE
    }
    x.N = size(x.left) + size(x.right) + 1;
    return x;
}
```

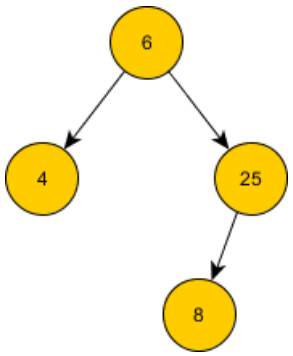
   (a) Finds the in-order successor to x.

   (b) Replaces the current sub-tree rooted at x with the new sub-tree without the minimal node.

   (c) It sets the old left sub-tree of x to be a child of the successor node being moved up.

   (d) Updates the size variable of the tree rooted at x.

3. Trace a (non-balancing) BST through the following operations:




```
//assume that the "tree" variable below has already been constructed,
//per the tree shown above.
tree.add(new Integer(3));
tree.add(new Integer(7));
tree.add(new Integer(2));
tree.remove(new Integer(6));
```

Draw the structure of the BST after each call. Images must be cumulative. [10 points]

Short Answer: Hashtables

4. Consider implementing a hashcode function for a string class. Which of the following statement(s) is/are True? Select all that apply. [5 points]

   (a) A hash function takes a string of arbitrary length and generates a fixed sized hash.

   (b) A hash function takes a string of fixed length and generates a hash of variable size.

   (c) A hash function may give different hash values for identical strings.

   (d) None of the above are true.

5. Is the hashCode() implementation in the following Point2D class valid? That is, would a hashtable be able to properly store keys with this hashCode? Explain. [10 points]

```
public class Point2D {
    public int x, y;

    public Point2D(int x, int y) { this.x = x; this.y = y;}

    public int hashCode() {
        return x % Integer.MAX_VALUE;
    }
}
```

   (a) Yes - it implements the required hashCode method using the attributes in the class.

   (b) Yes - since MAX_VALUE is larger than any number, the modulus doesn't change the value of the point, and no information is lost.

   (c) No - different Point2Ds might have the same hashcode.

   (d) No - it is missing the modulus by M.

6. Imagine that you have a linear probe hashtable, with M=11. Draw the final hashtable after adding these keys: 34, 13, 10, 3, 0, 15, 14. Use the hash function $hash(k, i) = (k \ mod \ 11 + i) \ mod \ 11$, where i is the number of times the algorithm has tried to insert the key. Your drawing should include the main size M array. [10 points]

Short Answer: Undirected Graphs

7. Consider using the DFS algorithm. From a performance standpoint, would it be better for the algorithm to use a Graph ADT implemented using an adjacency matrix or adjacency list? Explain. [5 points]

   (a) Adjacency matrix - it will provide faster support for visiting each of the adjacent nodes.

   (b) Adjacency matrix - it will provide faster support for checking if two nodes are connected.

   (c) Adjacency list - it will provide faster support for visiting each of the adjacent nodes.

   (d) Adjacency list - it will provide faster support for checking if two nodes are connected.

8. In which of the following scenario(s) would you select using the DFS algorithm over BFS? Select all that apply. [5 points]

   (a) You have limited memory available for storing intermediate results and you are given a wide graph.

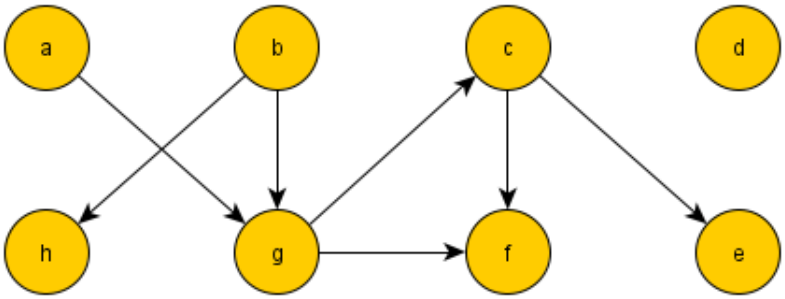
   (b) You need to find the optimal solution that is the shortest path for the given problem.

   (c) The solution is far away as in many levels away from the source node.

   (d) When are you implementing a peer to peer network like a social networking site where you are attempting to find the degree of connection between two individuals. The degree of connection means the number of intermediate friend introductions you need in order to get to the required person.

Short Answer: Directed Graphs

9. For the given directed graph, which of the following sequences is not a topological sort? [5 points]



(a) a, b, h, g, c, d, f, e

(b) a, b, c, d, e, f, g, h

(c) a, b, g, c, e, d, f, h

(d) All of these are valid topological sorts.

10. When using DFS to check for a cycle in a directed graph, can we simply check for the algorithm visiting a marked node? Explain yes or no. [10 points]

Programming: Binary Search Trees

On the real exam, there could be programming questions on BSTs, hash tables, or graphs.

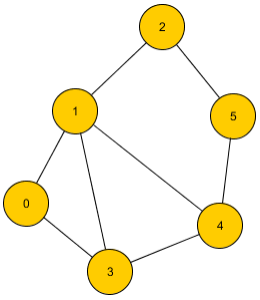
11. One of the data structures we studied is Binary Search Trees (BST). A simple implementation of a node for a binary tree is shown below. For this question, you are to implement a method called isBST that takes first node in a binary tree (its root) and returns true if it is the root of a BST, and false otherwise. Be careful with your syntax. [30 points]

```
private class Node {
    public final Key key;
    public Value val;
    public Node left, right;
    public int N;

    public Node(Key key, Value val, int N) {
        this.key = key; this.val = val; this.N = N;
    }
}


private static boolean isBST(Node root) {
```

Programming: Undirected Graphs

12. Run the **BFS** algorithm on this graph to compute the shortest paths between 0 and every other node. For reference, the BFS algorithm is shown below. Use the adjacency list above for the order of the nodes explored and follow the trace format shown before. Your answer must include the values of v, *queue*, and *edgeTo*, as they update. [10 points]

```
private void bfs(Graph G, int s) { //by Sedgewick. for reference.
    Queue<Integer> queue = new LinkedList<>();
    marked[s] = true;
    queue.add(s);
    while (!queue.isEmpty()) {
        int v = queue.remove();
        for (int w : G.adj(v))
            if (!marked[w]) {
                edgeTo[w] = v;
                marked[w] = true;
                queue.add(w);
            }
    }
}
```



0	3, 1
1	0, 4, 3, 2
2	1, 5
3	1, 0, 4
4	1, 3, 5
5	2, 4

Figure 1: Sample graph.

Before loop:
marked[ ]=
queue={ }
Loop 1:
v=
edgeTo[ ]=
edgeTo[ ]=
queue = { }

Extra Questions

The following questions were used on previous practice exams - they are not part of the practice exam, and may use content not covered in the current semester, but are provided for additional practice.

Short Answer: Symbol Tables & Binary Search Trees

1. [Acuña] Trace an initially empty symbol table (called ST) through the following operations:

```
SymbolTable<> ST = new BinarySearchTree<String, Integer>();
ST.put("CSE110", 150);
ST.put("CSE205", 150);
ST.put("CSE230", 100);
ST.put("SER316", 100);
System.out.println(ST.get("CSE205"));
ST.put("CSE205", 250);
ST.put("SER316", 75);
```

   Give the contents of the symbol table after the code has been executed. Use the format ABC### : ### (e.g., "CSE110 : 150") to give your answer. Use separate lines for each key/value pair. [10 points]

2. Consider the task of implementing a SymbolTable vs an OrderedSymbolTable. For the purposes of get and put, which you be more likely to create an efficient implementation for? Explain. (Hint: no knowledge of BSTs or hash tables is needed.) [10 points]

3. When adding and removing nodes from a heap, the tree structure will always remain balanced. Why is it the case that when these operations are performed over a BST, there is a chance that the tree will become unbalanced? [5 points]

   (a) In a BST there is exactly one place a node may be added.

   (b) In a BST nodes are always placed in level nearest to the root node.

   (c) Because heaps use an array, while BSTs use a linked node structured.

   (d) This is is a trick question, both BSTs and heaps are always balanced.

Short Answer: Undirected Graphs

4. If you are asked to run a DFS or BFS by hand, and given a picture of the graph, and the code for the algorithm, why is there a chance you and a classmate who correctly execute the algorithm will produce different paths? [10 points]

Short Answer: Directed Graphs

5. In the DFS based topological sort, what would happen if nodes were pushed on the stack before their children were explored, rather than after? Would the algorithm still work? Explain. [10 points]