CMPT225, Spring 2021

Midterm Exam

Name_____

SFU ID: |__|__|__|__|__|__|__|__|__|

Problem 1	
Problem 2	
Problem 3	
Problem 4	
TOTAL	

Instructions:

1. You should write your solutions directly in this word file, and submit it to Coursys. Submitting a pdf is also ok.
2. Submit your solutions to Coursys before March 12, 23:59. No late submissions, no exceptions
3. Write your name and SFU ID on the top of this page.
4. This is an open book exam. You may use textbooks, calculators, wiki, stack overflow, geeksforgeeks, etc. If you do, specify the references in your solutions.
5. Discussions with other students are not allowed. Posting questions online asking for solutions is not allowed.
6. The exam consists of four (4) problems. Each problem is worth 25 points.
7. Write your answers in the provided space.
8. You may use all classes in standard Java, and everything we have learned.
9. Explain all your answers.
10. **Really, explain all your answers.**

Good luck!

Problem 1 [25 points]

A. (3 pts each) For each sentence decide whether it is True or False.
Write a brief explanation.

1) Let $T(n) = 10n^4 + 5n + 3$. Then $T=\Omega(n^3)$.

2) Let $T(n) = 10^n$. Then $T=\Theta(2^n)$.

3) For all positive integers n, the function foo(n) will return 0.

```
public int foo(int n) {
    if (n>0)
        foo(n-1);
    return 0;
}
```

4) For all integers n (positive or negative), the function bar(n) will return 0.

```
public int bar(int n) {
    if (n>0)
        return bar(2*n);
    return 0;
}
```

B. (5 pts) Use big-O notation to express the running time of foo() on an array of length n. **Explain your answer.**

```
public void foo(int array[]) {
        fooRec(array, 0, array.length-1);
}

public static void fooRec(int array[], int start, int end) {
        if (start == end)
                array[start]++;
        else {
                int mid = (end + start)/2;
                fooRec(array, start, mid);
                fooRec(array, mid+1, end);
                fooRec(array, mid+1, end);
        }
}
```

C. (4 pts) Use big-O notation to express the running time of bar() on an array of length n. **Explain your answer.**

```
public void bar(int array[]) {
        barRec(array, 0, array.length-1);
}

public static void barRec(int array[], int start, int end) {
        if (start <= end) {
                for (int i = start; i <= end; i++)
                        a[i] += 1;
                barRec(a, start+1, end-1);
        }
}
```

(4 pts) Rewrite bar() so that it has the same functionality, but the running time is O(n). **Explain your answer.**

Problem 2 [25 points]

A. (15 pts) Write a class StackReverse that supports the following operations, with running time O(1) for each operation.

```
public class StackReverse<T> {

        public StackReverse() - a constructor, creates an empty stack

        public void push(T item) - adds an element to the stack

        public T pop() - removes an element from the stack

        public void reverse() - reverses the order of the elements. That is,
               the element that was the last in the stack becomes the first,
               and vice versa

        public int size() - returns the number of elements in the stack

        public boolean isEmpty() - checks if the stack is empty


}
```

The running time of each operation must be O(1).

Before writing code, explain your answer.

B. (10 pts) Write an algorithm that gets an expression as a String in prefix notation and returns a String with the expression in postfix notation.
For example, on input "/ * 2 + 3 4 - 18 16" the method returns "2 3 4 + * 18 16 - /".
You may assume the input is always valid
Explain your answer.

```
public String prefix2postfix(String prefix) {

}
```

Problem 3 [25 points]

In this problem use the following definition of Binary Tree. You may assume the classes have the standard getters/setters.

```
public class BTNode<T> {
    private T data;
    private BTNode<T> leftChild;
    private BTNode<T> rightChild;
    private BTNode<T> parent;

}

public class BinaryTree<T> {
    private BTNode<T> root;

}
```

A. (10 pts) Write the method equals(Object other) for the class Binary Tree. The method returns true if the argument is a BinaryTree with the same data. You should compare the data in different nodes using equals() method in the class T.

The running time should be O(n) in the worst case, where n is the size of the smaller tree. For example, if one tree has n vertices, and the other has n^2 vertices, the running time should be O(n). **Explain your algorithm and running time**.

```
public boolean equals(Object other) {

}
```

B. (10 pts) Write a method for the class Binary Tree that returns the depth of the shallowest leaf in the tree. That is, among all leaves, you need to return the smallest depth of a leaf. What is the running time of your algorithm
Explain your algorithm and running time.

```
public int depthOfShallowestLeaf() {

}
```

C. (5 pts) Write a definition of a Ternary Tree in Java. Each node has data of a generic type, a pointer to the parent, and at most three children: left, middle, and right.

```
public class TernaryTree<T> {

}
```

Problem 4 [25 points]

A. (6 pts) Let A = [4, 5, 10, 6 , 7, 12, 14, 8, 15, 2].

- Apply build-minHeap algorithm on A using the linear time algorithm we saw in class.
- Draw the tree representation of the heap.
- Draw the array representing the heap.
- **Draw the intermediate steps.**

Recall the buildHeap algorithm:

```
Treat the array as a complete binary tree
For each vertex v starting from the bottom
    Apply heapify(v)
```

B. (3 pts) Apply removeMin() on the heap obtained in part A, and draw the resulting heap.

C. (3 pts) Add 3 to the heap obtained in part B, and draw the resulting heap.

D. (13 pts) Write a function that gets an array A of length n of integers, and $0 \leq k \leq n$, and returns an array B of length k containing the smallest k elements in A. In the end A must be in the same state as in the beginning.
The **running time** must be **O(n log(k))** and **extra space used** should be **O(k)**.

For example, on input A =[4,1,5,7,2,3,1,3] and k=4 the output should be B = [1,1,2,3]. The order of the elements in B is not important

Explain your idea before writing the code.