CMPT 225 D100, Fall 2025

Midterm Exam
October 21, 2025

Name_____

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

Problem 1	
Problem 2	
Problem 3	
Problem 4	
TOTAL	

Instructions:

1. The duration of the exam is 100 minutes.
2. Write your full name and SFU ID **clearly**.
3. This is a closed book exam, no calculators, cell phones, or any other material.
4. The exam consists of four (4) problems, each worth 25 points
5. Write your answers in the provided space.
6. There is an extra page at the end of the exam. You may use it if needed.
7. Explain all your answers.
8. Really, explain all your answers.

Good luck!

Problem 1 (Java syntax, Big-O notation) [25 points]

Consider the following class. Assume it has a proper constructor and all setters/getters.

```
class Point {
    int x;
    int y;

    // Constructors,setters,getters

    @Override
    public boolean equals(Object other) {
        if (other == null || getClass() != other.getClass())
            return false;
        Point point = (Point) other;
        return x == point.x && y == point.y;
    }
}
```

A. (5 points) What will be the output of the following code.

```
Point p1 = new Point(1,2);
Point p2 = new Point(1,2);
if (p1 == p2)
   System.out.println("p1 == p2 is true");
if (p1.equals(p2))
   System.out.println("p1.equals(p2) is true");
```

B. (5 points) Explain the purpose of the if statement in the code

```
        if (other == null || getClass() != other.getClass())
            return false;
```

C. (5 points) Prove that the running time of the function foo(n) is O(n log(n)).

```
int foo (int n) {
   int sum = 0, i, j;
   for (i=0; i<n; i++)
       for (j=1; j<i; j=j*2)
           sum += 1;
   return sum;
}
```

D. (10 points) Use big-O notation to express the running time of the function sort() on an array of length n? **Explain your answer.**

```
public static void sort(int array[]) {
   sortHelper(array, 0, array.length-1);
}

public static void sortHelper(int a[], int start, int end) {
   int len = end-start+1;
   if (len<=3) {
       // sort the subarray a[start…end] of length <= 3
       Arrays.sort(a, start, end+1);
       return;
   }
   else { // len>3, we apply recursion
       if  (start < end) {
           int k1 = start + len/3;
           int k2 = start + 2*len/3;
           sortHelper(a, start, k2);
           sortHelper(a, k1, end);
           sortHelper(a, start, k2);
       }
   }
}
```

Problem 2 (Stack, Queues, Linked Lists) [25 points]

A. Write a class `QueueOfInts` that supports the operations of a queue, with running time O(1) for each operation.

```
public class QueueOfInts {

   public QueueOfInts() constructor creates an empty queue of ints

   public void enqueue(int item) adds an item to the queue

   public int dequeue() removes an item from the queue and returns it

   public int size()  returns the number of elements in the queue

   public boolean isEmpty() checks if the queue is empty

   public int sum()  returns the sum of all numbers in the queue.
                     The method does not modify the queue.

}
```

Implementing correctly all methods except for `sum()` is worth up to 15 points.

Before writing code, explain your answer.

Problem 3 (Binary Trees) [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 points) Write the method countLeaves() for the class Binary Tree. The method returns the number of leaves in the tree.
Explain the idea of the algorithm and its running time.

```
   public int countLeaves() {

   }
```

B. (10 points) Write the method inOrderTraversal() for the class Binary Tree. The method returns an ArrayList that contains the in order traversal of the tree.
Explain the idea of the algorithm and its running time.

```
public ArrayList<T> inOrderTraversal() {




}
```

C. (5 points) Draw ***all*** Binary Trees whose preOrder traversal is [1,2,3,4,5].

Problem 4 (Binary Search Trees) [25 points]

A. (10 points) Consider the class BinarySearchTree, defined as follows.

```
public class BinarySearchTree<T extends Comparable<T>> {
  private BTNode<T> root;
};
```

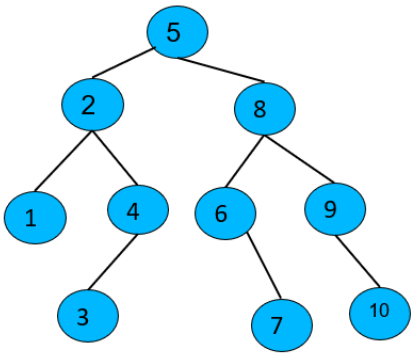
Use the definition of BTNode from Problem 3.

Write the method find() for the class BinarySearchTree. It gets an item, and returns the node containing elt or returns `null` if elt is not in the tree.

```
public BTNode<T> find(T elt) {









}
```

B. (5 points) Given the Binary Search Tree below.

- Remove 4 and 5 from the tree and draw the result.



C. (10 points each question) For each of the following statements, decide if it is True or False. **Prove your answer**.

1. There exists an AVL tree of height 3 (i.e., with 4 levels) with at most 3 leaves.

2. There exists an AVL tree of height 6 with 13 leaves.

Extra page

Master Theorem

Let $T(n) = a\,T(n/b) + f(n)$, $T(1) = O(1)$

Define $c = \log_b(a)$

- If $f(n) = O(n^d)$ for $d<c$, then $T(n) = \Theta(n^c)$
- If $f(n) = \Omega(n^d)$ for $d>c$, then $T(n) = \Theta(n^d)$
- If $f(n) = \Theta(n^c)$, then $T(n) = \Theta(n^c \log(n))$