Lab 09 Computer Science 1027B

Introduction

The purpose of these labs is to provide practice problems for you to get used to the concepts being covered in class. It is very important that you do these labs on your own without relying on available AI tools. Questions on exams are often very similar to exercises presented in these labs, so understanding them and solving these problems gives you a huge advantage for the exams. Make sure that you complete both the code as well as provide written answers to the blue questions within the boxes.

When you have concluded this lab, you should submit the following files to OWL Brightspace:

- BinaryTreeDepth.java
- MinimumDaysToPrintStatues.java
- UtilityMethodsTests.java
- written_answers.txt

Topics Covered

- Debugging

Exercise 01

The following code is intended to calculate the **depth** of a binary tree. However, there are bugs in the code. Identify these bugs and fix the code so that the program correctly calculates and prints the depth of the given tree.

```java
class TreeNode {
    int value;
    TreeNode left;
    TreeNode right;

    TreeNode(int value) {
        this.value = value;
        left = null;
        right = null;
    }
}
```

```java
public class BinaryTreeDepth {
    public static void main(String[] args) {
        TreeNode root = new TreeNode(1);
        root.left = new TreeNode(2);
        root.right = new TreeNode(3);
        root.left.left = new TreeNode(4);
        root.left.right = new TreeNode(5);
        root.right.right = new TreeNode(6);

        int depth = calculateDepth(root);
        System.out.println("The depth of the tree is: " + depth);
    }

    public static int calculateDepth(TreeNode node) {
        if (node == null) {
            return 1;
        } else {
            int leftDepth = calculateDepth(node.left);
            int rightDepth = calculateDepth(node.right);

            return Math.min(leftDepth, rightDepth) + 1;
        }
    }
}
```

Q 01.1
Write down the steps that you took to solve the problem. How did you approach it?

Q 01.2

While there are many, discuss four edge cases that might be useful to test to ensure that this method is working as intended.

Exercise 02

Your friend sends you their code to look over, claiming they can't seem to figure out what's wrong. They are trying to write a solution in Java for the following problem:

https://open.kattis.com/problems/3dprinter

The following is their code:

```
import java.util.Scanner;

public class MinimumDaysToPrintStatues {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.close();

        int days = 0;
        int printers = 2;

        // while we do not have enough printers to print the statues in one day
        while (printers <= n) {
            printers *= 2;
            days += 2;
        }

        // add the days required to print the remaining statues with the available printers
        days += (int) Math.ceil((double) n / printers);

        System.out.println(days);
    }
}
```

Debug their program, solving whatever issues are causing their solution to function incorrectly. When you are done, plug your solution into the 'Edit & Submit' tab on the webpage linked above to see if you're passing the test cases and to ensure you've effectively debugged your friend's code.

Q 02.1

Write down the steps that you took to solve the problem. How did you approach it?
How would you explain to your friend how they could debug their own code next time?

Exercise 03

Assume that you are writing a class that tests the functionality of the methods in the **UtilityMethods** class:

```java
public class UtilityMethods {

    // method to find the maximum of two numbers
    public int findMax(int a, int b) {
        return a > b ? a : b;
    }

    // method to reverse a string
    public String reverseString(String input) {
        StringBuilder reversed = new StringBuilder(input);
        return reversed.reverse().toString();
    }

    // method to check if a number is prime
    public boolean isPrime(int number) {
        if (number <= 1) return false;
        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) return false;
        }
        return true;
    }

    // method to find the factorial of a number
    public int factorial(int number) {
        if (number == 0) return 1;
        return number * factorial(number - 1);
    }

    // method to find the sum of elements in an array
    public int sumArray(int[] array) {
        int sum = 0;
        for (int num : array) {
            sum += num;
        }
        return sum;
    }

    // method to find the largest element in an array
    public int findLargest(int[] array) {
        int largest = array[0];
        for (int num : array) {
            if (num > largest) {
                largest = num;
            }
        }
        return largest;
    }

    // method to check if a string is a palindrome
    public boolean isPalindrome(String input) {
        String cleaned = input.replaceAll("\\s+", "").toLowerCase();
        String reversed = new StringBuilder(cleaned).reverse().toString();
        return cleaned.equals(reversed);
    }

    // method to sort an array in ascending order
    public int[] sortArray(int[] array) {
        int[] sortedArray = array.clone();
        for (int i = 0; i < sortedArray.length - 1; i++) {
            for (int j = 0; j < sortedArray.length - 1 - i; j++) {
                if (sortedArray[j] > sortedArray[j + 1]) {
                    int temp = sortedArray[j];
                    sortedArray[j] = sortedArray[j + 1];
                    sortedArray[j + 1] = temp;
                }
            }
        }
        return sortedArray;
    }
}
```

The class should be called **UtilityMethodsTests** and should contain at least two tests for each method provided within the UtilityMethods class (consider edge cases). Execute all of the tests in the main method within UtilityMethodsTests and print useful information telling the user of your class which tests and passing/failing.

Q 03.1

Explain the importance of using test classes in software development.
How do they contribute to the debugging process?
In cases of large and demanding projects, what are the potential risks of not regularly testing code functionality?