Exercises: $\frac{100+100+100}{3} = 100$

TOTAL $\frac{100+100}{2} = \boxed{100}$

Program: 100

Tamir Krief, Iaian Milton, Blessing Abumere
COSC 336
9/12/2024

Assignment 1

Exercise 1.

100

1. Theta Evaluations for $T_a(n), T_b(n), T_c(n)$:

$$T_a(n) = \Theta(n) \quad ✓$$

$$T_b(n) = \Theta(n) \quad ✓$$

$$T_c(n) = \Theta(\log(n)) \quad ✓$$

2. Is $T_b(n) = O(T_a(n))$? ✓ **Yes, they both have the same asymptotic growth rate.**

3. Is $T_c(n) = \theta(T_a(n))$? **No,** ✓ **since $\Theta(\log n)$ grows slower than $\Theta(n)$, $\Theta T_a(n)$ is not equal to $T_c(n)$.**

100

Exercise 2. Example of a function $f(n)$ with the property that $f(n)$ is $\omega(n^2)$ and also $f(n)$ is $o(n^3)$:

$$f(n) = n^2(\log n) \quad ✓$$

100

Exercise 3. Running time of the program:

$$\Theta(n^2) \quad ✓$$

Programming Task 1.

java table of results

2, 5,5,1,11,11,11,3,5,5,5,5,4,7	4	✓
1,0,0,1,1,1,0,0,0,1,1,1,1,0,1,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1	8	✓
1,2,2,3,3,3,4,4,4,4,5,5,5,5,5,6,6,6,6,6,6,7,7,7,7,7,7,7,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2	17	✓
Random Sequence	14	✓

java pdf file:

```
/** Group Members: Tamir Krief, Iaian Milton, Blessing Abumere */
import java.util.Random;

public class Assignment1 {

    public static void main(String[] args){
        int[] sequence1 = {2,5,5,1,11,11,11,3,5,5,5,5,4,7};
        int[] sequence2 =
{1,0,0,1,1,1,0,0,0,1,1,1,1,0,1,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1};
        int[] sequence3 = {1, 2,2,3,3,3,4,4,4,4,5,5,5,5,5,6,6,6,6,6,6,7,7,7,7,7,7,7,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2};
        int[] sequence4 = GenerateBits(4000);

        System.out.print("Max Continuous Subsequences");
        System.out.println(
            "\nSequence 1: " + MaxContinuousSubsequence(sequence1) +
            "\nSequence 2: " + MaxContinuousSubsequence(sequence2) +
            "\nSequence 3: " + MaxContinuousSubsequence(sequence3) +
            "\nPsuedoRandom Sequence of Bits: " + MaxContinuousSubsequence(sequence4)
        );
    }

    /** generates an array of bits using COUNT */
    public static int[] GenerateBits(final int COUNT){
        return GenerateBits(COUNT,new Random());
    }


    /** generates an arrays of bits of size {COUNT} and uses random object for the psuedorandom
part*/
    public static int[] GenerateBits(final int COUNT,Random random){
        if (COUNT < 0) throw new IllegalArgumentException("Positive numbers only");

        int[] bits = new int[COUNT]; //array of bits

        //generates either a 1 or 0 using random.nextBoolean() and puts in array
        for (int i=0; i<COUNT; i++)
            bits[i] = random.nextBoolean() ? 0 : 1;        ✓

        return bits;
    }
    /** returns the number of max continuous subsequence
     * BaseCase: Works by first checking if array length is 0 and returns 0 if it is
     * d[0]? : Initializes max and count to 1; Computed by checking if the current bit is the same as
the last one; d[i] == d[i-1]
     * O(n) : Starts array at index 1 and Loops through it and checks if the current bit is the same
as the last one each time
     *          if current bit is same as the last one then count goes up by 1
     *          else: if the curret bit isnt the same as the last one then count and max are compared
and count is reset to 1
     *              if count is greater than max then max is set to count
    */
    public static int MaxContinuousSubsequence(int[] bits){
        if (bits.length == 0) return 0; //base case

        //initializes max and count to 1
        int max = 1;
        int count = 1;

        //goes through the array and checks if the current bit is the same as the last one
        for (int i = 1 ; i < bits.length ; i++){
            if (bits[i] == bits[i-1]) //if current bit is same as the last one then count goes up by    ✓
1
                count++;
            else {
                if (count > max) //if count is greater than max then max is set to count
                    max = count;

                count = 1;
            }

        }

        //for the case of when every bit is the same
        if (count > max)
            max = count;


        return max;
    }
}
```

✓

You did not use d[0], d[1], ... , but your logic is correct.