Homework 1

Graded

Student

Giancarlos Marte

Total Points

90 / 104 pts

Question 1

Geometric Series

6 / 6 pts

✔ **– 0 pts** Correct

– 6 pts Not answered

– 3 pts part a

– 3 pts part b

– 1 pt part b not complete/ final answer not correct

– 1 pt part a not complete/ final answer not correct

Your final score for HW1: 9

Question 2

Binary Digits in Number

8 / 10 pts

– 0 pts Correct

– 10 pts Not answered / Completely incorrect

– 6 pts part a not answered

– 4 pts part b not answered

– 2 pts part a not complete/incorrect

✔ **– 2 pts** part b not complete/incorrect

Question 3

Log functions of different bases

10 / 10 pts

- ✔ **– 0 pts** Correct
- **– 10 pts** Not answered
- **– 5 pts** part a not correct
- **– 5 pts** part b not correct/ not answered
- **– 2 pts** part a not complete
- **– 2 pts** part b not complete
- **– 10 pts** Cheating!

Question 4

Growth Rates of functions

5 / 7 pts

- **– 0 pts** Correct
- **– 7 pts** Not answered
- ✔ **– 2 pts** Not completely correct
- **– 7 pts** Cheating!

Question 5

Big Oh, non-recursive functions

12 / 14 pts

- **– 0 pts** Correct
- **– 14 pts** Not answered
- **– 3 pts** part a not correct
- **– 3 pts** part b not correct
- **– 2 pts** part c.1 not correct
- ✔ **– 2 pts** part c.2 not correct
- **– 3 pts** part d not correct
- **– 14 pts** Cheating!

Question 6

Big Oh, recursive function

15 / 17 pts

- **– 0 pts** Correct
- **– 17 pts** Not answered
- **– 3 pts** a)recurrence formula
- **– 4 pts** a)runtime
- **– 3 pts** a)big problem
- **– 4 pts** b)small modification
- **– 3 pts** b)why the runtime is linear
- **– 1 pt** a)formula not complete
- ✔ **– 2 pts** a)runtime not complete
- **– 1 pt** a)runtime not complete
- **– 1 pt** a)big problem not complete
- **– 1 pt** b)why the runtime is linear not complete
- **– 2 pts** b)small modification not complete
- **– 2 pts** a)recurrence formula wrong
- **– 17 pts** Cheating

Question 7

Designing a Java class

12 / 12 pts

- ✔ **– 0 pts** Correct
- **– 12 pts** Not answered
- **– 2 pts** private variables
- **– 4 pts** constructor
- **– 1.5 pts** setCount
- **– 1.5 pts** increment()
- **– 1.5 pts** getword()
- **– 1.5 pts** getCount()
- **– 2 pts** not complete
- **– 0.5 pts** not complete
- **– 12 pts** Cheating!

Question 8

Java OO principles

8 / 13 pts

- **– 0 pts** Correct
- **– 13 pts** Not answered
- **– 4 pts** part a
- **– 5 pts** part b
- **– 4 pts** part c
- **– 2 pts** part b not complete
- ✔ **– 1 pt** not complete
- ✔ **– 4 pts** part b not complete
- **– 2 pts** part c not complete
- **– 1 pt** part c not complete
- **– 13 pts** Cheating!

Question 9

Interfaces

14 / 15 pts

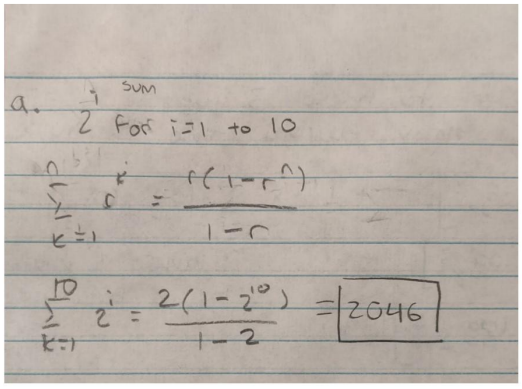
- **– 0 pts** Correct
- **– 15 pts** Not answered
- **– 4 pts** part a
- **– 5 pts** part b
- **– 3 pts** part c
- **– 3 pts** part d
- ✔ **– 1 pt** part a not complete
- **– 1 pt** part b not complete
- **– 1 pt** part c not complete
- **– 2 pts** part c wrong
- **– 2 pts** part d wrong
- **– 2 pts** part a not complete
- **– 3 pts** part b not complete
- **– 1 pt** part d not complete
- **– 15 pts** Cheating!

Question assigned to the following page: 1

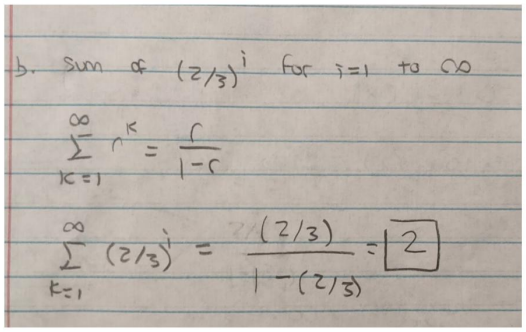
Giancarlos Marte
Homework 1
2/8/21

1. Geometric series

a)



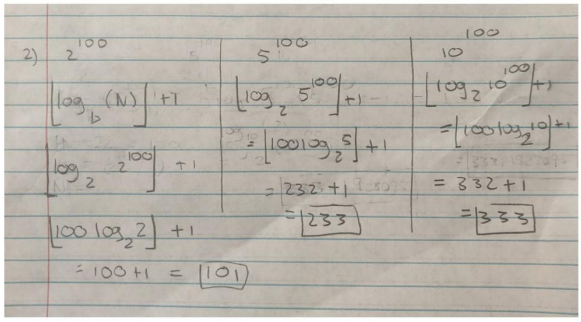
b)

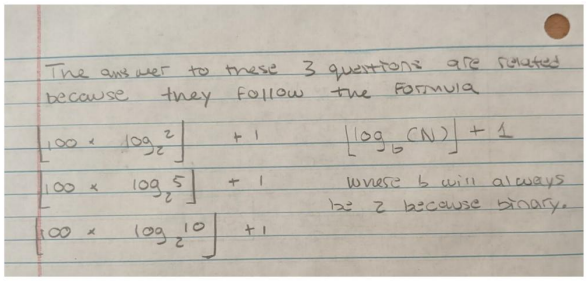


Question assigned to the following page: 2

2. Binary Digits in Numbers

There are 101 digits for 2^100, 233 for 5^100 and 333 for 10^100

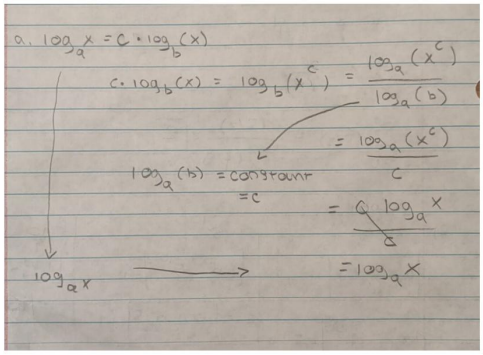




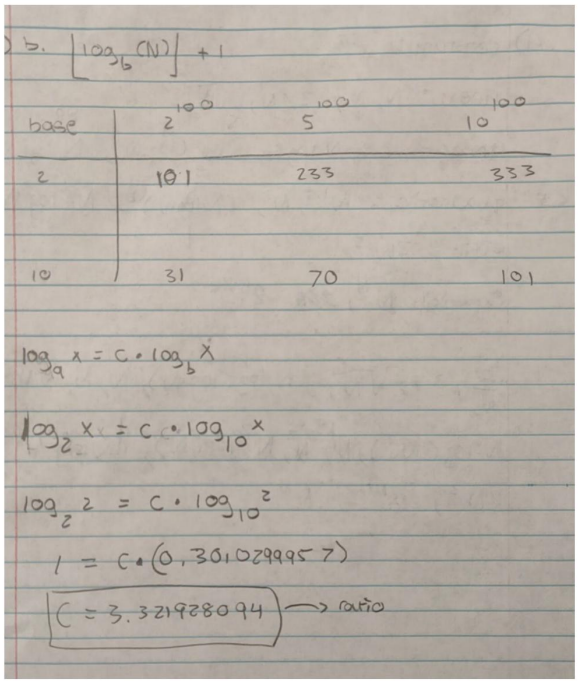
Question assigned to the following page: 3

3. Log functions of different bases

a)



b)

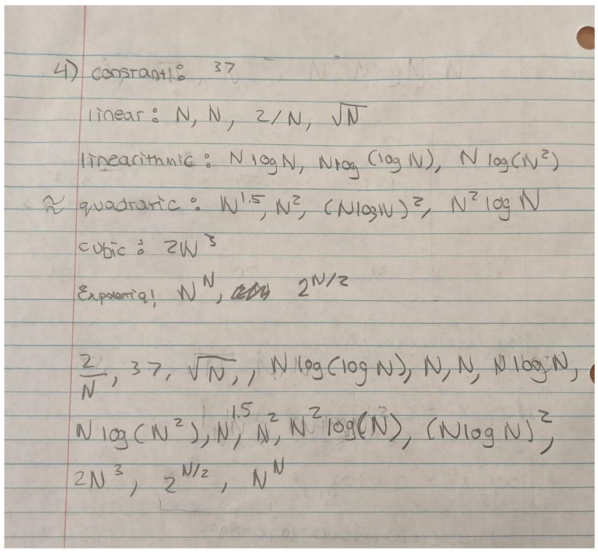


Question assigned to the following page: 4

4. Growth Rates of Functions

2/N, 37, sqrt(N), Nlog(logN), N, N, NlogN, Nlog(N^2), N^1.5, N^2, N^2log(N), (NlogN)^2, 2N^3, 2^(N/2), N^N

The only two with the same growth rates are N and N, which are the same functions.



Question assigned to the following page: 5

5. Big Oh for Non-Recursive Functions

a)

[1] innermost loop variable & operations -> (s += i this is constant)
[2] for loop inside -> (goes from 0 to N - 2 this is O(N))
[3] outer loop -> (goes from 0 to N - 1 this is O(N))
[4] variables -> (constant)

Therefore the big O running time is O(N^2)

b)

[1] innermost loop -> (s += i2 this is constant)
[2] for loop inside -> (goes from 0 to N - 2 this is O(N))
[3] variable & operations inside outer loop -> (constant)
[4] outer loop -> (goes from 0 to N - 1 this is O(N))
[5] variables -> (constant)

Therefore the big O running time is O(N^2), which means it has the same big O analysis as the first version of mysterSum.

c.

```
int mysterySum(int n) {
        int s = 0;
        for (int i = 0; i < n; i++) {
                s += i * i;   // O(1) replacement of inner loop
        }
        return s;
 }
```

d.

Formula: ((n-1) n (2n - 1)) / 6
→ for i^2 series

```
int mysterySum(int n) {
        int s = 0;
        s = ((n - 1) * (2 * n - 1) * n) / 6;
        return s;
}
```

Question assigned to the following page: 6

6. Big Oh for a Recursive Function

a)
O(1) because if statement
T(n-1) + T(n -1) because double recursive calls

T(n) = O(1) + T(n - 1) * T(n - 1) // recursive formula
The runtime is O(N^2)

The problem with this function is that it is doing two recursive calls each time, which causes a lot of redundancy that is not needed.

b)

```
int power2(int n) {
        int a1 = 2;
        if (n == 0) {
                return 1;
        }
        return a1 * power2(n - 1);
}
```

The runtime is linear because there is only one recursive call, while everything else is constant.

Question assigned to the following page: 7

7. Designing and Encapsulated Java Class

```
public class WordUsage {
    private String word;
    private int count;

    public WordUsage(String word, int count) {
        this.word = word;
        this.count = count;
    }
    public WordUsage(String word) {
        this.word = word;
        this.count = 1;
    }

    public String getWord() {
        return word;
    }

    public int getCount() {
        return count;
    }

    public void setCount(int count) {
        this.count = count;
    }

    private void increment() {
        count++;
    }
}
```

Question assigned to the following page: 8

8. Review Java OO Principle

a)
The four most important object methods are equals, getClass, toString and hashCode.

b)

```
boolean checkWords(String s, String t) {
        if (s.length != t.length) {
                return false;
        }
        int l = s.length
        for (int i = 0; i < l; i++) {
                if (s.charAt(i) != t.charAt(i)) {   // if statement that checks charter by character
                        return false;
                }
        }
        return true;
```

This function (comparing s = “abc” & t = “abx”) is checking the content(characters) of the String objects.

If you try to check for equality by doing if (s==t), you are checking if they are the exact same object including their location in memory.

c)

```
String s = "xyz";
Integer eight = 8;
System.out.println(s.hashCode()); // S hashCode = 119193
System.out.println(eight.hashCode()); // 8 hashcode = 8
```

Question assigned to the following page: 9

9. Interfaces

a) An interface in Java is an abstract class that groups together methods and constants. It contains no code implementations for the methods. The purpose of it is to generalize certain code to prevent redundancy across different classes.

b)

```
public interface UnionFind {
    void union(int p, int q);
    int find(int p);
    boolean connected(int p, int q);
    int count();
}
```

```
public class UF implements UnionFind {}
```

c)
The constructor and main method are not in the interface.

d)
Yes, it does qualify for implementing the interface because it uses all of the methods from it, which is a requirement when using an interface.