GRIFFITH COLLEGE DUBLIN

Assignment Cover Sheet

Student name: Ana Trevisan

Student number: 3014953

Faculty: Computing Science

Course: BSc (Hons) Computer Science **Stage/year:** 2nd year

Subject: Relational Databases

Study Mode: Full time Part-time x

Lecturer Name: Bilal Yousuf

Assignment Title: Assignment 2

No. of pages:

Disk included? Yes No (**x**)

Additional Information: (ie. number of pieces submitted, size of assignment, A2, A3 etc)

Date due: 16/03/2021

Date submitted: 16/03/2021

Plagiarism disclaimer:

I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.

I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.

Signed: Ana Trevisan **Date:** 16/03/2021

Please note: *Students* ***MUST*** *retain a hard / soft copy of* ***ALL*** *assignments as well as a receipt issued and signed by a member of Faculty as proof of submission.*

Question 1:
Using the statement execution times defined for HAL, calculate the running time for the given function (Show all steps to get full marks).

static int freq(int f[]){	// 50 (function invocation) + 10 (parameter) = 60
int k = 1; int j = 0;	// 10 + 10 = 20
while(j < f.length){	// 10 * (n + 1) = 10n + 10
if(f[j] * 2 == j)	// (50 + 10 + 10) * n = 70n
k = k * f[j];	// (10 + 10 + 50) * n = 70n
j++;	// 20 * n = 10n
}	// return k; 50
return k;	
}	

Time = 60 + 20 + 10n + 10 + 70n + 70n +20n + 50 = 150 +170n

Question 2:
For the following pseudo codes, find the Big-Oh notation (Show all steps to get full marks).

```
1) Algorithm Factorial(a):
    Input: An integer a
    Output: The value of a factorial (a!)

    Factorial(a)
        factorial <-- 1 //O(1)
        for k=1 to a do // O(n)
            factorial <-- factorial * k //O(1)
        return factorial
    endAlg
```

//TOTAL = O(n)

```
2)Algorithm Power(a, b):
    Input: Two integers a and b
    Output: The value of a to the power b

    Power(a, b)
        power  <-- 1 // O(1)
        for k=1 to b do // O(n)
            power <-- power * a
        return power
    endAlg
```

//TOTAL = O(n)

3) Algorithm LinearSearch(A, n, q):

Input: An integer array A of size n and a query q that we wish to search the array for.
Output: The position of q in A or -1 if q is not in A

```
LinearSearch(A, n, q)
    index <-- 0 //O(1)
    while (index < n) and (A[index] <> q) do //O(n)
        index <-- index + 1
    if (index = n) then
        return -1
    else
        return index
endAlg
```

TOTAL = O(n)

Question 3:

```java
import java.util.Arrays;

public class AnaTrevisan_3014953_Assignment02 {
   public static int factorial(int a) {
       int factorial = 1;
       for(int i = 1; i <=a; i++) {
           factorial = factorial * i;
       }
       return factorial;
   }

   public static int power(int a, int b) {
       int power = 1;
       for(int i = 1; i <= b; i++) {
           power = power * a;
       }
       return power;
   }

   public static int LinearSearch(int[] a, int n, int q) {
       int i = 0;
       while(i<n && a[i] != q){
           i = i+1;
           if (i == n)
               return -1;
       }
       return i;
   }

   public static void main(String[] args) {
       // Factorial test
       int num = 3;
       System.out.println(num + "! = " + factorial(num));
       // Test on the power method
       int a = 2;
       int b = 3;
       System.out.println(a + " ^ " + b + " = " + power(a, b));
       // Test of the Linear Search
       int [] arr = {1, 3, 4, 5, 6};
       int length = arr.length;
       int q = 5;
       System.out.println("Search element " + q + " in array: " + Arrays.toString(arr)
+" ; result: "+LinearSearch(arr, length, q));
       q = 8;
       System.out.println("Search element " + q + " in array: " + Arrays.toString(arr) +
" ; result: "+LinearSearch(arr, length, q));
   }
}
```

Output:

```
3! = 6
2 ^ 3 = 8
Search element 5 in array: [1, 3, 4, 5, 6] ; result: 3
Search element 8 in array: [1, 3, 4, 5, 6] ; result: -1
```