Laboratory practice No. 01
Recursion and complexity laboratory.

Isabella Montoya Henao
Universidad Eafit
Medellín, Colombia
imontoyah@eafit.edu.co

Maria Isabel Arango Palacio
Universidad Eafit
Medellín, Colombia
miarangop@eafit.edu.co

3) Practice for final project defense presentation

3.1 Complexity of the algorithm to calculate the longest common sequence

Worst case :

```
return max((subsecuencia(x[:i-1],y), subsecuencia(x , y[:j-1]))
```

Complexity:

$T(m,n) = c_5 + T(m,n-1) + T(m-1,n)$

Let`s p = n + m , that is $T(p) = c_5 + T(p-1) + T(p-1)$

$Recurrence\ equation\ solution = c_5(2^{p-1}) + c_1(2^{p-1})$

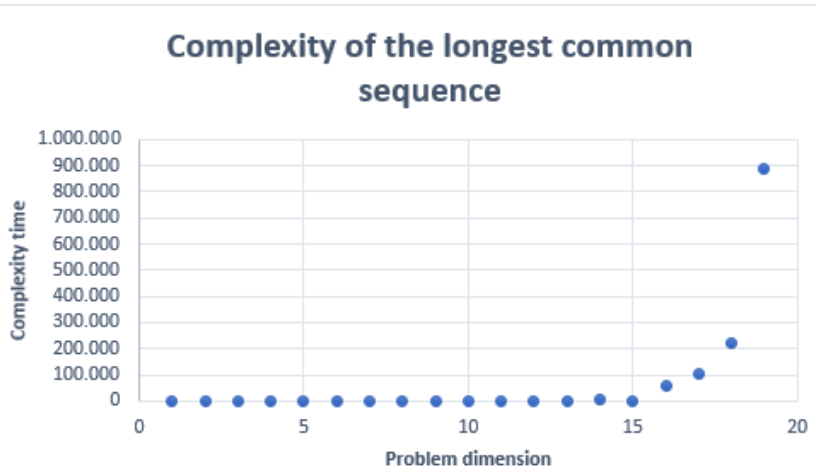
1. T(n) is $O(c_5(2^{p-1}) + c_1(2^{p-1}))$
2. $O(c_5(2^{p-1}) + c_1(2^{p-1})) = O(2^{p-1} + 2^{p-1}) \Rightarrow (Product\ law)$
3. $O(2^{p-1} + 2^{p-1}) = O(2^p + 2^p) \Rightarrow (Sum\ law)$
4. $O(2^p + 2^p) = O(2(2^p)) \Rightarrow (Common\ factor)$
5. $O(2(2^p)) = O(2^p) \Rightarrow (Product\ law)$

T(n) is $O(2^p)$, namely it's a problem with exponential order.

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

3.2 Graphic with the different sizes of the problem and the time

n	T(n)
1	0.0
2	0.0
3	0.0
4	0.0
5	0.0
6	0.0
7	0.00099
8	0.00261
9	0.01321
10	0.05016
11	0.16211
12	0.19725
13	1.607
14	6.959
15	51
16	58.269
17	102.503
18	219.541
19	884.880



3.3 Is the complexity of the exercise 1.1 appropriate to find the longest common sequence between mitochondrial DNA like the datasets ?

The complexity of the DNA'S algorithm is not appropriate to find the longest common sequence between mitochondrial DNA because it has an exponentially complexity in the worst case and according to the Big-O complexity Chart it is horrible to have this type of complexity because as the elements(n) increase the time and the operations will double it. So imagine how much time it will take to find the longest common sequence between two strings with 300.000 characters, it will not be efficient.

3.4 How the exercise GroupSum5 works?

In the exercise groupSum5 you have a variable that is the target and an array. So, you need to sum certain elements of the array so that you can achieve the target. The certain elements that you are going to sum are the multiples of 5 but if after them there is a 1 you can not sum it. You need to implement recursion so, you should do some cases: one to identify if after the multiple of five there is a 1, other is the opposite(if there is no 1s) and

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

finally to identify the multiples and sum them to achieve the target. If the target becomes 0, the function will return true, otherwise it will return false.

3.5 CodingBat

Recursion 1 :

1. BunnyEars

In the exercise BunnyEars you have a number of bunnies and each bunny has two big floppy ears. They ask you to compute the total number of ears across all the bunnies. For that, we will have to implement a recursive function that adds 2 (because of the two big floppy ears) while the number of bunnies decreases in 1 until we reach the base case (when bunnies=0 returns 0).

$$C1 = 3$$
$$T(n) = c_2 + T(n-1)$$
$$Recurrence\ equation\ solution = c_2 n + c$$
$$O(n)$$

n : Number of bunnies

2. Triangle

In this exercise you have a triangle that is made of blocks and each row has a certain number of blocks and they increase consecutively. So you need to use recursion to determine the number of blocks that the triangle has.

$$C_1 = 3$$
$$C_2 = 3$$
$$T(n) = T(n-1) + c_2$$
$$Recurrence\ equation\ solution = c_2 n + c_1$$
$$O(n)$$

n : Number of rows of the triangle

3. Bunny Ears 2

The objective of this exercise is to determine the number of ears that each bunny has. But you need to consider that the even number have different number of ears than the

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

ood bunnies, so you need to do to cases in the code in which you can determine the type of bunny and the ears that i thas

$$C_1 = 3$$
$$C_2 = 6$$
$$T(n) = c_2 + T(n-1)$$
$$Recurrence\ equation\ solution = c_2 n + c$$
$$O(n)$$

n : Number of bunnies

4. SumDigits

You will have a non negative integer and you need to return the sum of all the digits. To do that you need to use %10 that returns the number located further to the right part of the number and the /10 returns the number without the digit located in the right. So, the conclusion is that you need to decompose the number to sum every digit.

$$C_1 = 3$$
$$C_2 = C_1$$
$$T(n) = C_1 + T(n/10)$$
$$Recurrence\ equation\ solution = \frac{c_1 log(n)}{log(10)} + C_1$$
$$O(log(n))$$

n : Integer given

5. Fibonacci

The function will return the number of the Fibonacci sequence in the position n, to get that number we must use recursion until reach the base case, which in this situation are:

1. If is 0, it will return 0
2. If is 1, it will return 1

The other part is the recursive calling, where the function returns the sum of the two previous numbers of the current n.

$$C_1 = 3$$
$$C_2 = 3$$
$$T(n) = C + T(n-1) + T(n-2)$$
$$Recurrence\ equation\ solution = -C + CF_n + CL_n$$
$$O(2^n)$$

n : Integer given

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

Recursion 2 :

1. GroupSum6

In the exercise groupSum6 you have a variable that is the target, another that is the start(it helps you to get a specific position in the array) and an array. So, you need to sum certain elements of the array so that you can achieve the target. The condition is that you must sum all the 6`s in the array given. You need to implement recursion so you will have two cases: when the num in the array is equal to 6 you will include it so your target will decrease to 6, otherwise you call your function and try to make the target equal to 0 subtracting the number of the array from each other. Finally if your target is equal to 0, your function returns true, otherwise it returns false.

Mejor caso :

$C_1 = 5$

$C_2 = 6$

$T(n) = C_2 + T(n-1)$

$Recurrence\ equation\ solution = C * n + C_1$

$O(n)$

Peor Caso:

$C_3 = C_2$

$C_4 = 1$

$T(n) = C_2 + T(n-1) + T(n-1)$

$Recurrence\ equation\ solution = C(2^{n-1}) + C_1 * 2^{n-1}$

$O(2^n)$

n : Number of indices that are missing to finish the array´s go through

2. GroupNoAdj

In the exercise groupNoAdj you have a variable that is the target and an array. So, you need to sum certain elements of the array so that you can achieve the target. The certain elements that you are going to sum have the condition that if a value in the array is chosen to be in the group, the value immediately following it in the array must not be chosen. What we must do is to implement a recursive function that returns true if the target is equal to 0, otherwise to call the function and subtract to the target the element in the array, but always skipping one position. If the target never becomes 0, the function will return false.

$C_1 = 6$

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

$C_2 = 2$
$T(n) = T(n-1) + T(n-2) + C_2$
$Recurrence\ equation\ solution \rightarrow T(n) = -C + C_1 * F_2 + C_2 * L_n$

n : Number of indices that are missing to finish the array´s go through

3. GroupSum5

In the exercise groupSum5 you have a variable that is the target and an array. So, you need to sum certain elements of the array so that you can achieve the target. The certain elements that you are going to sum are the multiples of 5 but if after them there is a 1 you can not sum it. You need to implement recursion so, you should do some cases: one to identify if after the multiple of five there is a 1, other is the opposite(if there is no 1s) and finally to identify the multiples and sum them to achieve the target.

Mejor caso :
$C_1 = 6$
$C_2 = 2$
$C_3 = 12$
$C_4 = 7$
$T(n) = C_3 + T(n-1)$
$Recurrence\ equation\ solution = C * n + C_1$
$O(n)$

Peor Caso:
$C_5 = 6$
$T(n) = C_5 + T(n-1) + T(n-1)$

$Recurrence\ equation\ solution = C\,(2^{n-1}) + C_1 * (2^{n-1})$

$O(2^n)$

n : Number of indices that are missing to finish the array´s go through

4. SplitArray

In first place you should think of an auxiliary method that will receive the arguments to do the split of the array. Here you need to go through all the elements of the array and analyze if you can divide it and have the same result of the sum in each group. But here you cannot use loops, just recursion.

$C_1 = 7$

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

$C_2 = 10$
$T(n) = 2 * T(n-1) + C_2$
$Recurrence\ equation\ solution = C_2(2^{n-1}) + C_1 * 2^{n-1}$
$O(2^n)$

n : Number of indices that are missing to finish the array´s go through

5. GroupSumClump

In the exercise groupSumClump you have a variable that is the target and an array. So, you need to sum certain elements of the array so that you can achieve the target. But here you have a simple restriction and it is the following: if there are more than one equal element and consecutive you need to sum then all or not sum them at all. So you need to analyze in the code if adding them all will achieve the target or not.

Mejor caso :
$C_1 = 6$
$C_2 = 10$
$C_3 = 8$
$C_4 = 16$
$C_5 = 6$
$T(n) = T(n-1) + T(n-1) + C_3$
$Recurrence\ equation\ solution = C_3(2^{n-1}) + C_1(2^{n-1})$
$O(2^n)$

Peor Caso:
$C_4 = 16$
$C_5 = 6$
$T(n) = T(n-2) + T(n-2) + C_4$
$Recurrence\ equation\ solution = 2^{n/2}(C_2(-1)^n + C_1) - C_4$
$O(2^{n/2})$

n : Number of indices that are missing to finish the array´s go through

4) Practice for midterms

4.1 4.1.1. Line 6 : A
4.1.2. Line 9 : C

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4.1.3. Line 11 : A

4.2 1) 4.2.1. Line 9: floodFillUtil(screen, x+1, y+1, prevC, newC, N, M)
4.2.2. Line 10: floodFillUtil(screen, x-1, y+1, prevC, newC, N, M)

2) 4.2.3. Line 11: floodFillUtil(screen, x+1, y-1, prevC, newC, N, M)
4.2.4. Line 12: floodFillUtil(screen, x-1, y-1, prevC, newC, N, M)

3) T(p) = T(p-2) + T(p-2) + T(p-2) + T(p-2) + T(p-1) + T(p-1) + T(p-1) + T(p-1) + C
T(p) = 4T(p-2) + 4T(p-1) + C

$Recurrence\ equation\ solution$T(p) = $\frac{-C}{7} + C_1(2 - 2\sqrt{2})^p + C_2(2(1 + \sqrt{2})^p$

O(8^n)

4.3 *B :* $T(n,m) = C * n * m^2$

4.4
```
int Lucas(n){
  if(n==0)
        return 2;
  if(n==1)
        return 1;
  return Lucas(n-1) + Lucas(n-2)
}
```

4.4.1 C

The complexity of Lucas` algorithm is: *T(n) = T(n-1) + T(n-2) + C,* *que es* $O(2^n)$

4.5 *1. A*
2. B

```
public static boolean isPal(String s){
      if(s.length()==0 || s.length()==1){
          return true;
    }
      if(s.charAt(0)== (s.charAt(s.length()-1))){
          return isPal(s.substring(1, s.length()-1));
    } else {
          return false;
    }
}
```

4.6 *A*

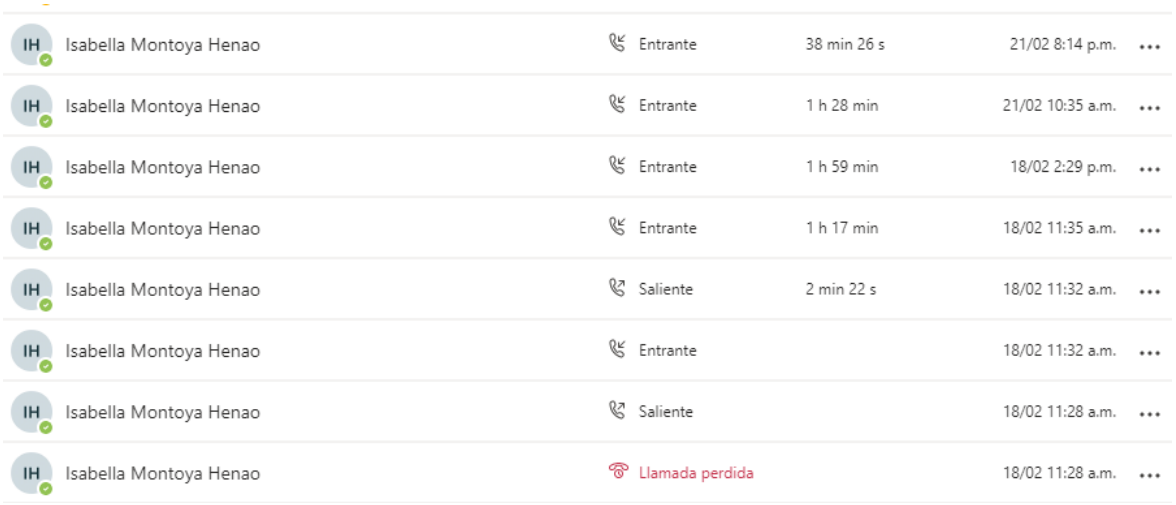
4.7 Return *sumaAux (n,i+2)*
Return *sumaAux (n, i+1)*

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4.8
4.9 B : 6

6) Teamwork and gradual progress

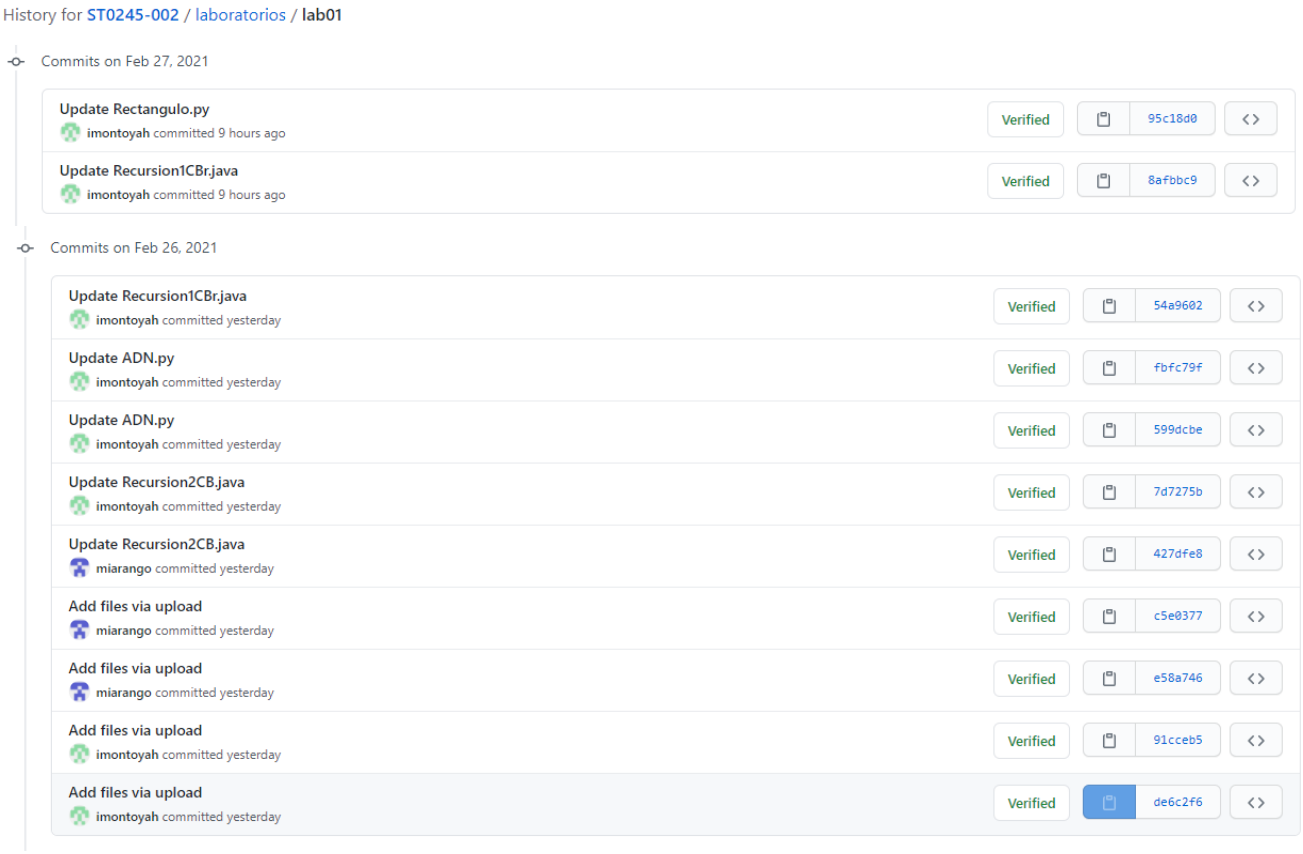
6.1 Meeting minutes



IH	Isabella Montoya Henao	Entrante	38 min 26 s	21/02 8:14 p.m.
IH	Isabella Montoya Henao	Entrante	1 h 28 min	21/02 10:35 a.m.
IH	Isabella Montoya Henao	Entrante	1 h 59 min	18/02 2:29 p.m.
IH	Isabella Montoya Henao	Entrante	1 h 17 min	18/02 11:35 a.m.
IH	Isabella Montoya Henao	Saliente	2 min 22 s	18/02 11:32 a.m.
IH	Isabella Montoya Henao	Entrante		18/02 11:32 a.m.
IH	Isabella Montoya Henao	Saliente		18/02 11:28 a.m.
IH	Isabella Montoya Henao	Llamada perdida		18/02 11:28 a.m.

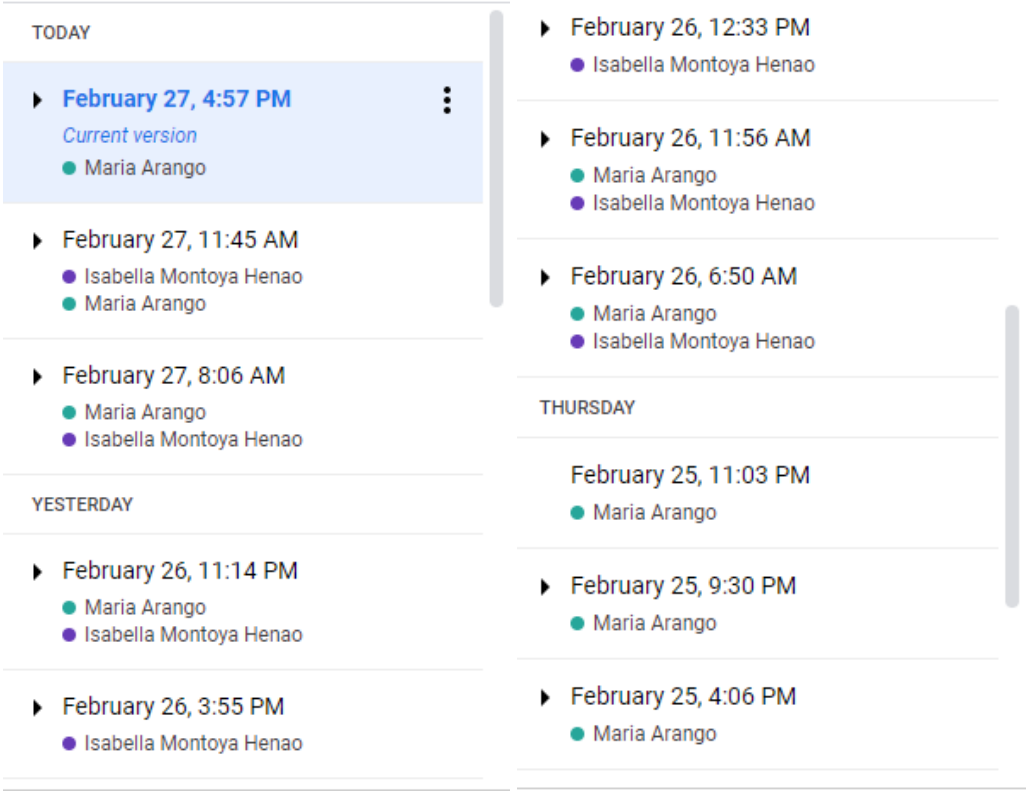
6.2 History of changes of the code

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473



PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

6.3 History of changes of the report



PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473