ESTRUCTURA DE DATOS 1
Código ST0245

Laboratory practice No. 1: Recursión

Martín Ospina Uribe
Universidad Eafit
Medellín, Colombia
mospinau1@eafit.edu.co

María José Bernal Vélez
Universidad Eafit
Medellín, Colombia
mjbernalv@eafit.edu.co

1) Mock project

1.1) ADN subsequence

This algorithm receives as parameters two DNA sequences and compares them to find the size of the longest common subsequence in the same relative order. To achieve this, we used a recursive method, which has a base case when either the first or the second ADN sequence has a length of 0. Additionally, when it finds two equal characters in the sequence, it adds 1 to the return value, and if it doesn't, it continues searching for similarities.

1.2) Ways

This algorithm finds all the possible ways in which 2x1cm^2 rectangles can be organized in a container of 2xncm^2 (with *n* the length of the container which is received as a parameter). Our algorithm has a base case when n is less or equal to 2, and returns the value of n, which is the possible organization. If the number is higher, it uses recursion to call itself again by subtracting 1 and 2 to the value of n (organizing the rectangles either horizontally or vertically), and adding those results.

2) Online exercises

2.1) Recursion 1

Bunny ears: this algorithm uses recursion to find the amount of ears that a given value of bunnies have, which is received as a parameter. To do this, it has a base case when the amount of bunnies is 0, which returns 0, and if it is greater than 0, it returns 2 plus the number of ears that one less bunny has (recursive call: bunnies-1).

Fibonacci: This algorithm uses recursion to generate the number in the n position (which is received as a parameter) of the Fibonacci sequence. In order to do this, it has a base case when the n is 0 and when n is 1, which returns 0 and 1 respectively, and if it's greater than 1 it returns the sum of the two previous terms calling itself (recursive call: fibonacci(n-1)+fibonacci(n-2)).

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

Inspira Crea Transforma   Vigilada Mineducación   www.eafit.edu.co

ESTRUCTURA DE DATOS 1
Código ST0245

Bunny ears 2: in this algorithm, a number of bunnies is received as a parameter, and it counts the amount of ears that those bunnies have, but with the additional condition that an even number of bunnies have 3 ears because they have a raised foot and an odd number of bunnies have 2 ears. This algorithm works the same way as "Bunny ears", but the recursive part has a conditional that checks if the number of bunnies is even or odd.

Triangle: This algorithm receives a number as a parameter that refers to the number of rows in a triangle (made out with blocks) and it counts the number of blocks that the triangle has for that number of rows (each row has the number of blocks that corresponds to its index). The base case for this algorithm is when the number of rows is 0, which returns 0, and if it's greater than 0 it returns the total amount of blocks in the triangle (blocks in that row + blocks in previous rows).

Sum digits: this method returns the sum of the digits of a given non negative integer recursively. Its base case is when the number is less or equal to 0, which returns 0. However, if the number is greater than 0, it returns its module by 10 (returning the rightmost digit), plus the recursive method that divides the number by 10 (which removes the rightmost digit).

2.2) Recursion 2

Group sum 6: This algorithm receives an array of integers and determines if it is possible to choose a group of those integers such that it sums a given target, with the additional condition that all 6`s must be chosen. Its base case is when the position that is being analyzed (start) is greater than or equal to the length of the array, which returns either true or false if the condition is matched. Else, it calls itself again, either using the number analized (if it's 6 it will always be used) or not using it (in both cases we add 1 to the start value).

Group no adj: this algorithm receives an array of integers and determines if it is possible to choose a group of those integers such that it sums a given target, with the additional limitation that if a number is chosen, the one immediately after cannot be chosen. Its base case is when the position that is being analyzed (start) is greater than or equal to the length of the array, which returns either true or false if the condition is matched. Else, it calls itself again, either using the integer in the current position (adding 2 to the start value) or not using it (adding only 1 to the start position).

Group sum 5: This algorithm receives an array of integers and determines if it is possible to choose a group of those integers such that it sums a given target, with the additional condition that all multiples of 5 in the array must be included and if the value immediately following a multiple of 5 is 1, it must not be used.   Its base case is when
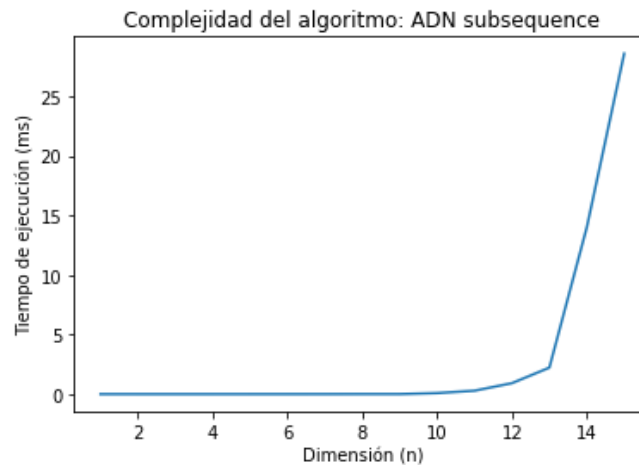
PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

UNIVERSIDAD EAFIT ®

Acreditación Institucional
Renovación
2 0 1 8 - 2 0 2 6
Resolución MEN 2158 de 2018

the position that is being analyzed (start) is greater than or equal to the length of the array, which returns either true or false if the condition is matched. Else it calls itself again, either using the number analized (if it's a multiple of 5 it will always be used) or not using it (if the number is 1 and the previous number was a multiple of five the number will never be used) in both cases we add 1 to the start value.

Group sum clump: this algorithm receives an array of integers and determines if it is possible to choose a group of those integers such that it sums a given target, with the additional limitation that all adjacent and identical values can either all be chosen or not. Its base case is when the position that is being analyzed (start) is greater than or equal to the length of the array, which returns either true or false if the condition is matched. Else, it calls itself again, either using all the adjacent and equivalent integers or not, by counting how many there are and modifying the start position.

Split array: this algorithm receives an array of integers, and it determines if it is possible to divide it in two subgroups so that the sums of the elements on the subgroups are the same. To achieve this, it uses an auxiliary method that has a base case when the start position is greater or equal to the length of the array, which returns either true or false if the condition is matched or not. Finally, on the recursive portion, it analyzes both possible cases, which are to add the given element to the first subgroup or the second one.

3) Practice for final project defense presentation

3.1 Complexity of worst case of exercise 1.1.
- **Recurrence equation:**

$$T(n, m) = T(n - 1, m) + T(n, m - 1) + c$$
$$with\ p = n + m,\ T(p) = 2T(p - 1) + c$$
$$T(n) = c * (2^p - 1) + c * 2^{p-1}$$

- **Big O notation:**

$$O(2^p)$$

3.2 Experimental time complexity
The following graph shows the times taken by the ADN subsequence algorithm to run in 15 different sizes of the problem.

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

UNIVERSIDAD
EAFIT ®

A Acreditación
Institucional
Renovación
2 0 1 8 - 2 0 2 6
Resolución MEN 2158 de 2018

Inspira Crea Transforma

Vigilada Mineducación   **www.eafit.edu.co**

ESTRUCTURA DE DATOS 1
Código ST0245

The results match the complexity found theoretically, as it portrays an exponential complexity. Due to this, it is nearly impossible to calculate the time that it will take to find the longest subsequence between two mitochondrial ADNs (that have approximately 300,000 characters each), as its complexity would be $O(2^{300,000+300,000}) = O(2^{600,000})$.

3.3 Is the complexity of the algorithm in exercise 1.1 appropriate to find the longest common subsequence between mitochondrial ADNs such as the ones in the datasets?

No, because this algorithm has an exponential complexity, which will not work with long ADN sequences such as the ones in the datasets.

3.4 Explain with your own words how groupSum5 works.

In point 2.2 of this report the functionality of groupSum5 is explained.

3.5/3.6 Complexity of online exercises and meaning of variables.

Bunny ears:
- **Recurrence equation:**

$T(n) = T(n-1) + c$
$T(n) = c * n + c_1$

- **Big O notation:**

$O(n)$

- **n =** number of bunnies

Fibonacci:
- **Recurrence equation:**

$T(n) = T(n-1) + T(n-2) + c$
$T(n) = -c + c_1 F_n + c_2 L_n$ (F_n is the nth fibonacci number and L_n is the nth Lucas number)

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

UNIVERSIDAD EAFIT ®
Acreditación Institucional
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

Inspira Crea Transforma

Vigilada Mineducación   www.eafit.edu.co

- **Big O notation:**

$O(2^n)$

- **n =** position of the Fibonacci number that is being looked for

Bunny ears 2:
- **Recurrence equation:**

$T(n) = T(n-1) + c$
$T(n) = c * n + c_1$

- **Big O notation:**

$O(n)$

- **n =** number of bunnies

Triangle:
- **Recurrence equation:**

$T(n) = T(n-1) + c$
$T(n) = c * n + c_1$

- **Big O notation:**

$O(n)$

- **n =** number of rows of the triangle

Sum digits:
- **Recurrence equation:**

$T(n) = T(n/10) + c$
$T(n) = c * log_{10}(n) + c_1$

- **Big O notation:**

$O(log_{10}(n))$

- **n =** number which digits are going to be added

Group sum 6:
- **Recurrence equation:**

$T(n) = c_7 + T(n-1) + c_8 + T(n-1) + c_9$

$T(n) = 2T(n-1) + c$
$T(n) = c * (2^n - 1) + c_1 * 2^{n-1}$

- **Big O notation:**

$O(2^n)$

- **n =** position of the element of the array that is being added

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

UNIVERSIDAD EAFIT®

Acreditación Institucional
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

Inspira Crea Transforma

Vigilada Mineducación   www.eafit.edu.co

ESTRUCTURA DE DATOS 1
Código ST0245

Group no adj:
- **Recurrence equation:**

$T(n) = c_5 + T(n - 2) + c_6 + T(n - 1) + c_7 + c_8$

$T(n) = T(n - 2) + T(n - 1) + c$

- **Big O notation:**

$O(2^n)$

- **n =** position of the element of the array that is being added

Group sum 5:
- **Recurrence equation:**

$T(n) = c_{10} + T(n - 1) + c_{11} + T(n - 1) + c_{12}$

$T(n) = 2T(n - 1) + c$

$T(n) = c * (2^n - 1) + c_1 * 2^{n-1}$

- **Big O notation:**

$O(2^n)$

- **n =** position of the element of the array that is being added

Group sum clump:
- **Recurrence equation:**

$T(n) = 2 * n * c_{14} + 2T(n - count) + c_{15}$

- **Big O notation:**

$O(2^{n/2})$

- **n =** position of the element of the array that is being added

Split array:
- **Recurrence equation:**

$T(n) = T(n - 1) + c_9 + T(n - 1) + c_{10}$

$T(n) = 2T(n - 1) + c$

$T(n) = c * (2^n - 1) + c_1 * 2^{n-1}$

- **Big O notation:**

$O(2^n)$

- **n =** position of the element of the array that is being added to either group 1 or 2.

4) Practice for midterms
 4.1 1. a
  2. c
  3. a

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

UNIVERSIDAD EAFIT®

Acreditación Institucional
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

Inspira Crea Transforma

Vigilada Mineducación **www.eafit.edu.co**

4.2 1. <u>Line 9:</u> floodFillUtil(screen, x+1, y+1, prevC, newC, N, M)
   <u>Line 10:</u> floodFillUtil(screen, x-1, y-1, prevC, newC, N, M)
2. <u>Line 11:</u> floodFillUtil(screen, x+1, y-1, prevC, newC, N, M)
   <u>Line 12:</u> floodFillUtil(screen, x-1, y+1, prevC, newC, N, M)
3. $T(p) = 4T(p-1) + 4T(p-2) + c$

$$T(p) = -\frac{c}{7} + c_1(2 - 2\sqrt{2})^p + c_2(2(1 + \sqrt{2}))^p$$

Big O notation: $O(8^n)$

4.3 b

4.4 4.4.1. c

4.5 1. a
2. b

6) Teamwork and gradual progress (optional)

Member	Date	Done	Doing	To do
Martín and Maria José	17/02/2021	Codingbat Recursion 1 exercices Exercise 4.1	Codingbat Recursion 2 exercises	Explain Codingbat Recursion 1 codes
Martín and Maria José	18/02/2021	Codingbat Recursion 2	Think about exercise 4.2	Explain Codingbat Recursion 2 codes
Martín and Maria José	19/02/2021	Plan next week's meetings		
Martín and Maria José	24/02/2021	Exercises 1.1 and 1.2 (mock project) Finish midterm exercises (#4)	Explain codes from point 1 and 2	Practice for final project defense presentation
Martín and	25/02/2021	Find complexities in	Check answers	Work on 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

UNIVERSIDAD **EAFIT**® **A** Acreditación Institucional Renovación 2018 - 2026 Resolución MEN 2158 de 2018

ESTRUCTURA DE DATOS 1
Código ST0245

Maria José		exercise 3.	to exercises.	report.
Martín and María José	28/02/2021	Finish the report and upload everything to GitHub.		

6.1 Meeting minutes

Name	Type	Duration	Date
MU Martin Ospina Uribe	↗ Outgoing	2h 28m	6:28 PM ...
MU Martin Ospina Uribe	↗ Outgoing	9s	6:26 PM ...
MU Martin Ospina Uribe	☎ Missed call		6:25 PM ...
MU Martin Ospina Uribe	↗ Outgoing	7s	6:25 PM ...
MU Martin Ospina Uribe	↗ Outgoing	1h 25m	2/25 6:37 PM ...
MU Martin Ospina Uribe	↙ Incoming	45s	2/25 6:35 PM ...
MU Martin Ospina Uribe	☎ Missed call		2/25 6:35 PM ...
MU Martin Ospina Uribe	↗ Outgoing	30m 58s	2/25 6:03 PM ...
MU Martin Ospina Uribe	↗ Outgoing	2h 14m	2/24 5:24 PM ...
MU Martin Ospina Uribe	↗ Outgoing	44m 51s	2/18 8:20 PM ...
MU Martin Ospina Uribe	↗ Outgoing	1h 58m	2/18 6:22 PM ...
MU Martin Ospina Uribe	↗ Outgoing	2h 9m	2/17 8:02 PM ...

6.2 History of changes of the code

Commits on Feb 28, 2021

Versión final
mjbernalv committed 1 minute ago   Verified   90d529d <>

Versión final
mjbernalv committed 25 minutes ago   Verified   e2a59f7 <>

Versión final
martinospina committed 3 hours ago   Verified   57103dd <>

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

UNIVERSIDAD EAFIT®   Acreditación Institucional Renovación 2018 - 2026 Resolución MEN 2158 de 2018

6.3 History of changes of the report

▶ **February 28, 9:27 PM**   ⋮
Current version
● Maria José Bernal
● Martin Ospina

WEDNESDAY

▶ February 24, 7:34 PM
● Maria José Bernal

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