ESTRUCTURA DE DATOS 1
Código ST0245

Laboratory practice No. 3: ArrayList and LinkedList

Nayibe Tatiana Vélez David
Universidad Eafit
Medellín, Colombia
ntvelezd@eafit.edu.co

Alejandro Fernández Restrepo
Universidad Eafit
Medellín, Colombia
afernader@eafit.edu.co

Adelaida Maldonado Esguerra
Universidad Eafit
Medellín, Colombia
amaldonade@eafit.edu.co

3) Practice for final project defense presentation

3.1 In this case, data structures are equally efficient in their implementation to solve the problem, because their effectiveness is not altered.

ARRAYLIST	LINKEDLIST
O(n)	O(n)

3.2 EXPLANATION EXERCISE 2.1

This exercise consisted of simulating a broken keyboard; Its main objective was to give a word order through character search: búsqueda [,] ’.
What we mainly did was to take each line entered through the user by keyboard, work it as a list in order to perform the various operations, which were essentially send at the beginning of the line and at the end of it through the identification of The mentioned characters.

What we mainly do is convert what is entered into a chain in order to analyze character by character and thus discover the square brackets that you want to find; After finding them, we compare the position found and if it is different from the brackets searched, they are sent at the end or at the beginning of the list, using the addLast and add methods; After performing this operation, what is done by removing the square brackets through the remove method through an auxiliary method, which is what allowed the position found to be removed; In this way we conclude by printing the updated list, which will be the one that will show the ordered list without brackets on it.

EXPLANATION EXERCISE 2.1

The problem is to interpret a series of commands that instruct an arm
robotic on how to manipulate blocks that are on a flat table. Initially
there are n blocks on the table (listed from 0 to n-1) with the bi block adjacent to the block bi + 1 for all integers i with i <n-1.

For this problem we solve it as follows:
First a stack is created with an amount n that the user enters in such a way that the numbers from 0 to n are added in an orderly manner.
Then there is a method which, given a String that the user enters, separates this string with a Split by spaces to analyze if it is a command for the robot, first you see if the first word is move or pile, if it is move it evaluate if the third word is onto or over, if it is pile, evaluate if the third word is onto or over in the same way, in any of the 4 cases the number is taken in position 2 and in position 4, the first It is the block to be moved or stacked and the second is where it will move or stacked, or if the user enters the word that the program ends.

When the command is already to be done and the blocks to be moved and in which direction they are going to move, a method is called that performs the command that the user has written. The first is move onto where a and b are block numbers, put the block above block b after returning any block that is stacked on the blocks a and b to their initial positions, this we did with two while and a temporary stack to keep the data , the first while goes through all the positions of the arrangement looking for which stack is the block to be moved, when it finds it it is in the stack of the stack returning the remaining blocks to their initial position. In the second while, it looks for where the block b is and when it is found, it returns all the blocks above it to its initial position, and in the end it removes block a from its stack and places it on top of block b.

The second move over does the same as the move onto only that it only has the first while (described above) since it is not necessary to remove the blocks that are above the block b and simply place it above the stack of b.

The third pile onto where a and b are block numbers, moves the stack of blocks that
It consists of block a and all blocks stacked on it, above b. All the blocks above block b are moved to their initial position before it is stacked. The blocks stacked on the block retain their original order after being moved. We do this with a while looking for the stack where the block is and the blocks that are on it and we store them in a temporary stack, then in another while we look for the block by and return all the blocks on it to their initial position, at the end we put the temporary stack in the order that was on block b.

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

The last pile over is done the same as n pile onto only that the blocks are not flown over b to their initial positions and the temporary pile is simply placed on top of the pile where block b is located.

3.3 Complexity exercise 2.1

This exercise has a complexity: O(n^2)

3.4 Explanation of variables exercise 2.1

- s = string to analyze
- i= variable with which the route will be made
- j = variable with which it will be identified
- s= chain size

4) Practice for midterms

4.1 *c*
4.2 *c*
4.3 Line 2: q.size()>1
Line 3: <=
Line 4: q.remove()
Line 6: q.remove()

4.4 Line 3: lista.size()
Line 4: lista.addLast(auxiliar.pop())

4.5 Line 12: auxiliar1.size() >0
Line 16: auxiliar2.size()>0
Line 18: personas.offer(edad)

4.6 A) O(n^3)

4.7 C) O(n)

4.8 4.8.1= a) O(k)
4.8.2= c)12
4.8.3= c) O(1)

4.9 4.9.1= d) O(n)
4.9.2= a) 6
4.9.3= b) O(n)

4.10 4.10.1=b) O(max(list)*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

4.10.2 =b) O(n)

4.11 4.11.1= Line13= s1.size()>1
4.11.2= Line14=s1.pop()
4.11.3=Line 17=s2

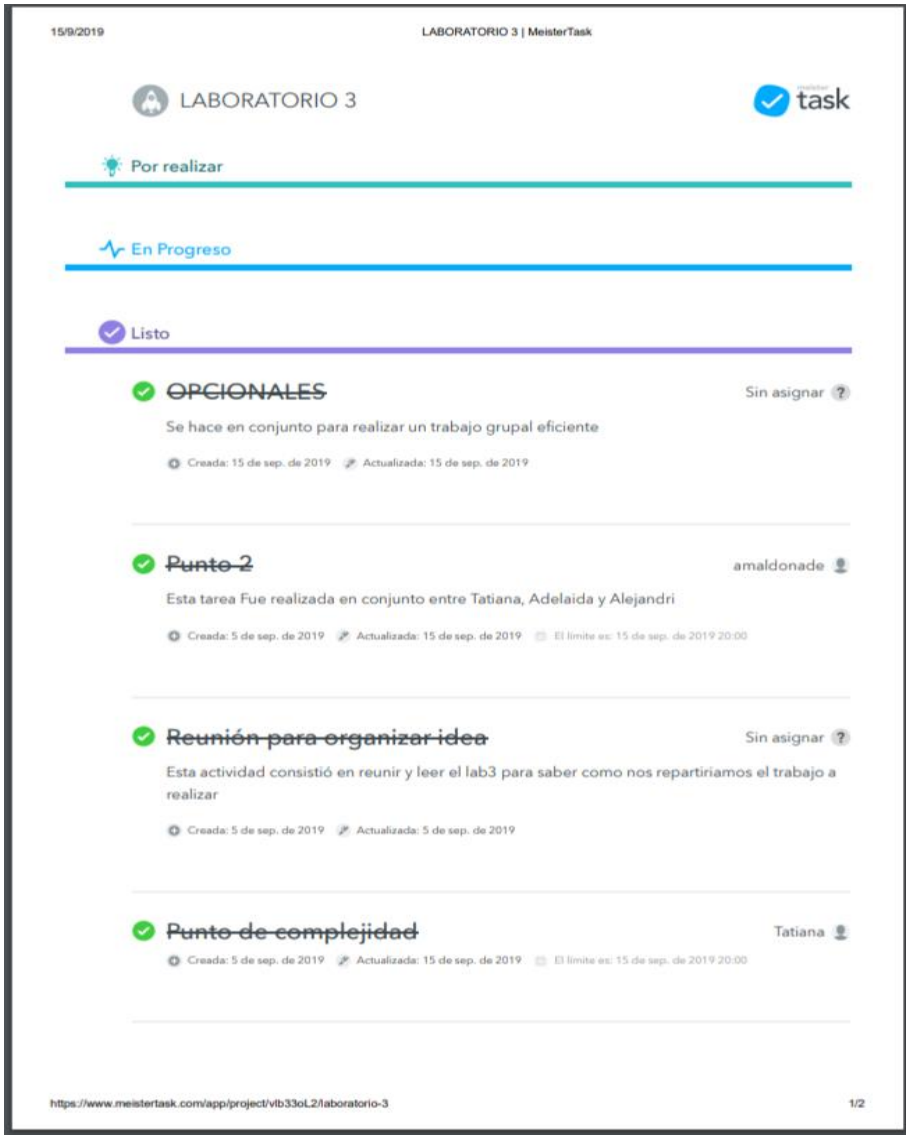
4.12 4.12.1= 4) 0,2,4,6,8,10
4.12.2= 2) O(n)

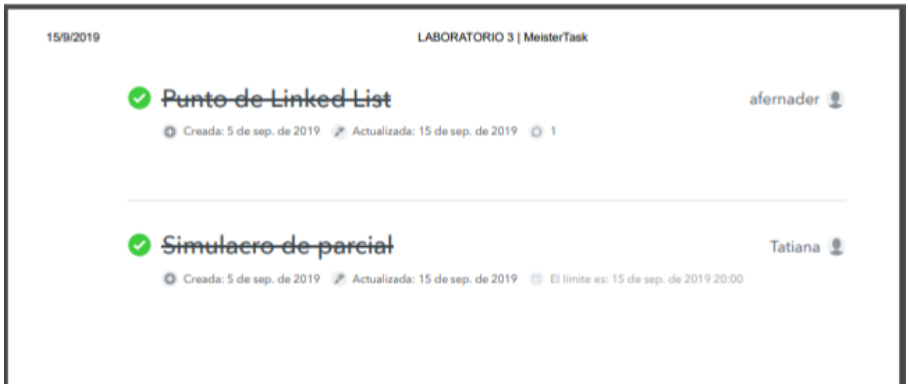
4.13 4.13.1= 1) O(n)
4.13.2= 1) O(n)

4.14 4) 5,4,3,2

OPTIONAL EXERCISES

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