Laboratorio Nro. 3:
Implementación de Listas Enlazadas (Linked List)

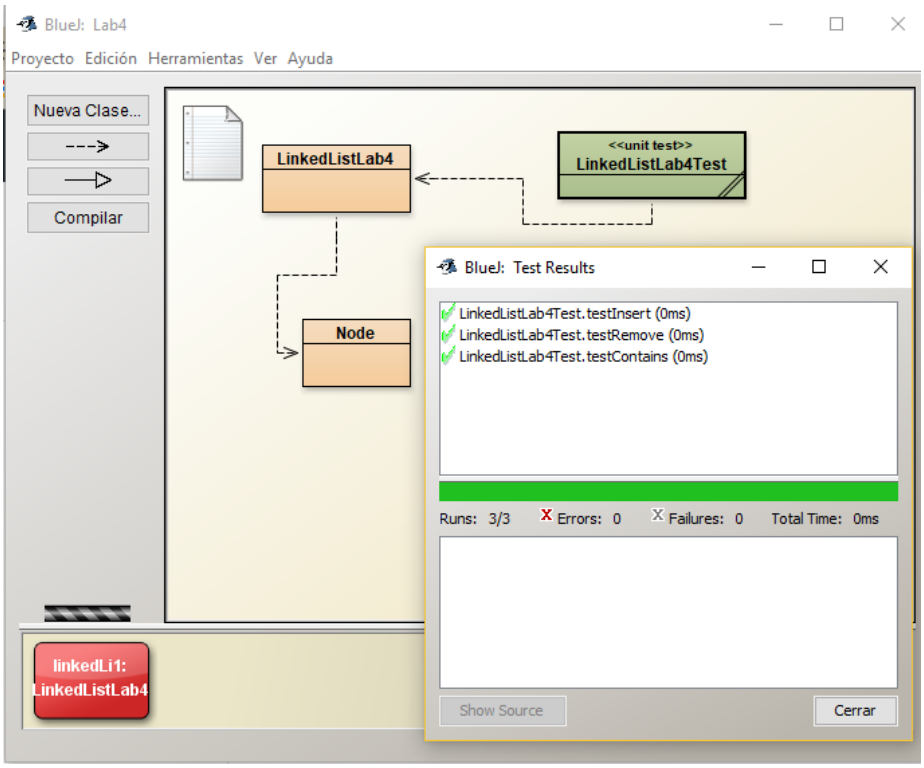
Santiago Escobar Mejía
Universidad Eafit
Medellín, Colombia
sescobarm@eafit.edu.co

Sebastián Giraldo Gómez
Universidad Eafit
Medellín, Colombia
sgiraldog@eafit.edu.co

Luisa María Vásquez
Universidad Eafit
Medellín, Colombia
lmvasquez@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1) Teniendo en cuenta lo anterior, verifiquen, utilizando *JUnit*, que todos los *tests* escritos en el numeral 1.2 pasan. Muestren, en su informe de PDF, que los tests se pasan correctamente; por ejemplo, incluyendo una imagen de los resultados de los *tests*.



DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

3.2 , 3.3 y 3.4)

MoveAontoB:

- **Parte 1**: Este método, realiza una búsqueda en el arreglo de stacks, hasta que en uno de ellos encuentra el valor de a y b, almacena la posición del arreglo donde está el stack que contiene el bloque a y b.

  Complejidad O=2*n*m donde n es el tamaño del arreglo y m es la cantidad de elementos sobre el bloque a y b, ya que el método search busca desde el tope del stack

- **Parte 2:** Luego, reorganiza a su posición original los valores que estén sobre el bloque a y el bloque b, por medio de el valor de arreglos relacionados con los stacks en el mismo orden de sus valores.

  Complejidad O=p+q donde p es el numero de bloques sobre el bloque a y q el numero de bloques sobre el bloque b

- **Parte 3:** Por ultimo hace el movimiento del bloque a hacia el stack donde está el bloque b justo encima de el.

  Complejidad O=1

  Complejidad total del método O= (2*n*m)+p+q+C

MoveAoverB:

- **Parte 1:** Este método, realiza una búsqueda en el arreglo de stacks, hasta que en uno de ellos encuentra el valor de a y b, almacena la posición del arreglo donde está el stack que contiene el bloque a y b.

  Complejidad O=2*n*m n es el tamaño del arreglo y m es la cantidad de elementos sobre el bloque a y b, ya que el método search busca desde el tope del stack

- **Parte 2:** Luego, reorganiza a su posición original los valores que estén sobre el bloque a, por medio de el valor de arreglos relacionados con los stacks en el mismo orden de sus valores.

  Complejidad O=p donde p es el numero de bloques sobre el bloque a

- **Parte 3:** Por ultimo hace el movimiento del bloque a hacia el stack donde está el bloque b justo encima de el.

  Complejidad O=1

  Complejidad total del método O= (2*n*m)+p+C

PileAontoB:

- **Parte 1:** Este método, realiza una búsqueda en el arreglo de stacks, hasta que en uno de ellos encuentra el valor de a y b, almacena la posición del arreglo donde está el stack que contiene el bloque a y b.

  Complejidad O=2*n*m n es el tamaño del arreglo y m es la cantidad de elementos sobre el bloque a y b, ya que el método search busca desde el tope del stack

- **Parte 2:** Luego, reorganiza a su posición original los valores que estén sobre el bloque b, por medio de el valor de arreglos relacionados con los stacks en el mismo orden de sus valores.

  Complejidad O=q donde q es el numero de bloques sobre el bloque b

- **Parte 3:** Mueve todo el stack sobre el boque a hacia un stack en el que queda en el orden contrario

  Complejidad O=p donde p es el numero de bloques sobre a

- **Parte 4:** Mueve el bloque a al Stack auxiliar ya con la seguridad de que esta en el tope del stack
  Complejidad O=1

- **Parte 5:** Mueve el stack de auxiliar a sobre el bloque b, y lo regresa a su orden original de tal forma de que el bloque a fue el primer bloque ingresado y quede justo sobre el bloque b

  Complejidad O=r donde r es el tamaño del stack auxiliar

  Complejidad del método O=(2*n*m)+q+p+r+C

PileAoverB:

- **Parte 1:** Este método, realiza una búsqueda en el arreglo de stacks, hasta que en uno de ellos encuentra el valor de a y b, almacena la posición del arreglo donde está el stack que contiene el bloque a y b.

  Complejidad O=2*n*m n es el tamaño del arreglo y m es la cantidad de elementos sobre el bloque a y b, ya que el método search busca desde el tope del stack

- **Parte 2:** Mueve todo el stack sobre el boque a hacia un stack en el que queda en el orden contrario

  Complejidad O=p donde p es el numero de bloques sobre a

- **Parte 3:** Mueve el bloque a al Stack auxiliar ya con la seguridad de que esta en el tope del stack

  Complejidad O=1

- **Parte 4:** Mueve el stack de auxiliar a la pila donde está el bloque b y lo regresa a su orden original, de tal forma de que el bloque a fue el primer bloque ingresado

  Complejidad O=r donde r es el tamaño del stack auxiliar

  Complejidad del método O=(2*n*m)+p+r+C

ImprimirBloques:

- Imprime el nombre del stack y llama al método que concatena el stack como un string en el orden adecuado y imprime cada stack del arreglo de la misma manera

  Complejidad total del método O= n*(2*m)+C donde n es el tamaño del arreglo y 2*m es la complejidad de el método stackImp

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

StackImp:

- Concatena el stack señalado en el arreglo por medio del parámetro que recibe como un string y lo retorna como string, para eso para el Stack a un auxiliar y luego lo concatena para que así quede en el orden que tenia originalmente

  Complejidad total del método O=n+m+C donde n es el tamaño del stack y m también por lo que al ser el mismo valor es igual a O=2n+C

4) Simulacro de Parcial

1) De acuerdo a lo anterior, responda las siguientes preguntas:

a) ¿Qué condición colocaría en el ciclo while de la línea 3? (10%)
R//= lista.size() >0

b) Complete la línea 7 de forma que el algoritmo tenga sentido (10%)
R//= lista.add(auxiliar.pop());

2)

a) ¿Que condiciones colocaría en los 2 ciclos while de líneas 12 y 16, respectivamente?

Línea 12: !auxiliar1.isEmpty()
Línea 16: !auxiliar2.isEmpty()

b) Complete la línea 18 con el objeto y el llamado a un método, de forma que el algoritmo tenga sentido

Línea 18: personas.offer(edad);

3) Cuál es la complejidad asintótica, para el peor de los casos, de la función procesarCola(q, n)?

R//= c) O(n^2)

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co