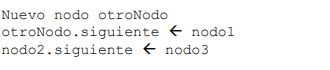
1)



Inserta “otroNodo” en la lista, quedando como anterior a nodo1.

2)

Texto

Descripción generada automáticamente

Elimina nodo2 de la lista.

3)

Texto

Descripción generada automáticamente con confianza media

Inserta “otroNodo” en la lista, quedando entre nodo1 y nodo2.

4)

Texto, Carta

Descripción generada automáticamente

El algoritmo está mal hecho, ya que dará siempre error en tiempo de ejecución.

5)

Texto

Descripción generada automáticamente

Inserta correctamente “otroNodo” en la lista, quedando como último nodo.

6)

a)

Id alumno -> Pensé en utilizar un long ya que es el entero más grande de la tabla de datos primitivos y la letra no aclara cual sería el máximo valor para ingresar

Concurrencia -> Para la concurrencia me pareció correcto utilizar short ya que su máximo entero positivo a alcanzar me pareció razonable para almacenar concurrencias a una clase (32767)

Clases curso -> También me pareció razonable usar short para almacenar la cantidad de clases de un curso.

En total, cada instancia de un Alumno/Objeto de estas características contaría con un peso de 8 bytes + 2 bytes + 2 bytes = 12 bytes.

En el caso de utilizar un array sabemos que el array pesará cantidad de elementos \*peso de cada elemento. En este caso nosotros tendremos 12 bytes por cada objeto Alumno que almacenaremos en el Array.

Pero en el caso de utilizar una lista encadenada deberemos agregar un nuevo atributo que hará referencia al próximo objeto. Una referencia a un objeto pesa 4 bytes en java, por lo que se concluye que cada objeto Alumno pesará 16 bytes en total si utilizáramos una lista encadenada.

b)

Considero que ambas estructuras están limitadas por el ID del alumno. Esto se debe a que el ID sería único para poder diferenciar alumnos entre sí. Por ende, las estructuras soportarían la cantidad de usuarios que el tipo de dato long soporte. Cuando se alcance ese número y se sobrepase, tendríamos IDs duplicados. Obviamente esto pasará siempre que tengamos memoria para almacenar esa cantidad de alumnos.