Prueba 2 1S - 2015	NOMBRE: NRO.MATRICULA : □ **Estructura de Datos** □ **Complejidad Computacional**

Pilas y Colas

1. Nombre 2 aplicaciones **computacionales** de pilas o colas. [*1 ptos*]

   a. _____

   b. _____

2. Nombre ***dos diferencias*** y ***dos similitudes*** entre pilas y colas [2 ptos]

Dif. 1: _____

Dif. 2: _____

Sim. 1: _____

Sim. 2: _____

3. ¿Especifique el ***tiempo de ejecución*** de los siguientes métodos? [*4 ptos*]

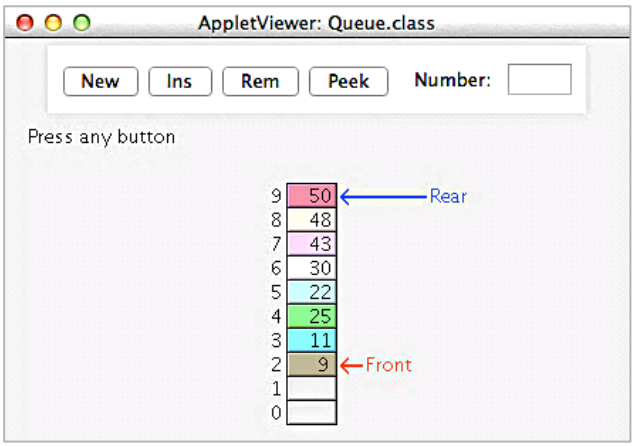
   - **Insertar** en una ***Pila***: O( )
   - **Eliminar** en una ***Pila***: O( )
   - **Insertar** en una ***Cola de Prioridad*** : O( )
   - **Eliminar** en una ***Cola de Prioridad*** : O( )

4. El término ***prioridad*** en una Cola de Prioridad significa: [*1 pto*]

   a. Los elementos de más alta prioridad son insertados primero.
   b. El programador debe priorizar el acceso al arreglo asociado.
   c. El arreglo asociado esta ordenado de acuerdo a la prioridad de los ítems.
   d. Los elementos de más baja prioridad son eliminados primero.

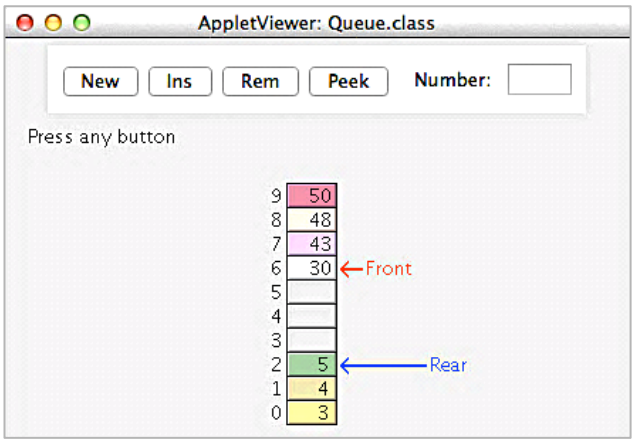
5. Dada las colas de la figuras. Si insertamos los valores indicados en cada caso. ¿En que posición (índice) quedarán almacenados? [*4 ptos*]

Insertar el valor "5"



Posición : _____

Insertar el valor "31"

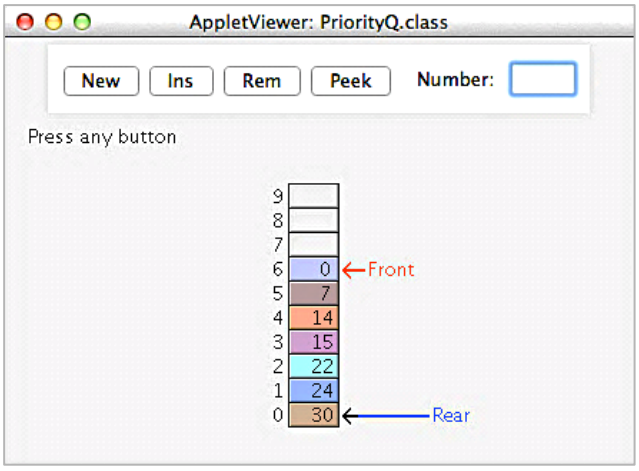


Posición : _____

6. Dada la cola de prioridad de la siguiente figura. [*2 ptos*]

a) ¿Qué pasa en el arreglo si se ejecuta el método *Peek*?

b) ¿En que posición (índice) del arreglo queda el elemento "7" después de ejecutar este método?



(a) _____

_____

_____

_____

(b) Posición : _____.

Listas Enlazadas

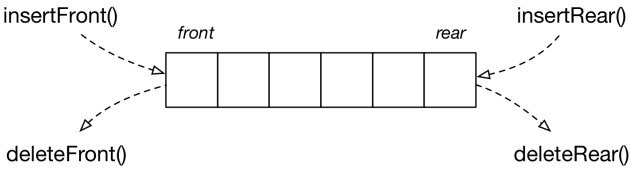
7. Nombre dos **ventajas** de las ***listas enlazadas*** con respecto a los ***arreglos***. [*2 ptos*]

   a. _____

   b. _____

8. La figura siguiente muestra [*2 ptos*]

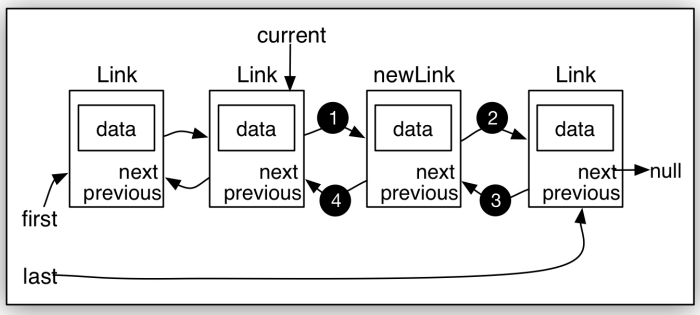
   a. Una lista empotrada
   b. Una lista doblemente terminada
   c. Una lista doblemente enlazada
   d. Una deque
   e. Un cola doblemente ordenada
   f. Ninguna de las anteriores



9. En el método *insertFirst()* de la Lista Enlazada (*linkList.java*), la sentencia ***newLink.next=first;*** significa [*2 ptos*]

   a. El próximo nuevo link a ser insertado referenciará a *first*.
   b. *first* referenciará al nuevo link.
   c. El atributo *next* del nuevo link referenciará al link antiguo de *first*.
   d. *newLink.next* referenciará al link del nuevo *first* en la lista.

10. Defina las **conexiones** (1,2,3 y 4) necesarias para insertar *NewLink.* [*6 ptos*]



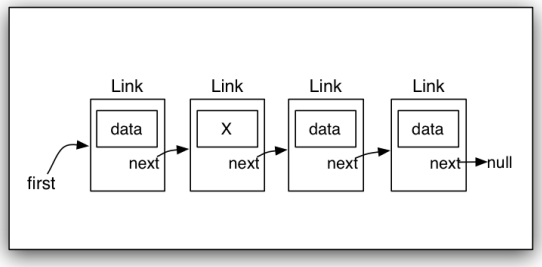
Conexión 1: _____= _____

Conexión 2: _____= _____

Conexión 3: _____= _____

Conexión 4: _____= _____

11. Dada la siguiente figura: [3 ptos]



a) Nombre la estructura de datos mostrada _____

b) Nombre las variables auxiliares necesarias para eliminar "X"

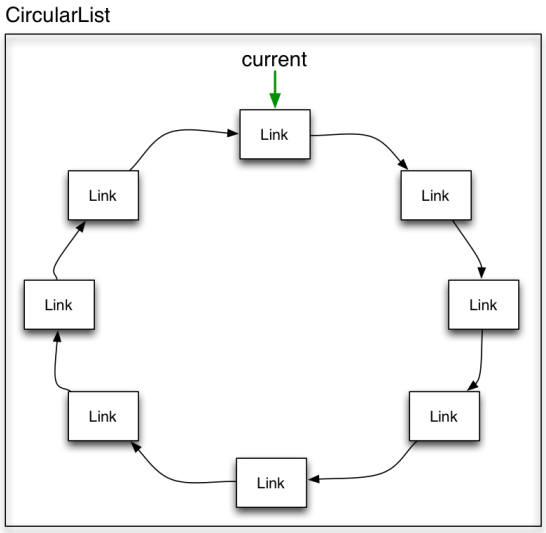
_____

c) Especifique las conexiones necesarias para eliminar "X"

_____

_____

12. Implemente el método ***insertLink()*** de la lista circular. [6 ptos]



public void insertLink(int valor) *//inserta un elemento*
{

}

13. Implemente los métodos ***insertFirst()*** e ***insertLast()*** de la lista doblemente enlazada siguiente [4 ptos] :

```
class Link  {
        public long dData;
        public Link next;
        public Link previous;

        public Link(long d) {
                dData = d; }

        public void displayLink() {
                System.out.print(dData + " "); }
}
```

```
class DoublyLinkedList   {
        private Link first;
        private Link last;

        public DoublyLinkedList() {
                first = null;
                last = null;}

        public boolean isEmpty()  {
                return first == null; }
}
```

*public void **insertFirst**(long dd) // inserta al comienzo de la lista*
{

}

*public void **insertLast**(long dd) // inserta al final de la lista*
{

}