Listas Circulares:

* Es una lista en la que cada nodo tiene un sucesor y un predecesor.
* El sucesor del “Ultimo” es el “primero”.
* El predecesor del “Primero” es el “último”.
* Iniciando en cualquier nodo, se puede recorrer la lista completa.
* El “first” apunta al “ultimo elemento”.
* Son útiles cuando se necesita acceder rápidamente del inicio/fin de la lista.
* El recorrido se detiene cuando el current es igual al first.

Pilas:

* Conjunto de elementos colocados uno sobre otro.
* Únicamente se puede acceder el último elemento insertado.
* Siempre se inserta al final (el final se llama tope).
* Naturaleza LIFO (Last input, First output).
* Tiene 3 operaciones:

Push: Insertar elemento en el tope.

Pop: Eliminar elemento en el tope.

Peek: Ver el elemento en el tope.

* Se puede implementar con listas y Arrays.

Class Stack{

Private int maxsize;

Private Object[] stackarray;

Private int topPosition == -1;

Void push(Object newObject){

If( topPosition < maxsize){

Stackarray[++topPosition] = new Object;

}

}

Object pop(){

Return stackarray[topPosition--]; // No elimina el elemento

}

}

Queue:

* Conjunto de elementos colocados uno detrás de otro, parecido a una fila en la vida real.
* Tiene una cabeza de la lista, el cual es el primer elemento que se encuentra en ésta, la cola solo tiene acceso a este elemento.
* Los elementos se insertan al final, parecido a la pila, la diferencia es que la cola es de naturaleza FIFO (first input fist output).
* Tiene 3 operaciones:
  + Queue: Añadir elemento a la cola, este irá de último en esta.
  + Dequeue: Quitar elemento de la cola, retorna el primer elemento de esta.

public class Cola {

class Nodo {

int info;

Nodo sig;

}

private Nodo raiz,fondo;

Cola() {

raiz=null;

fondo=null;

}

boolean vacia (){

if (raiz == null)

return true;

else

return false;

}

void insertar (int info)

{

Nodo nuevo;

nuevo = new Nodo ();

nuevo.info = info;

nuevo.sig = null;

if (vacia ()) {

raiz = nuevo;

fondo = nuevo;

} else {

fondo.sig = nuevo;

fondo = nuevo;

}

}

int extraer ()

{

if (!vacia ())

{

int informacion = raiz.info;

if (raiz == fondo){

raiz = null;

fondo = null;

} else {

raiz = raiz.sig;

}

return informacion;

} else

return Integer.MAX\_VALUE;

}

public void imprimir() {

Nodo reco=raiz;

System.out.println("Listado de todos los elementos de la cola.");

while (reco!=null) {

System.out.print(reco.info+"-");

reco=reco.sig;

}

System.out.println();

}

public static void main(String[] ar) {

Cola cola1=new Cola();

cola1.insertar(5);

cola1.insertar(10);

cola1.insertar(50);

cola1.imprimir();

System.out.println("Extraemos uno de la cola:"+cola1.extraer());

cola1.imprimir();

}

}