UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

EVALUACIÓN	Parcial	**GRUPO**	N3C	**FECHA**	27/06/2022
MATERIA	Aed1				
CARRERA					
CONDICIONES	El parcial será entregado en los librillos proporcionadas, con letra clara, entendible. **IMPORTANTE:** - Duración 2 hs. - Sin material Puntaje Mínimo: 1 Puntaje Máximo: 45				

Docente con el que curso	**Estudiante NRO**	Estudiante Nombre	Nota

Ejercicio 1 (10 ptos)

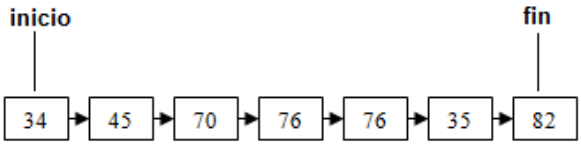
A partir de una implementación de una lista simplemente enlazada, como se muestra a continuación

```
public class Lista implements ILista {
    private NodoLista inicio;
    private NodoLista fin;

    //Constructor
    public void Lista(){
        this.inicio=null;
        this.fin=null;
    }
```

```
public class NodoLista{
    private int dato;
    private NodoLista sig;

    //Constructor
    public NodoLista(int n){
        this.dato=n;
        this.sig=null;
    }
```



Se dispone de las siguientes funciones de la Clase Lista: Inicio(), getInicio() , agregarInicio()
Y de la clase NodoLista: NodoLista (int n), getSig(), getDato()

Se solicita implementar:

a. Realizar un algoritmo que reciba una Lista como parámetro y la imprima por consola en orden inverso.
Ejemplo: Para la lista que se muestra en figura, imprimiría: 82,35,76,76,70,45,34
Firma: **public static void mostrarAlVerre(Lista l) (5 ptos)**

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Solucion Ejercicio 1.a:

```
public static void mostrarAlVerre(Lista L){
        mostrarAlVerre(L.getInicio());
       System.out.println("");
}

public static void mostrarAlVerre( NodoLista n){
        if (n==null){
            return;
        }
        mostrarAlVerre(n.getSig());
        System.out.print(n.getDato()+"-");
}
```

Se solicita implementar, detallando pre y post condiciones, uno de los dos siguientes algoritmos a elección:

b. Realizar un algoritmo que, dada una lista y dado un entero, cuente todos los nodos con números menores e iguales al mismo. Ej: para la lista dispuesta como ejemplo, si el número dado fuera 76, se debería retornar: **6.**
   Firma: **int contarMenorEIgual(Lista lista, int num) (5 ptos)**

Solución Ejercicio 1.b:

```
public static int contarMenorEIgual(Lista lista, int num){
   int cantidad = 0;
   NodoLista aux = lista.getInicio();
   while (aux!=null){
      if (aux.getDato()<=num){
         cantidad = cantidad +1;
      }
      aux= aux.getSig();
   }
   return cantidad;
}
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

c. Realizar un algoritmo recursivo que, dado un entero dado como parámetro, indique si el mismo existe o no en la lista,
Firma: **boolean existe(NodoLista nodo, int num) (5 ptos)**

Solución Ejercicio 1.c:

```
public static boolean existe (NodoLista nodo, int num){
    if (nodo==null){
        return false;
    }
    if (nodo.getDato()==num){
        return true;
    }
    return existe(nodo.getSig(),num);
}
```

Ejercicio 2 (15 ptos)

Dado el siguiente vector ordenado:

10	14	19	25	30	45	49	51

Se solicita

a. implementar un método recursivo que, dado un entero que indica la posición tope del array hasta la que hay que sumar, retorne la suma desde la posición 0 a esa posición tope. Para el ejemplo del vector dispuesto a continuación con tope 3, el retorno debería ser: 68
Firma: **int suma(int vec[], int tope) (10 ptos)**

Solucion Ejercicio 2.a:

```
public static int suma(int vec[], int tope){
        if (tope>vec.length){
                tope = vec.length -1;
        }
        if (tope<0){
                return 0;
        }
        return vec[tope] + suma(vec,tope -1);
}
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

b. Realizar el diagrama de llamadas para el array dispuesto anteriormente con tope 4. **(5 ptos)**

Solución Ejercicio 2.b:

Suma(vec,4)
vec[4]:30 + suma(vec, 3)
vec[3]:25 + suma(vec, 2)
vec[2]:19 + suma(vec, 1)
vec[1]:14 + suma(vec, 0)
vec[0]:10 + suma(vec,-1)
suma(vec,-1) = 0
vec[0]:10 + 0 = 10
vec[1]:14 + 10 = 24
vec[2]:19 + 24 = 43
vec[3]:25 + 43 = 68
vec[4]:30 + 68 = 98
Suma(vec,4) = 98

Ejercicio 3 (10 ptos)

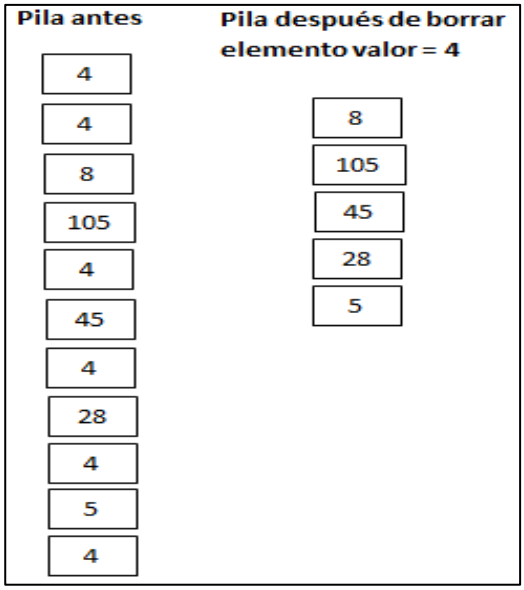
Dada una pila desordenada de elementos y un dato entero, implementar un método que permita eliminar todas las ocurrencias de ese entero en la pila.

Firma: **void eliminarEnPila(Pila p, int valor) (10 ptos)**

Se disponen de las siguientes operaciones de Pila:

new Pila(), esVacia(), cima(), apilar(), desapilar()

Es posible crear estructuras adicionales (Lista, Cola, etc.) disponiendo de sus operaciones si fuera necesario.



Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Solución Ejercicio 3:

```
public static void eliminarEnPila(Pila p, int valor){

        Pila aux = new Pila();

        while (!p.esVacia()){
                if ((int)p.cima()!=valor){
                        aux.apilar(p.cima());
                }
                p.desapilar();
        }
        while (!aux.esVacia()){
                p.apilar(aux.cima());
                aux.desapilar();
        }
}
```

Ejercicio 4 (10 ptos)

Matriz Original:

```
int mat[][] =   {{0,0,0,0,0,0,0,},
                 {0,0,0,0,0,0,0,},
                 {0,0,0,0,0,0,0,},
                 {0,0,0,0,0,0,0,}}
```

Matriz que devuelve la función:

```
int  mat[][] =  {{1,1,1,1,1,1,1,},
                 {1,0,0,0,0,0,1,},
                 {1,0,0,0,0,0,1,},
                 {1,1,1,1,1,1,1,}}
```

Realizar un algoritmo que, dado una matriz de enteros inicializada en 0 (cero), retorne otra matriz con los bordes completados con el valor 1.

Firma: **int[][] cuadroMatriz(int mat[][])**

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Solución Ejercicio 4:

```
public static int[][] cuadroMatriz(int mat[][]){
    int matRet[][] = new int[mat.length][mat[0].length];
    for (int lin=0; lin<mat.length; lin++){
        for (int col= 0; col<mat[0].length;col++){
            if (lin==0 || lin == mat.length-1){
                matRet[lin][col] = 1;
            }else{ if (col==0 || col== mat[0].length-1)
                matRet[lin][col] = 1;
            }
        }
    }
    return matRet;
}
```