UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70
www.ort.edu.uy

EVALUACION	EXAMEN AED1 solución	**GRUPO**	TODOS	**FECHA**	06/02/2024
MATERIA	Algoritmos 1				
CARRERA	Analista Programador / Analista en Tecnologías de la Información				
CONDICIONES	**- Puntos: 100** **- Duración: 2 horas** **- Sin material**				
Nombre	**Nro estudiante**		**Nota**		

Ejercicio 1 (15 ptos)

```
//pre: se asumen que no se repiten los números en una misma fila y en una misma columna
public boolean existe(int mat[][]){
        int largoFilas = mat.length;
        int largoCol = mat[o].length;
        boolean existe = false;

        for(int i=0; i< largoFilas && !existe; i++){

                int maxFila = Integer.MIN_VALUE;
                int colMax = -1;

                for(int j=0; j< largoCol; j++){
                        if(mat[i][j] > maxFila){
                                maxFila = mat[i][j];
                                colMax = j;
                        }
                }

                Int minCol = Integer.MAX_VALUE;
                for(int k=0; k< largoFilas; k++){
                        if(mat[k][colMax] < minCol){
                                minCol = mat[k][colMax];
                        }
                }
                If(maxFila == minCol){
                        existe = true;
                }
        }
        return existe;
}
```

UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70
www.ort.edu.uy

Ejercicio 2 (20 ptos)

Dado el siguiente algoritmo y el siguiente array, de largo n:


```
void ordenar(int arr[], int n)
{
    int i, j, min_idx;

    for (i = 0; i < n - 1; i++)
        min_idx = i;
        for (j = i + 1; j < n; j++) {
            if (arr[j] < arr[min_idx])
                min_idx = j;
        }
        if (min_idx != i)
            swap(arr, min_idx ,i);
    }
}
```

a) Implemente el método swap, que utiliza el algoritmo:

```
public void swap(int v[],int valor1 , int valor2){

        int aux = v[valor1];
        v[valor1] = v[valor2];
        v[valor2] = aux;
 }
```

b) Indique cual es la secuencia de ordenamiento al aplicarlo

64-25-12-22-11
11-25-12-22-64
11-12-25-22-64
11-12-22-25-64
11-12-22-25-64

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70
www.ort.edu.uy

Ejercicio 3 (40 ptos)

Dado dos listas simplemente encadenadas de números enteros positivos, de igual o diferente tamaño, se pide:

a) Realizar un algoritmo recursivo que reciba los nodos iniciales de cada lista e indique (mediante un valor booleano) si las listas son exactamente iguales (son iguales si tienen el mismo tamaño y sus nodos coinciden en valores y posiciones) (**15 puntos**).

```
public boolean iguales(Nodo nodoL1, Nodo nodoL2){

        if(nodoL1 == null && nodoL2 !=null  || nodoL1 != null && nodoL2 ==null){
                return false;
        }
        else{
                if(nodoL1 == null && nodoL2 ==null){
                        return true;
                }
                esle{



                        if(nodoL1.getDato() != nodoL2.getDato()){
                                return false;
                        }
                        else{
                                iguales(nodoL1.getSiguiente(), nodoL2.getSiguiente()){
                        }

                }
        }
}
```

b) Realizar el diagrama de llamadas para el siguiente ejemplo (**10 puntos**):

L1: 34-33-1-45-66-3-11-77
L2: 34-33-1-8-66-5

```
iguales(nodoL1, Nodo nodoL2)
        nodoL1(34)
        nodoL2(34)
        iguales(nodoL1, Nodo nodoL2)
                nodoL1(33)
                nodoL2(33)
                iguales(nodoL1, Nodo nodoL2)
                        nodoL1(1)
                        nodoL2(1)
                        iguales(nodoL1, Nodo nodoL2)
                                nodoL1(45)
                                nodoL2(8)
                                return false
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70
www.ort.edu.uy

```
                        fin
                return false
                fin
        return false
        fin
return false
fin
```

c) Realizar un algoritmo que reciba ambas listas y un valor, y retorne una pila con los elementos de las listas cuyo valor este por encima del valor dado, de forma de que se coloquen en el tope de la lista los valores de L1 (de comienzo a fin) y luego los valores de L2 (de comienzo a fin). Para el ejemplo anterior y un valor de 40, se debería retornar la siguiente pila (**15 puntos**):

45
66
77
66

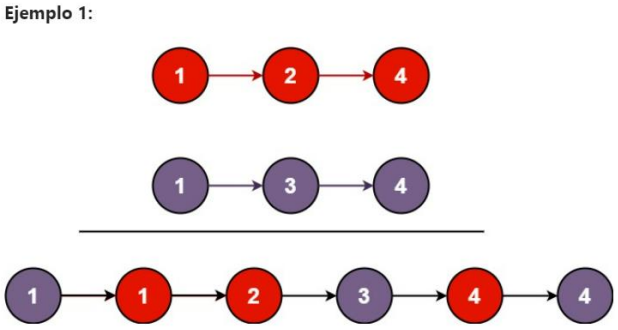
Se disponen de las estructuras vistas en clase (Cola y Pila) con los métodos básicos vistos de dichas estructuras (Cola: esVacia, vaciar, encolar, desencolar, frente; Pila: esVacia, vaciar, top, desapilar, push)

```
public Pila proceso (Lista l1, Lista l2, int valor){
        Pila pAux = new Pila();
        Nodo nL1 = l1.getInicio()
        Nodo nL2 = l2.getInicio()

        while(nL1 != null){
                if(nL1.getDato() > valor){
                        pAux.push(nL1.getDato());
                }
                nL1 = nL1.getSiguiente();
        }

        while(nL2 != null){
                if(nL2.getDato() > valor){
                        pAux.push(nL2.getDato());
                }
                nL2 = nL2.getSiguiente();

        }

        Pila pRet = new Pila();

        while(!pAux.esVacia()){
                pRet.push(pAux.top());
                pAux.desapilar();
        }
        return pRet;
}
```

Ejercicio 4 (25 ptos)

Se proporcionan dos listas simplemente enlazadas de enteros, ambas ordenadas en forma ascendente. Se pide realizar un algoritmo que reciba ambas listas como parámetro y retorne una nueva lista con la unión de ambas listas en forma ordenada (el algoritmo debe ser lo más eficiente posible).

Ejemplo 1:



Entrada: lista1 = [1,2,4], lista2 = [1,3,4]
Salida: [1,1,2,3,4,4]

```
public Lista union (Lista l1, Lista l2){

	Lista lRetorno = new Lista();
	Nodo nL1 = l1.getInicio();
	Nodo nL2 = l2.getInicio();

	while(nL1 != null ||  nL2 != null){
		if(nL1 != null &&  nL2 != null){
			if(nL1.getDato() <= nL2.getDato()){
				lRetorno.insertarFinal(nL1.getDato());
				nL1 = nL1.getSiguiente();
			}
			else{
				lRetorno.insertarFinal(nL2.getDato());
				nL2 = nL2.getSiguiente();

			}
		}
		else{

			if(nL1 != null){
				lRetorno.insertarFinal(nL1.getDato());
				nL1 = nL1.getSiguiente();
			}
			else{
				lRetorno.insertarFinal(nL2.getDato());
				nL2 = nL2.getSiguiente();
			}
		}
	}
	return lRetorno;
```