Listas

Ejercicio Nro. 1:

La función **agregar** agrega un nuevo nodo con el valor **x** al final de la lista referenciada por **p**.

```
void agregarNodo(Nodo*& p, int x)
```

Ejercicio Nro. 2:

La función **mostrar** recorre la lista **p** y muestra por pantalla el valor que contienen cada uno de sus nodos.

```
void mostrar(Nodo* p)
```

Ejercicio Nro. 3:

La función **liberar** recorre la lista **p** liberando la memoria que ocupan cada uno de sus nodos.

```
void liberar(Nodo*& p)
```

Ejercicio Nro. 4:

La función **buscar** permite determinar si alguno de los nodos de la lista **p** contiene el valor **v**.

Retorna un **puntero al nodo** que contiene dicho valor o **NULL** si ninguno de los nodos lo contiene.

```
Nodo* buscar(Nodo* p, int v)
```

Ejercicio Nro. 5:

La función **eliminar** permite eliminar de la lista **p** al nodo que contiene el valor **v**.

```
void eliminar(Nodo*& p, int v)
```

Ejercicio Nro. 6:

La función **eliminarPrimerNodo** elimina el primer nodo de la lista y retorna el valor que este contenía.

```
int eliminarPrimerNodo(Nodo*& p)
```

Ejercicio Nro. 7:

La función **insertarOrdenado** permite insertar el valor **v** respetando el ctriterio de ordenamiento de la lista **p**; se presume que la lista está ordenada o vacía. Retorna la dirección de memoria del nodo insertado.

```
Nodo* insertarOrdenado(Nodo*& p, int v)
```

Ejercicio Nro. 8:

La función **ordenar** ordena la lista direccionada por **p**. La estrategia consiste en eliminar uno a uno los nodos de la lista e insertarlos en orden en una lista nueva; finalmente hacer que **p** apunte a la nueva lista.

```
void ordenar(Nodo*& p)
```

Ejercicio Nro. 9:

La función **buscaEInsertaOrdenado** busca el valor **v** en la lista **p**. Si no lo encuentra entonces lo inserta respetando el criterio de ordenamiento. Retorna un puntero al nodo encontrado o insertado y asigna el valor **true** o **false** al parámetro **enc** según corresponda.

```
Nodo* buscaEInsertaOrdenado(Nodo*& p, int v, bool& enc)
```

Pilas

Ejercicio Nro. 1:

Dada una pila y un valor **X**, desarrollar un procedimiento que elimine los 2 primeros nodos de la pila y deje el valor **X** como primero.

Ejercicio Nro. 2:

Dada una pila y un valor **X**, desarrollar un procedimiento que inserte **X** como tercer valor de la pila. Retornar un parámetro con valor 'S' o 'N' según haya sido exitoso o no el requerimiento.

Ejercicio Nro. 3:

Dada una pila y dos valores **X** e **Y**, desarrollar un procedimiento que inserte el valor **X** en la posición **Y** de la pila si es posible.

Ejercicio Nro. 4:

Dada una pila y dos valores **X** e **Y**, desarrollar un procedimiento que reemplace cada valor igual a **X** que se encuentre en la pila por el valor **Y** retornando la pila modificada. En caso de no haber ningún valor igual a **X** retornar la pila sin cambio.

Ejercicio Nro. 5:

Definir una función **INVERSA** que evalúe dos conjuntos de caracteres separados por un punto y retorne True si los conjuntos son inversos (ej: ABcDe.eDcBA) o False si no lo son. Los conjuntos deben ingresarse letra por letra por teclado.

Ejercicio Nro. 6:

Desarrollar un procedimiento que ingrese por teclado un conjunto de Apellidos de estudiantes y los imprima en orden inverso al de ingreso.

Ejercicio Nro. 7:

Dada una pila desarrollar un procedimiento que ordene la misma de acuerdo al valor de sus nodos y la retorne.

Para los siguientes ejercicios utilizaremos un nodo que contiene un registro con los datos:

- Legajo
- Nombre y Apellido
- Curso

Colas

Ejercicio Nro. 1:
Dada una cola, desarrollar un procedimiento que elimine 2 nodos de la misma (indicar con un parámetro 'S'/'N' si ello fue, o no posible)

Ejercicio Nro. 2:
Dada una cola, desarrollar una función que devuelva la cantidad de nodos que tiene.

Ejercicio Nro. 3:
Dadas dos colas **COLA** y **COLB**, desarrollar un procedimiento que genere una única cola **COLC** a partir de ellas. (Primero los nodos de **COLA** y luego los de **COLB**).

Ejercicio Nro. 4:
Dada una cola, imprimirla en orden natural si tiene más de 100 nodos, caso contrario imprimirla en orden inverso.

Ejercicio Nro. 5:
Dadas dos colas **COLA** y **COLB**, desarrollar un procedimiento que genere otra cola **COLC** por apareo del campo **LEGAJO** del registro (define orden creciente en ambas).
Nota: **COLA** y **COLB** dejan de ser útiles después del apareo.