**Pilas/Colas ejercicios**  
  
1.- Hacer una función que reciba como parámetros una pila, y un número.  
La función debe retornar tantos elementos como diga el número (segundo parámetro).

Entrada : mifuncion([‘Manzana’,‘Cebolla’,‘Apio’,‘Naranja’,‘Papaya’,‘Sandía’,‘Melón’],

Salida: [‘Manzana’,‘Cebolla’,‘Apio’,‘Naranja’]

2.- Escribe una función “reemplazar” que tenga como parámetros una pila de elementos de tipo Number, y dos valores también de tipo Number “nuevo” y “viejo”, de forma que si el segundo valor aparece en algún lugar de la pila, sea reemplazado por el primero (Solo la primera vez), y hará pop de los elementos más nuevos.

Entrada: reemplazar([3,2,3,4,6,8,1,2,5,5], 7, 2)

Salida: [3,2,3,4,6,8,1,7]

3.- Un conductor maneja de un pueblo origen a un pueblo destino, pasando por varios  
pueblos. Una vez en el pueblo destino, el conductor debe regresar a casa por el mismo  
camino. Muestre el camino recorrido tanto de ida como de vuelta.

Recorrido: Pueblo Origen → pueblo 1 → pueblo 2 → destino

Regreso: destino → pueblo 2’ → pueblo 1 → Pueblo Origen

4.-  Un almacén tiene capacidad para apilar “n” contenedores. Cada contenedor tiene un número de identificación. Cuando se desea retirar un contenedor específico, deben retirarse primero los contenedores que están encima de él y colocarlos en otra pila, efectuar el retiro y regresarlos a su respectivo lugar.  
  
5.- Se tiene una cola de colores y se tiene que dividir en dos colas, respetando el orden y alternando a partir de su índice. los pares en una y los nones en otra

Cola original: [ amarillo, rosa, rojo, verde, azul, negro, morado, blanco]

Cola 1: [ amarillo, rojo, azul, morado]

Cola 2: [rosa, verde, negro, blanco]

6.- Se tiene una cola en la cual se han repartido tickets con el orden de atención. Sin embargo, llegada la hora de inicio hay muchos “colados”, es por esto que se le ordena al vigilante que retire a todos aquellos que no tienen ticket. Muestra la cola inicial, qué elementos fueron retirados de la cola y la cola final.  
  
Sugerencia: desencolar cada elemento, si tiene el ticket se vuelve a encolar, si no se retira.

Cola: [

{ user:‘User1’, ticket:true },

{ user:‘User2’, ticket:true },

{ user:‘User3’, ticket:false },

{ user:‘User4’, ticket:true },

{ user:‘User5’, ticket:false },

{ user:‘User6’, ticket:false },

{ user:‘User7’, ticket:true },

{ user:‘User8’, ticket:true },

{ user:‘User9’, ticket:true },

{ user:‘User10’, ticket:false },

{ user:‘User11’, ticket:true },

];

**Listas Enlazadas**  
  
1.- Implemente un procedimiento que inserte un dato de modo similar al insertar, al final de la lista. Pero ahora, no se debe permitir insertar datos repetidos, si un dato ya está almacenado entonces la lista no cambia.  
  
2.- Escribir un programa para formar una lista que realice las siguientes tareas:

a) Crear una Lista Enlazadas de Números Enteros Positivos al azar, la inserción se realiza por el último nodo.

b) Recorrer la lista para mostrar los Elementos por pantalla.

c) Mostrar todos los Nodos que superen un valor determinado.

**Arboles binarios.**

1. Escribe una función que dados dos árboles binarios A y B, determine si son idénticos o no.
2. Escribe una función que dado un árbol binario A, obtenga una copia B del mismo.
3. Escribe una función que visualice los nodos que están en el nivel n de un árbol binario.
4. Escribe una función que devuelva el número de hojas de un árbol binario.

**Ejercicios de Búsqueda (Secuencial y Binaria)**  
  
1 .- Crear una función que encuentre el número más repetido en un array  
Ej. Input - [3,6,1,8,2,3,6,3,2,5,6]  
Salida -> 6  
  
2.- Crear una función que encuentre las letras en un arreglo de números y regrese un nuevo arreglo de letras  
Ej input -> [8,‘e’,7,2,‘a’,‘d’,‘f’,2,3,1,4,3]  
Salida → [‘e’,‘a’,‘d’,‘f’]  
  
3.- Crear una función que busque el número más grande en un arreglo  
Ej [1,4,7,2,4,1,9,4,0,2,4,5,12]  
Salida -> 12  
  
4.- Crear una función que busque el número más pequeño en el arreglo  
Ej. [1,4,5,-1,-7,2,3,9]  
Salida -->  -7  
  
5.- Crear una clase llamada  **Contacto** con los siguientes datos

* Nombre
* Apellidos
* Telefono

Después crear una clase que sea **ListaContactos** la cual tendrá dos métodos

* Agregar un contacto
* Buscar un contacto por nombre

Prueba que tus clases funcionen correctamente.  
  
**Ejercicios ordenamiento**

1. Crea una función que ordene de forma descendente  un arreglo de números
   1. Ej. Entrada --> [9,3,1,6,5,88,-1,2,7]
   2. Salida → [88,9,7,6,5,3,1,-1]
2. Crear una función que ordene una lista de caracteres según su valor decimal de acuerdo a la tabla ascii ref: <http://www.asciitable.com/>
   1. Ej. Entrada ---> [‘b’,‘h’,‘w’,‘e’,‘a’]
   2. Salida ----> [‘a’,‘b’,‘e’,‘h’,‘w’]
3. Crear una clase Alumno con los siguientes datos:
   1. Nombre y calificaciones(arreglo de 5 números) y tiene un método para obtener el promedio
   2. Crear otra clase llamada Salon que tenga un método que ordene alumnos de mayor a menor promedio
4. Crear una funcion que ordene palabras de menor numero de letras a mayor
   1. Ej Entrada [“adios”,“hola”,“maximo”,“uno”,“despedida”]
   2. salida -> [“”uno,“hola”,“adios”,“maximo”,“despedida”]
5. Crear una función que ordene un arreglo de coordenadas x, y desde a partir del punto mas cercano a 0,0
   1. Entrada -> [[7,3],[2,2],[1,0],[4,3]]
   2. Salida ---> [[1,0],[2,2],[4,3],[7,3]]