

Actividad 1






Actividad de Aprendizaje 01. Tipos de Dato primitivos y Tipos de Dato Estructurados

Problema:

Diseñe y codifique un programa que muestre un menú con las opciones:

- Tamaño y rangos de los Tipos de Dato Primitivos
- Ejemplo de uso de Tipo de dato Estructurado
- Salir

que funcione de forma cíclica hasta optar por la opción c) para salir del programa.

 Inicio  Tablero  Eventos  Mis Cursos  Este curso

Sin signo), **entero corto** (con y sin signo), **entero largo** (con y sin signo), **real de precisión simple** y **real de doble precisión** en una tabla basada en el siguiente modelo:

Tipo de dato	Bits	Valor mínimo	Valor máximo
Caracter con signo			
Caracte sin signo			
Entero corto con signo			
Entero corto sin signo			
Entero largo con signo			
Entero largo sin signo			
Real de precisión simple			
Real de doble precisión			

La opción b) Solicitará un valor entero n para determinar el tamaño de matriz a utilizar, con un mínimo de 3 y un máximo de 10, luego rellenará dos matrices de tamaño $n \times n$ con valores aleatorios de tipo **real** comprendidos en el rango entre -100.00 y 100.00, mostrará el contenido de las matrices en un formato legible y comprensible con dos decimales, además de:

- Matriz resultante de la multiplicación de dichas matrices
- Matriz resultante de la suma de las mismas

Requerimientos:

- El estilo de programación debe ser Orientado a Objetos
- Los datos para la tabla mostrada en la opción a) deben surgir de los propios recursos del lenguaje de programación (biblioteca de funciones), por lo que una simple transcripción de tamaños y rangos no será válida.
- Los valores aleatorios para la opción b) deben ser diferentes en cada ejecución.

Actividad 2

Arreglos, Arreglos de Registros y Arreglos de Objetos

Problema:

Haga un programa que sirva para almacenar un inventario, cada producto en el inventario tiene los datos:

- código de barras (cadena con 13 dígitos)
- nombre (cadena)
- peso (real)
- fecha de entrada (clase completa)
- precio de mayoreo (real)
- precio al menudeo (real)
- existencia actual (entero)

El programa debe ofrecer opciones para ingresar nuevo producto, y retirar producto actualizando la cantidad en existencia. Toda operación se hace mediante el código de barras, representado por una cadena numérica de 13 posiciones.

Requerimientos:

- a. El estilo de programación debe ser Orientado a Objetos
- b. Pueden entrar nuevos lotes de producto con un código ya existente, actualizando existencia.
- c. Cuando un producto se agota queda con cantidad actual en ceros, no es necesario eliminarlo
- d. Debe ser posible almacenar al menos 500 entradas de producto
- e. Utilice una clase para el manejo de fechas
- f. Utilice un registro para todos los datos del producto
- g. Utilice un arreglo (de registros) para almacenar los productos
- h. Utilice clases independientes y separadas (con sus respectivas relaciones) para:
 - La fecha
 - El producto
 - La colección de productos
 - El menú

Actividad 3

Actividad de Aprendizaje 03. La Lista, implementación estática

Problema:

Una radiodifusora necesita publicar su lista de éxitos de la 50 canciones más escuchadas, y le pasa los datos a su webmaster, aún no saben en qué orden se publicará la lista, si en orden alfabético, por nombre del autor, o del intérprete, por nombre de la canción, o posición en el ranking, por lo que será necesario utilizar una estructura de datos que permita un manejo aleatorio de los datos, es decir una lista.

Haga un programa que cubra el problema. Deberá mostrar la lista en pantalla todo el tiempo, permitiendo añadir nuevos elementos y así como eliminarlos, y mostrar los cambios al momento.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Utilizará un arreglo para almacenar los datos
- c) La clase *Lista* y todas su operaciones deberán alojarse en una librería, separándola del resto del programa
- d) El uso de la Lista debe hacerse exclusivamente a través de sus respectivos métodos
- e) Las operaciones a implementar, independientemente de que sean utilizadas o no en éste programa son:
 - inicializa,
 - vacía,
 - llena,
 - inserta,
 - elimina,
 - recupera,
 - primero,
 - último,
 - anterior,
 - siguiente,
 - anula

Actividad 4

Actividad de Aprendizaje 04. Aplicación de Pila y Cola

Problema:

Implemente un programa que reciba una cadena que contenga una expresión con notación infija y la convierta a su equivalente expresión con notación posfija, e imprima el resultado en pantalla.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Las clases *Pila* y *Cola* deben tener una implementación estática (con arreglos).
- c) La cadena con la expresión infija se pasará a una Cola
- d) La conversión de infija a posfija debe hacerse con el método que utiliza una Pila
- e) La expresión resultante se pasará directamente a una Cola
- f) Los operadores a considerar son sólo binarios: suma (+), resta (-), multiplicación (*), división (/), y potencia (^)
- g) Se incluye el manejo de paréntesis como agrupador

Actividad 5

Actividad de Aprendizaje 05. Métodos de Búsqueda

Problema:

Reutilice el resultado de la actividad 03, y adapte a una nueva necesidad: almacenar toda la discografía de la radiodifusora que consta de cerca de 3,000 títulos más los que se vayan añadiendo. La lista servirá para las complacencias, por lo que debe contar también con el nombre del archivo MP3. Las complacencias se hacen a través del nombre de la canción o del nombre del cantante.

Haga un programa que realice la búsqueda en cualquiera de las dos formas y regrese el nombre del archivo MP3 correspondiente.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Debe ofrecer dos opciones de búsqueda: lineal y binaria
- c) Los métodos de búsqueda se implementarán como métodos de la clase Lista

Actividad 6

Actividad de Aprendizaje 06. Métodos de ordenamiento iterativos

Problema:

Reutilice el resultado de la actividad 05. Se ha encontrado un problema en el programa: la búsqueda binaria no funciona muy bien, el jefe de programadores se ha dado cuenta que la lista casi nunca está ordenada y por eso falla, por lo que es necesario añadirle una opción para ordenarla.

Haga un programa que realice el ordenamiento de la lista, opcionalmente por nombre de la canción como por nombre del cantante. Para evaluar su funcionamiento la radiodifusora quiere probar con distintos tipos de ordenamiento.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Debe ofrecer las opciones de ordenamiento: burbuja (mejorada), shell, inserción, y selección
- c) Los métodos de ordenamiento se implementarán como métodos de la clase Lista

Actividad 7

Actividad de Aprendizaje 07. Métodos de ordenamiento recursivos

Problema:

Haga un programa que genere valores enteros aleatorios entre 0 y 1'000,000 con los cuales se rellene un arreglo de 100,000 elementos y luego los ordene tomando en cuenta el tiempo requerido para ello e informe una vez terminado el proceso de ordenación. El programa informará el tiempo necesario para ordenar el arreglo con los métodos de ordenamiento:

- Burbuja (mejorada),
- Shell,
- Inserción,
- Selección,
- Mezcla, y
- QuickSort

Todos los tiempos se imprimirán en una sola pantalla para fines comparativos.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) El conjunto de elementos debe ser idéntico para cada caso de ordenamiento
- c) Todos los métodos de ordenamiento deben ser métodos de la misma clase

Actividad 8

Actividad de Aprendizaje 08. Apuntadores

Problema:

Modifique la lista resultante de las actividades 03, 05, y 06 de modo que la implementación se realice mediante un arreglo de apuntadores a objeto desarrollado, y adapte las operaciones que así lo requieran para que trabajen a través los respectivos apuntadores para llegar al objeto al que refieren.

Requerimientos:

1. El estilo de programación debe ser Orientado a Objetos
2. Debe incluir el manejo adecuado de apuntadores, es decir, la adecuada asignación de direcciones válidas/inválidas de acuerdo a la acción efectuada
3. Las operaciones que requieran creación o eliminación de elementos deben considerar la reserva y liberación de memoria en tiempo real, por lo que un elemento del arreglo que esté en desuso no debe apuntar a ninguna instancia
4. Salvo para fines de comparación, los ordenamientos no deben hacer uso, asignación, o reasignación de los datos del objeto, sino que manipularán los punteros contenidos en el arreglo, en otras palabras: en el arreglo intercambie apuntadores, no los datos detrás de los apuntadores.

Considere:

- El arreglo que forma parte de la lista contiene apuntadores
- Para la inserción y eliminación se desplazan los apuntadores, no los datos
- Cuide todo el tiempo qué elementos deben ser liberados
- Asigne NULL a los apuntadores cuando el elemento ha sido liberado (eliminado)
- La operación *anula* (deleteAll) **debe** eliminar todos los elementos, y liberar la memoria de cada uno de ellos.
- Debe implementar un destructor, para liberar memoria cuando la lista deje de existir
- Como un plus (no espere de los requerimientos) usted podría generar el arreglo de forma dinámica, con las responsabilidades que ello acarrea.

Actividad 9

Actividad de Aprendizaje 09. La Lista, implementación dinámica simplemente ligada

Problema:

Tome el problema y requerimientos de la actividad 05 (sólo con búsqueda lineal), y cubra las necesidades utilizando una lista simplemente ligada en lugar de una lista estática.

Requerimientos:

- a. El estilo de programación debe ser Orientado a Objetos
- b. Considere de forma separada (archivos separados) la clase *Nodo* respecto de la clase que servirá para instanciar los datos que almacena la lista (esto no aplica si decide utilizar el concepto de clases anidadas).
- c. La clase *Lista* y todas sus operaciones deberán alojarse en una librería, separándola del resto del programa
- d. El uso de la Lista debe hacerse exclusivamente a través de sus respectivos métodos
- e. Las operaciones a implementar, independientemente de que sean utilizadas o no en éste programa son, cuando menos: inicializa, vacía, llena, insertar, elimina, recupera, busca (lineal), primero, último, anterior, siguiente, y anula
- f. La actividad requiere un sólo tipo de Lista Simplemente Ligada, a opción del alumno puede ser lineal o circular, incluso puede optar por trabajar con encabezado "de atributos".

Entregables:

Actividad 10

Actividad de Aprendizaje 10. La Lista, implementación dinámica doblemente ligada

Problema:

Tome el problema y requerimientos de la actividad 09, y cubra las necesidades utilizando una lista doblemente ligada en lugar de una lista simplemente ligada.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Debe ser suficiente con el cambio de librería para que el resto del programa de forma idéntica.

Actividad 11

Actividad de Aprendizaje 11. La Pila y la Cola, implementación dinámica

Problema:

Implemente la pila y la cola **basadas en listas ligadas**, ~~a través del uso de herencia a partir de la lista simplemente ligada~~. Y aplíquelo a la actividad 04.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos
- b) Debe ser suficiente con el cambio de librería para que el resto del programa de forma idéntica.

Actividad 12

Actividad de Aprendizaje 12. El Árbol Binario de Búsqueda, implementación dinámica

Problema:

Haga un programa que genere una cantidad **N** (definida por el usuario) de valores aleatorios de tipo entero en el rango de 0 a 100,000 que se inserte al árbol conforme cada valor sea generado.

El programa mostrará en pantalla los valores generados en el orden en que se insertan, enseguida el resultado de los recorridos preorder, inorder y postorder, seguido de la altura correspondiente al subárbol izquierdo y al subárbol derecho debajo de la raíz del árbol.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos

Entregables:

Actividad 13

Actividad de Aprendizaje 13. El Árbol AVL, implementación dinámica

Problema:

Reutilice el programa resultante de la actividad 12. Agregue y/o modifique los métodos necesarios para que el árbol sea implementado como AVL.

Requerimientos:

- a) El estilo de programación debe ser Orientado a Objetos

