Instrucciones para Implementar el Pseudocódigo en Cualquier Lenguaje

Objetivo:

El estudiante debe trasladar el pseudocódigo a un lenguaje de su preferencia (**Python, Java, C++, C#, JavaScript, etc.**) y mejorar la visualización de los datos.

Paso 1: Elegir el Lenguaje

Seleccionar un lenguaje y su método de visualización:

- **Consola:** Formatear la salida con tablas o diagramas ASCII.
- **Interfaz gráfica:** Usar **Tkinter, Swing, WinForms, React.js, etc.**.

Paso 2: Implementar las Listas

Desarrollar las siguientes estructuras con funciones para insertar, eliminar y mostrar datos:

- **Lista Contigua (Arreglo)**
- **Lista Ligada (Simple)**
- **Lista Doblemente Ligada**
- **Lista Indexada**

Paso 3: Mejorar la Visualización

- **Consola:** Mostrar datos en tabla con encabezados.
- **Interfaz gráfica:** Usar listas visuales, botones y colores.

Paso 4: Menú de Opciones

Crear un menú interactivo con:

1. Insertar datos.
2. Eliminar datos.
3. Mostrar datos visualmente.
4. Salir del programa.

Entrega del Proyecto

- Código en el lenguaje elegido.
- Video de la ejecución.
- Breve explicación de los cambios hechos.

Pseudocodigo

```
// Simulación de Listas en PSeInt con Correcciones


Proceso Simulacion_Listas
  Definir opcion Como Entero

  Repetir
    // Menú principal para seleccionar el tipo de lista
    Escribir "Seleccione el tipo de lista a probar:"
    Escribir "1. Lista Contigua (Arreglo)"
    Escribir "2. Lista Ligada (Simple)"
    Escribir "3. Lista Doblemente Ligada"
    Escribir "4. Lista Indexada"
    Escribir "5. Salir"
    Leer opcion

    // Llamamos a la función correspondiente según la opción elegida
    Segun opcion Hacer
      Caso 1:
        Lista_Contigua()
      Caso 2:
        Lista_Ligada()
      Caso 3:
        Lista_Doblemente_Ligada()
      Caso 4:
        Lista_Indexada()
      Caso 5:
        Escribir "Saliendo del programa..."
      De Otro Modo:
        Escribir "Opción inválida, intente nuevamente."
    FinSegun
  Hasta Que opcion = 5
FinProceso

// ====================== LISTA CONTIGUA ======================
SubProceso Lista_Contigua
  Definir lista, n, i Como Entero
  Dimension lista[10] // Arreglo de tamaño fijo

  Escribir "Ingrese el número de elementos (máximo 10):"
  Leer n

  // Insertamos los elementos en la lista contigua
  Para i <- 0 Hasta n-1 Hacer
```

```
    Escribir "Ingrese el elemento ", i, ":"
    Leer lista[i]
  FinPara

  // Mostramos la lista contigua
  Escribir "Lista Contigua: "
  Para i <- 0 Hasta n-1 Hacer
    Escribir lista[i] " " Sin Saltar
  FinPara
  Escribir ""
FinSubProceso

// ====================== LISTA LIGADA ======================
SubProceso Lista_Ligada
  Definir nodo, cabeza, actual, fin1 Como Entero
  Dimension nodo[10,2] // Matriz que almacena el valor y el puntero al siguiente
nodo
  cabeza <- -1
  actual <- 0
fin <- 0  // Bandera de finalización

Escribir "Ingrese elementos (-1 para terminar):"
Mientras actual < 10 Y fin1 = 0 Hacer
        Definir valor1 Como Entero
        Leer valor1
        Si valor1 = -1 Entonces
        fin1 <- 1  // Marcamos la bandera para salir
Sino
        // Se inserta el nuevo nodo al inicio
        nodo[actual,0] <- valor1 // Guardamos el dato
        nodo[actual,1] <- cabeza // El siguiente apunta al anterior cabeza
        cabeza <- actual // Actualizamos la cabeza de la lista
        actual <- actual + 1
FinSi
FinMientras

// Mostramos la lista ligada
Escribir "Lista Ligada: "
actual <- cabeza
Mientras actual <> -1 Hacer
        Escribir nodo[actual,0] " -> " Sin Saltar
        actual <- nodo[actual,1]
FinMientras
Escribir "NULL"
FinSubProceso
```

```
// ==================== LISTA DOBLEMENTE LIGADA
====================
SubProceso Lista_Doblemente_Ligada
  Definir nodo, cabeza, actual, previo, fin1 Como Entero
  Dimension nodo[10,3] // Matriz con (dato, puntero siguiente, puntero anterior)
  cabeza <- -1
  actual <- 0
  previo <- -1
fin <- 0  // Bandera de finalización

Escribir "Ingrese elementos (-1 para terminar):"
Mientras actual < 10 Y fin1 = 0 Hacer
        Definir valor1 Como Entero
        Leer valor
        Si valor = -1 Entonces
        fin1 <- 1  // Salimos del bucle
Sino
        // Insertamos el nuevo nodo al inicio de la lista doblemente ligada
        nodo[actual,0] <- valor1 // Guardamos el dato
        nodo[actual,1] <- cabeza // Puntero siguiente apunta a la antigua cabeza
        nodo[actual,2] <- previo // Puntero anterior apunta al nodo previo

        // Si la lista no está vacía, el anterior de la cabeza apunta al nuevo nodo
        Si cabeza <> -1 Entonces
                nodo[cabeza,2] <- actual
        FinSi
        cabeza <- actual // Actualizamos la cabeza de la lista
        previo <- actual // Actualizamos el nodo previo
        actual <- actual + 1
FinSi
FinMientras

// Mostramos la lista en orden normal
Escribir "Lista Doblemente Ligada: "
actual <- cabeza
Mientras actual <> -1 Hacer
        Escribir nodo[actual,0] " <-> " Sin Saltar
        actual <- nodo[actual,1]
FinMientras
Escribir "NULL"
FinSubProceso

// ==================== LISTA INDEXADA ====================
SubProceso Lista_Indexada
  Definir lista, indice, n, i Como Entero
  Dimension lista[10], indice[10] // Una lista de valores y otra de índices
```

```
  Escribir "Ingrese el número de elementos (máximo 10):"
  Leer n

  // Guardamos los valores y sus índices
  Para i <- 1 Hasta n Hacer
    Escribir "Ingrese el elemento ", i, ":"
    Leer lista[i]
    indice[i] <- i // Cada valor tiene un índice asociado
  FinPara

  // Mostramos la lista indexada
  Escribir "Lista Indexada:"
  Para i <- 1 Hasta n Hacer
    Escribir "Índice: ", indice[i], " -> Valor: ", lista[i]
  FinPara
FinSubProceso
```