ESTRUCTURA DE DATOS

Trabajo Práctico Nº 1

Tema: Listas Simples

Apellido y Nombre: ..... Fecha:...../...../.....

EJERCICIOS

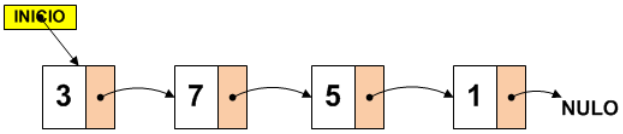
1) En base a la definición de *Lista Simple*, y considerando un único puntero al *inicio* de la lista, implemente sus operaciones fundamentales de teniendo en cuenta lo siguiente:
   - Una lista requiere de elementos, llamados nodos, que almacenen datos y que posean un indicador del próximo elemento de la lista.
   - Una operación de inicialización que permita crear (inicializar) una lista vacía.
   - Una operación que permita crear nodos.
   - Una operación de inserción que permita agregar un nuevo nodo al inicio de la lista.
   - Una operación de inserción que permita agregar un nuevo nodo al final de la lista.
   - Una operación de inserción que permita agregar, en orden, un nuevo nodo a la lista.
   - Una operación que extraiga un nodo del inicio de la lista.
   - Una operación que extraiga un nodo del final de la lista.
   - Una operación que extraiga un nodo específico (según un valor ingresado por el usuario) de la lista.
   - Una operación que permita buscar un nodo (valor) en la lista.
   - Una operación que permita mostrar el contenido de la lista.
2) Modifique la definición y operaciones básicas de listas simples (desarrolladas en el ítem 1) de modo que se incluya un elemento para registrar la cantidad de datos de la estructura.
3) Dada una lista de valores enteros, con un único puntero de inicio, realice lo siguiente:
   a) Consigne la declaración de tipos y variables de la estructura.
   b) Diseñe un procedimiento/función que permita *agregar* un nuevo elemento a la lista por el *inicio* o el *final* según un parámetro de opción.
   c) Diseñe un procedimiento/función que reemplace los valores positivos de la lista por su correspondiente valor factorial. Considere que el cálculo de factorial se realiza mediante un algoritmo *recursivo*.
4) Dada una lista de caracteres, con un único puntero de inicio, realice lo siguiente:
   a) Consigne la declaración de tipos y variables de la estructura.
   b) Diseñe un procedimiento/función que permita *quitar* un elemento de la lista por el *inicio* o el *final* según un parámetro de opción.
   c) Diseñe un procedimiento/función que convierta las minúsculas de la lista a mayúsculas.
5) Dada una lista de valores enteros
   a) Consigne la declaración de tipos y variables de la estructura.
   b) Diseñe un procedimiento/función que permita determinar el máximo valor de la lista.
   c) Diseñe un procedimiento/función que permita calcular el promedio de los valores positivos.
6) Dadas las siguientes listas:

   a) defina las estructuras de datos que permitan representarlas e
   b) implemente la operación *mezcla* que genere una nueva lista con el contenido de ambas, ordenado en forma creciente. Considere que las listas originales no deben modificarse.

7) Dada una colección de caracteres, se pretende ordenar dicha colección aplicando el algoritmo de *ordenación por Selección*. El algoritmo a implementar sólo debe realizar un intercambio en cada recorrido de la lista (por el mínimo carácter encontrado) para lograr la ordenación. Por ello, se solicita:
   a) Implemente mediante listas simples la colección de caracteres, y
   b) Desarrolle los procedimientos/funciones necesarios para implementar el algoritmo de ordenación *por Selección* para la estructura definida.

8) Dada una lista simple de caracteres, con un único puntero de *inicio*, defina la estructura de datos correspondiente y desarrolle un algoritmo que genere una nueva lista con las minúsculas de la lista original. Considere que la primera lista está ordenada de forma decreciente, mientras que la segunda debe ordenarse de forma creciente. Tenga en cuenta que la lista original no se modifica.
   Nota: PUEDE UTILIZAR LAS OPERACIONES BÁSICAS DE LISTA (EXCEPTO AGREGAR_ORDEN) EN LA SOLUCIÓN PLANTEADA.

9) Dada la siguiente lista:



   a) defina las estructuras de datos que permitan representarla e
   b) implemente las operaciones *agregar_final, mostrar_lista* y *sumar_nodos* todas ellas de forma **recursiva**.

10) Analice los siguientes fragmentos de código, describa las acciones que realizan y determine sus propósitos. Utilice la lista que se muestra a continuación para probar los algoritmos:

```
void enigma (pnodo lista)
{ if (lista!=NULL)
  { if (lista->sig==NULL)
      cout << lista->dato << endl;
    else
     { enigma(lista->sig);
       cout << lista->dato << endl
     }
  }
}

int misterio (pnodo lista)
{ if (lista==NULL)
    return 0;
  else
    return misterio(lista->sig) + 1;
}

pnodo desconocido (pnodo lista)
{ if (lista==NULL || lista->sig==NULL)
    return lista;
   else
    return desconocido(lista->sig);
}
```