15-9-2021

Lista estatica

Practica: 2

Materia: Estructura de datos

Sección: D01.

Código: 216584703

Carrera: Ingeniería en computación.

Nombre alumno: Padilla Pérez Jorge Daray

Nombre profesor: Julio Esteban Valdes Lopez



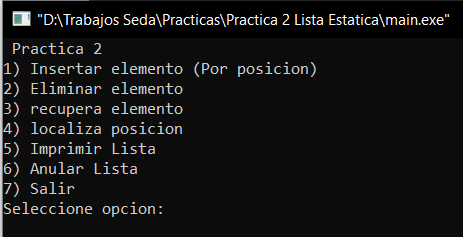


Introducción

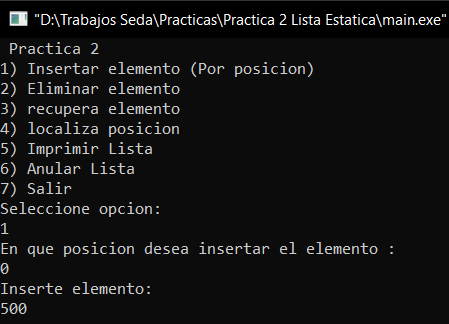
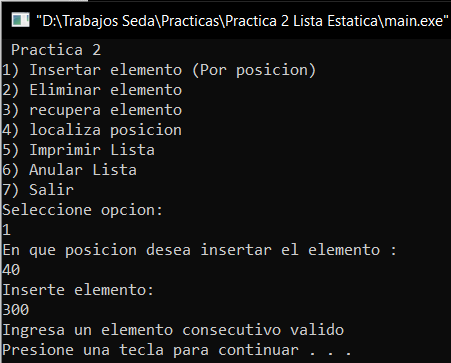
Mi práctica consiste en la implementación de una lista estructurada en la cual se implementan lo típico de un TDA Lista lo que viene siendo insertar en orden como si fuera una lista consultar elemento remover etc. En esta practica se logro hacer todo lo pedido para esta, además de poder realizar la función de localizar posición por elemento la cual me costó más, pero si se logró.

También se pudo lograr recorrer los datos una vez ingresados en la lista, aunque no se implementaron las funciones anterior y siguiente creo que es una función fácil de comprender y de hacer por lo que me parece bien que se hayan saltado esas 2 funciones que en si no existe una utilidad buena para ellos.

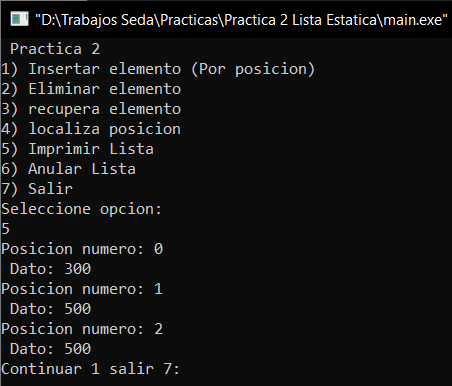
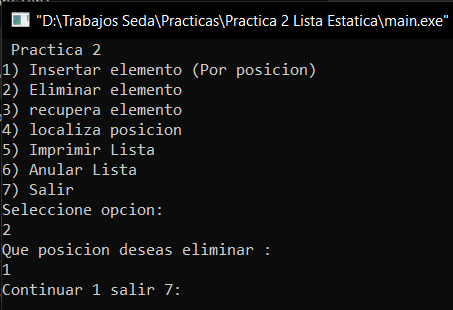
Pantallazos

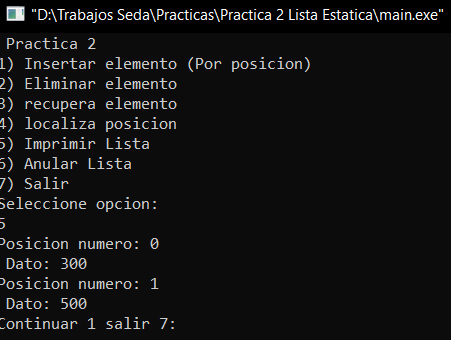


Aquí se aprecia el menú para el usuario la verdad que no se explica mucho más que ahí se ven todas las funciones del programa.

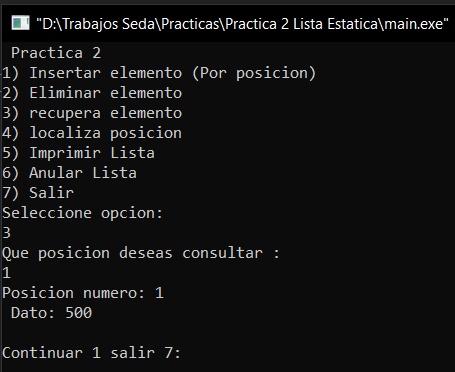
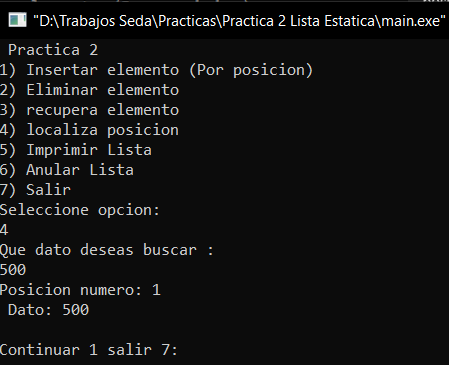
  
Aquí insertamos en la posición 0 ya que al ser una lista no podemos insertar más allá del primer índice, claro si es que esta vacia.

También a la derecha se ve que pasa si se inserta en otra posición que no es la correcta.

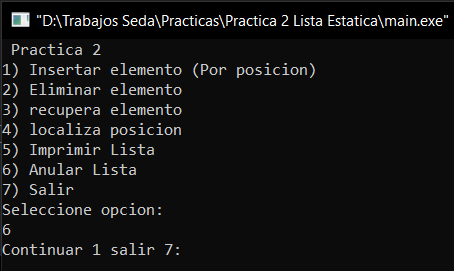
La función imprimir no tiene mas chiste que imprimir los datos registrados.

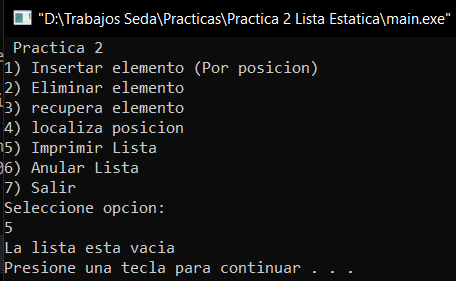
Si eliminamos la posición 1 en esta caso 500 al imprimirlo queda en la posición 2 el 500 pero la posición 3 desaparece.

Aquí se aprecia lo anterior.

aquí recuperamos el elemento por posición en este caso buscamos la posición 1 y no arroja el dato 500 almacenado ahí

En este caso parece igual a recupera, pero aquí se localiza la posición a través del dato en este caso pide el dato yo pongo 500 y como ya sabíamos está en la posición 1 que ahí aparece.

Al momento de anular la lista pues se eliminan todos los índices y lo que contenían estos mismos.



Aquí se aprecia mejor eso de anular la lista la cual al quererla imprimirla arroja el mensaje la lista esta vacia. .

Conclusión

Respecto a la realización del código concluyo que se logró bien la realización de este programa ya que a lo que se puede apreciar funciona de manera correcta como debería hacerlo una lista además de implementar bien sus funciones recalco mis conocimientos, y estoy abierto a posibles errores que pueda tener el programa al momento de que el profe la evalúe.

También se concluye que aunque no parezca que tiene una utilidad buena al momento de referirse a trabajo si lo piensas un rato le hayas muchas utilidades importantes tanto que pensándolo muchas aplicaciones y juegos realizan una lista implementada en estos mismos por lo cual espero poder aprender mas sobre las listas y los tipos de datos abstractos

**Código fuente**

**#include <iostream>**

**#include <cstring>**

**#define TAMLISTA 10**

**using namespace std;**

**void menu();**

**typedef int tipo\_dato;**

**struct Lista{**

**tipo\_dato datos[TAMLISTA];**

**void inicializa();**

**bool vacia();**

**bool llena();**

**void insertar(int pos, tipo\_dato elem);**

**void elimina(int pos);**

**int ultimo;**

**int primero;**

**void localiza(tipo\_dato dato);**

**void recupera(int pos);**

**void imprimir();**

**void anular();**

**Lista(){**

**inicializa();**

**}**

**};**

**void Lista::inicializa(){**

**ultimo = -1;**

**primero = 0;**

**}**

**bool Lista::vacia(){**

**return ultimo == -1;**

**}**

**bool Lista::llena(){**

**return ultimo == TAMLISTA - 1;**

**}**

**void Lista::insertar(int pos, tipo\_dato elem){**

**if (llena() || pos < 0 || pos > ultimo + 1){**

**cout<<"Ingresa un elemento consecutivo valido"<<endl;**

**system("pause");**

**return;**

**}**

**for(int i = ultimo+1 ; i > pos ; i-- ){**

**datos[i] = datos[i - 1];**

**}**

**datos[pos] = elem;**

**ultimo++;**

**}**

**void Lista::elimina(int pos){**

**if (vacia() || pos < 0 || pos > ultimo ){**

**cout<<"La lista esta vacia"<<endl;**

**system("pause");**

**return;**

**}**

**for (int i = pos ; i < ultimo ; i++){**

**datos[i] = datos[i + 1];**

**ultimo--;**

**}**

**}**

**void Lista::imprimir(){**

**if (vacia()){**

**cout<<"La lista esta vacia"<<endl;**

**system("pause");**

**return;**

**}**

**for(int i = primero ; i <= ultimo ; i++){**

**cout<<"Posicion numero: "<<i<<"\n Dato: "<<datos[i]<< " "<<endl;**

**}**

**}**

**void Lista::recupera(int pos){**

**if (vacia() || pos < 0 || pos > ultimo ){**

**cout<<"La lista esta vacia"<<endl;**

**system("pause");**

**return;**

**}**

**cout<<"Posicion numero: "<<pos<<"\n Dato: "<<datos[pos]<<" "<<endl;**

**cout<<endl;**

**}**

**void Lista::localiza(tipo\_dato dato){**

**if (vacia()){**

**cout<<"La lista esta vacia"<<endl;**

**system("pause");**

**return;**

**}**

**int i;**

**for (i=primero;i<=ultimo;i++){**

**if (datos[i] == dato){**

**cout<<"Posicion numero: "<<i<<"\n Dato: "<<datos[i]<<" "<<endl;**

**cout<<endl;**

**}**

**}**

**}**

**void Lista::anular(){**

**ultimo = -1;**

**}**

**struct Lista mi\_lista;**

**int main(){**

**int opc=0;**

**do{**

**system("cls");**

**menu();**

**cout<<"Continuar 1 salir 7:"<<endl;cin>>opc;**

**}while(opc!=7);**

**system("pause>>cls");**

**return 0;**

**}**

**void menu(){**

**int opc=0;**

**int dato,pos;**

**cout<<" Practica 2"<<endl;**

**cout<<"1) Insertar elemento (Por posicion)"<<endl;**

**cout<<"2) Eliminar elemento"<<endl;**

**cout<<"3) recupera elemento"<<endl;**

**cout<<"4) localiza posicion"<<endl;**

**cout<<"5) Imprimir Lista"<<endl;**

**cout<<"6) Anular Lista"<<endl;**

**cout<<"7) Salir"<<endl;**

**cout<<"Seleccione opcion:"<<endl;**

**cin>>opc;**

**switch(opc){**

**case 1: {**

**cout<<"En que posicion desea insertar el elemento :"<<endl;cin>>pos;**

**cout<<"Inserte elemento: "<<endl;cin>>dato;**

**mi\_lista.insertar(pos,dato);**

**}break;**

**case 2: {**

**cout<<"Que posicion deseas eliminar :"<<endl;cin>>pos;**

**mi\_lista.elimina(pos);**

**}break;**

**case 3: {**

**cout<<"Que posicion deseas consultar :"<<endl;cin>>pos;**

**mi\_lista.recupera(pos);**

**}break;**

**case 4: {**

**cout<<"Que dato deseas buscar :"<<endl;cin>>dato;**

**mi\_lista.localiza(dato);**

**}break;**

**case 5: {**

**mi\_lista.imprimir();**

**}break;**

**case 6: {**

**mi\_lista.anular();**

**}break;**

**case 7:break;**

**default:**

**cout<<"La opcion: "<<opc<<"No existe"<<endl;**

**}**

**}**