|  |  |  |
| --- | --- | --- |
|  | UNIVERSIDAD NACIONAL EXPERIMENTAL POLITÉCNICA  “ANTONIO JOSÉ DE SUCRE”  VICERECTORADO PUERTO ORDAZ  DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA  SECCIÓN DE COMPUTACIÓN – **COMPUTACION III** | logo_departamentologo_departamentologo_departamento |

**PRACTICA N° 3**

|  |
| --- |
| Objetivos. |

Aprender los conceptos básicos del uso de Listas simples enlazadas, como estructuras de datos dinámicas.

|  |
| --- |
| Base Teórica. |

**Manipuladores.-** Son comandos que se utilizan para organizar el despliegue de los resultados en la pantalla. Para utilizar los manipuladores se requiere la librería **iomanip.h**.

En el programa que se muestra se envían a pantalla dos valores **I/X**, uno entero y otro real, separados con espacios en blanco (**cout<<I<<" "<<X<<endl;)**. No se utilizan manipuladores. Luego se muestra como se utilizan los manipuladores y cuál es el efecto en la salida.

|  |  |  |
| --- | --- | --- |
|  |  | En la salida que se muestra a la izquierda se observa que no existe formato. Hay cifras sin decimales, hay cifras con 2 decimales, con 5 decimales, no hay alineación de los datos. |

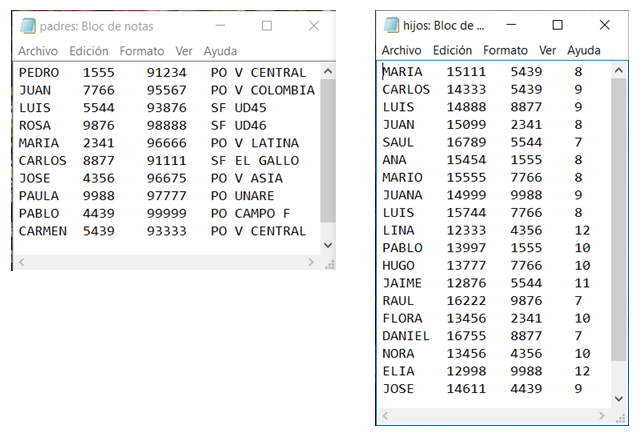
|  |  |  |
| --- | --- | --- |
| * **setw(n).** Usa **n** espacios, para ubicar X, alineado a la derecha.   **cout<<setw(2)<<I<<setw(10)<<X<<endl;** | | |
| * **setprecision(n).** Fija los (n) dígitos que van después del punto.   **cout<<setw(2)<<I<<setw(10)<<setprecision(3)<<X<<endl;** | | |
| * **fixed.** Hace que los valores tengan punto decimal.   **cout<<setw(2)<<I<<setw(10)<<fixed<<setprecision(3)<<X<<endl;** | | |
| A la derecha se muestra como sería la salida del programa que se muestra arriba utilizando los manipuladores. Los manipuladores deben colocarse antes de la variable que se va a mostrar. En el caso de ***setw***(n), si la variable tiene mas dígitos que n, el manipulador no tiene ningún efecto sobre la salida | |  |
| * **right/left**.- alinean la salida a la derecha (**right**) / izquierda (**left**). Por defecto **right**. | | |
| * Se coloca así: **cout<<left<<"LUNES"** * El efecto se mantiene hasta que se cambie la alineación. |  | |
| * Si se quiere texto alineado a la derecha, numero alineado a la izquierda se coloca: |  | |
|  | | |

1. **FUNCIONES PARA MANIPULACION DE LISTAS SIMPLES ENLAZADAS**

|  |  |
| --- | --- |
| **void INSERTAR\_INI(ptNODO &Lista, DATO Elem)**  **{ ptNODO NUEVO;**  **NUEVO=new NODO;**  **NUEVO->Info=Elem; NUEVO->Sig=NULL;**  **if(Lista==NULL) Lista=NUEVO;**  **else**  **{ NUEVO->Sig=Lista;**  **Lista=NUEVO;**  **}**  **}** | **void INSERTAR\_FIN(ptNODO &Lista, DATO Elem)**  **{ ptNODO A = Lista;**  **ptNODO NUEVO;**  **NUEVO = new NODO;**  **NUEVO->Info = Elem; NUEVO->Sig = NULL;**  **if(Lista == NULL) Lista = NUEVO;**  **else**  **{ while (A->Sig != NULL) A = A->Sig;**  **A->Sig = NUEVO; [2]**  **}**  **}** |
| **void INSERTAR\_NODO(ptNODO &Lista, DATO X, int POS)**  **{ int i;**  **ptNODO A = Lista, NUEVO;**  **NUEVO = new NODO;**  **NUEVO ->Info = X; NUEVO ->Sig = NULL;**  **if(POS==1 || Lista==NULL)**  **{ NUEVO->Sig = Lista;**  **Lista = NUEVO;**  **}**  **else**  **{ i=1;**  **while(i<POS-1 && A->Sig != NULL)**  **{ i++;**  **A=A->Sig;**  **}**  **if(A->Sig != NULL)**  **{ NUEVO->Sig = A->Sig;**  **A->Sig = NUEVO;**  **}**  **else A->Sig=NUEVO;**  **}**  **}** | **void MOSTRAR\_LISTA (ptNODO Lista)**  **{ ptNODO A=Lista;**  **if (A != NULL)**  **{ cout<<"Inicio ->>";**  **while(A!= NULL)**  **{ cout<<A->Info;**  **A=A->Sig;**  **}**  **cout<<"<<- FINAL";**  **getch();**  **}**  **else cout<<"Lista Vacia";**  **}** |
| **ptNODO POS\_INSER(ptNODO Lista, DATO Elem)**  **{ ptNODO P=NULL,A=Lista;**  **if(A != NULL)**  **{ while(Elem>=A->Info && A->Sig!=NULL)**  **{ P=A;**  **A=A->Sig;**  **}**  **if(Elem>=A->Info) P=A;**  **}**  **return P;**  **}** |
| **ptNODO BUSCAR(ptNODO Lista, DATO elem)**  **{ ptNODO A=Lista;**  **if(Lista!=NULL)**  **{ while(A->Info!=elem && A->Sig!=NULL) A=A->Sig;**  **if(A->Info == elem) return A;**  **else return NULL;**  **}**  **else**  **{ cout<<"Lista Vacia "<<endl;**  **return NULL;**  **}**  **}** | **void INSERTAR\_ORD(ptNODO &Lista, DATO Elem)**  **{ ptNODO A, NUEVO;**  **NUEVO=new NODO;**  **NUEVO->Info=Elem; NUEVO->Sig=NULL;**  **if(Lista == NULL) Lista=NUEVO;**  **else**  **{ A=POS\_INSER(Lista,Elem);**  **if(A==NULL)**  **{ NUEVO->Sig=Lista;**  **Lista=NUEVO; }**  **else**  **{ NUEVO->Sig=A->Sig;**  **A->Sig=NUEVO;**  **}**  **}**  **}** |
| **void BORRAR(ptNODO &Lista, DATO X)**  **{ ptNODO Aux;**  **Aux=BUSCAR(Lista,X);**  **if(Aux!=NULL)**  **{ if(Aux==Lista) Lista=Lista->Sig;**  **else**  **{ ptNODO Ant=Lista;**  **while(Ant->Sig != Aux) Ant=Ant->Sig;**  **Ant->Sig=Aux->Sig;**  **}**  **delete(Aux);**  **}**  **else cout<<"No existe el dato buscado";**  **}** | **void BORRAR\_DIR(ptNODO &Lista, ptNODO &P)**  **{ if(P==Lista)**  **{ Lista=Lista->Sig;**  **delete(P);**  **}**  **else**  **{ ptNODO Ant=Lista;**  **while(Ant->Sig != P) Ant=Ant->Sig;**  **if(Ant->Sig == P)**  **{ Ant->Sig = P->Sig;**  **delete(P);**  **}**  **else cout<<"Nodo no existe";**  **}**  **}** |

1. **DESARROLLO DE LA PRACTICA.**

Se tienen dos archivos de texto, que serán suministrados con la práctica, que contienen los datos de los PADRES (NOMBRE, CEDULA, TELEFONO, DIRECCIÓN) e HIJOS (NOMBRE, CEDULA, CEDULA\_PADRE, EDAD).



Escribir un programa tipo MENU, usando listas simples enlazadas para almacenar los datos leídos de los archivos. El MENU debe incluir las siguientes opciones:

1. Cargar Datos
2. Mostrar Datos Cargados
3. Buscar el teléfono de un padre conociendo la cedula de un hijo
4. Buscar/Mostrar los hijos de un padre usando su número de cedula.
5. Opción extra
6. Salir.