Estructuras básicas (Listas)

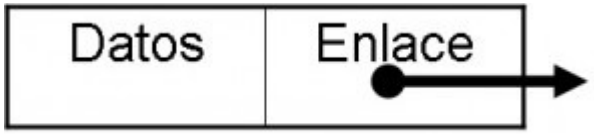
Profesor Francisco Alejandro Medina

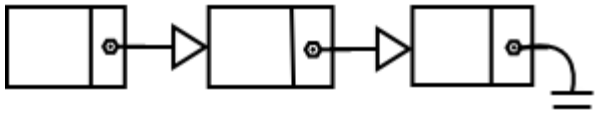
TDA Lista

- Una lista lineal es un conjunto de elementos de un tipo dado que se encuentran **ordenados** y pueden variar en número.
- Permite el recorrido de todos y cada uno de sus elementos, sin saltear ninguno y en forma **ordenada**.
- Ejemplo:
  - Guía telefónica
  - Lista de asistencia a un curso
  - Índice de un libro
  - Listado de compras
  - Listado de ingredientes de una receta

TDA Lista

- Una lista enlazada o encadenada es un conjunto de elementos mas un campo especial que contiene el puntero al elemento siguiente de la lista.
- Cada elemento de la lista debe tener al menos dos campos:
  - Elemento o dato.
  - Enlace, *link*, conexión al siguiente elemento.
- Los elementos de una lista son enlazados por medio de los campos enlaces.





Lista enlazada

TDA Lista

- **Operaciones básicas de la PILA**
- *Crear* inicializar una lista vacía.
- *Lista vacía* determinar si una lista esta vacía.
- *Lista llena* determina si la lista se ha llenado.
- *Insertar* inserta un elemento en la lista de forma que siga ordenada.
- *Buscar* busca un determinado elemento dentro de la lista.
- *Borrar* busca y elimina un elemento en la lista, manteniendo el orden.

Ejemplo 1

```
1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4  //-------------------------------------------------
5  struct nodo{
6         int nro;  // en este caso es un numero entero
7         struct nodo *sgte;
8  };
9  typedef struct nodo *Tlista;
10 //-------------------------------------------------
11 void insertarInicio(Tlista &lista, int valor)
12 {
13     Tlista q;
14     q = new(struct nodo);
15     q->nro = valor;
16     q->sgte = lista;
17     lista  = q;
18 }
19 //-------------------------------------------------
```

Ejemplo 1

```
19  //-------------------------------------------------------
20  void insertarFinal(Tlista &lista, int valor)
21  {
22      Tlista t, q = new(struct nodo);
23      q->nro  = valor;
24      q->sgte = NULL;
25      if(lista==NULL)
26      {
27          lista = q;
28      }
29      else
30      {
31          t = lista;
32          while(t->sgte!=NULL)
33          {
34              t = t->sgte;
35          }
36          t->sgte = q;
37      }
38  }
39  //-------------------------------------------------------
```

Ejemplo 1

```
39  //-------------------------------------------------------
40  int insertarAntesDespues()
41  {
42      int _op, band;
43      cout<<endl;
44      cout<<"\t 1. Antes de la posicion    "<<endl;
45      cout<<"\t 2. Despues de la posicion  "<<endl;
46      cout<<"\n\t Opcion : "; cin>> _op;
47      if(_op==1)
48          band = -1;
49      else
50          band = 0;
51      return band;
52  }
53  //-------------------------------------------------------
54  void insertarElemento(Tlista &lista, int valor, int pos)
55  {
56      Tlista q, t;
57      int i;
58      q = new(struct nodo);
59      q->nro = valor;
```

Ejemplo 1

```
60      if(pos==1)
61      {
62          q->sgte = lista;
63          lista = q;
64      }
65      else
66      {
67          int x = insertarAntesDespues();
68          t = lista;
69          for(i=1; t!=NULL; i++)
70          {
71              if(i==pos+x)
72              {
73                  q->sgte = t->sgte;
74                  t->sgte = q;
75                  return;
76              }
77              t = t->sgte;
78          }
79      }
80      cout<<"   Error...Posicion no encontrada..!"<<endl;
81  }
82  //-------------------------------------------------
```

Ejemplo 1

```
82  //-------------------------------------------------
83  void buscarElemento(Tlista lista, int valor)
84  {
85      Tlista q = lista;
86      int i = 1, band = 0;
87      while(q!=NULL)
88      {
89          if(q->nro==valor)
90          {
91              cout<<endl<<" Encontrada en posicion "<< i <<endl;
92              band = 1;
93          }
94          q = q->sgte;
95          i++;
96      }
97      if(band==0)
98          cout<<"\n\n Numero no encontrado..!"<< endl;
99  }
100 //-------------------------------------------------
```

Ejemplo 1

```
101  void MostrarLista(Tlista lista)
102  {
103       int i = 0;
104
105       while(lista != NULL)
106       {
107            cout <<' '<< i+1 <<") " << lista->nro << endl;
108            lista = lista->sgte;
109            i++;
110       }
111  }
112  //-------------------------------------------------------
113  void eliminarElemento(Tlista &lista, int valor)
114  {
115      Tlista p, ant;
116      p = lista;
117      if(lista!=NULL)
118      {
119          while(p!=NULL)
120          {
121              if(p->nro==valor)
122              {
123                  if(p==lista)
124                      lista = lista->sgte;
```

Ejemplo 1

```
125                     else
126                         ant->sgte = p->sgte;
127                     delete(p);
128                     return;
129                 }
130                 ant = p;
131                 p = p->sgte;
132             }
133         }
134         else
135             cout<<" Lista vacia..!";
136 }
137 //-----------------------------------------------
```

Ejemplo 1

```
138  void eliminaRepetidos(Tlista &lista, int valor)
139  {
140      Tlista q, ant;
141      q = lista;
142      ant = lista;
143      while(q!=NULL)
144      {
145          if(q->nro==valor)
146          {
147              if(q==lista) // primero elemento
148              {
149                  lista = lista->sgte;
150                  delete(q);
151                  q = lista;
152              }
153              else
154              {
155                  ant->sgte = q->sgte;
156                  delete(q);
157                  q = ant->sgte;
158              }
159          }
```

Ejemplo 1

```
160             else
161             {
162                 ant = q;
163                 q = q->sgte;
164             }
165         }// fin del while
166         cout<<"\n\n Valores eliminados..!"<<endl;
167     }
168     //-------------------------------------------------
169     void menu1()
170     {
171         cout<<"\n\t\tLISTA ENLAZADA SIMPLE\n\n";
172         cout<<" 1. INSERTAR AL INICIO                "<<endl;
173         cout<<" 2. INSERTAR AL FINAL                 "<<endl;
174         cout<<" 3. INSERTAR EN UNA POSICION          "<<endl;
175         cout<<" 4. MOSTRAR LISTA                    "<<endl;
176         cout<<" 5. BUSCAR ELEMENTO                   "<<endl;
177         cout<<" 6. ELIMINAR ELEMENTO 'V'             "<<endl;
178         cout<<" 7. ELIMINAR ELEMENTOS CON VALOR 'V' "<<endl;
179         cout<<" 8. SALIR                             "<<endl;
180          cout<<"\n INGRESE OPCION: ";
181     }
182     /*-------------------------------------------------------------*/
```

Ejemplo 1

```
182  /*-------------------------------------------------------------*/
183  //     Funcion Principal
184  int main()
185  {
186      Tlista lista = NULL;
187      int op;      // opcion del menu
188      int _dato;   // elemenento a ingresar
189      int pos;     // posicion a insertar
190      system("color 0b");
191      do
192      {
193          menu1();
194          cin>> op;
195          switch(op)
196          {
197              case 1:    cout<< "\n NUMERO A INSERTAR: ";
198                         cin>> _dato;
199                         insertarInicio(lista, _dato);
200                         break;
201              case 2:    cout<< "\n NUMERO A INSERTAR: ";
202                         cin>> _dato;
203                         insertarFinal(lista, _dato );
204                         break;
```

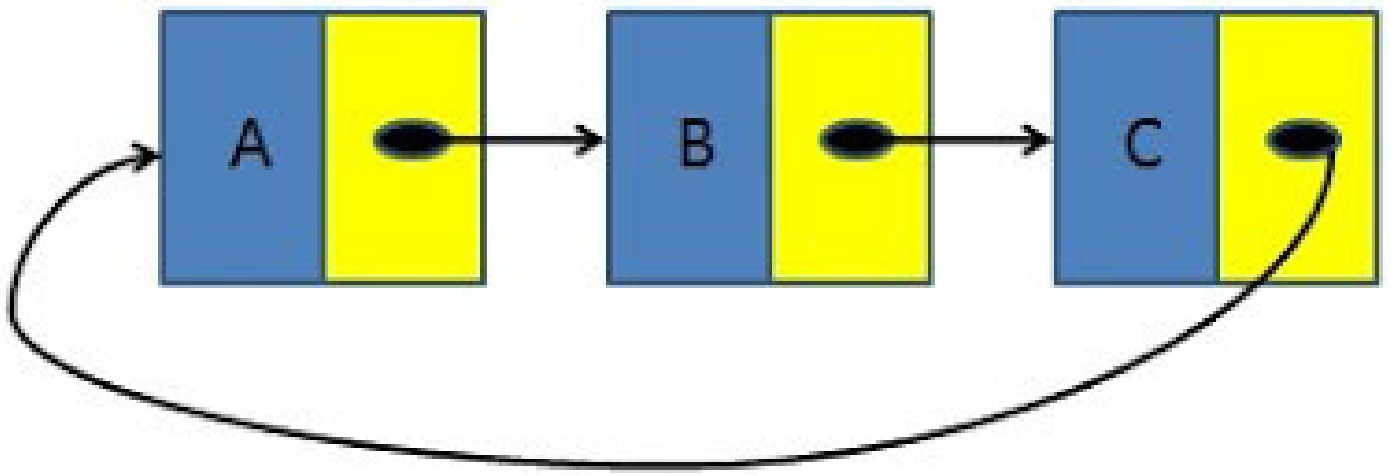
Ejemplo 1

```
205             case 3:   cout<< "\n NUMERO A INSERTAR: ";
206                       cin>> _dato;
207                       cout<< " Posicion : ";
208                       cin>> pos;
209                       insertarElemento(lista, _dato, pos);
210                       break;
211             case 4:   cout << "\n\n MOSTRANDO LISTA\n\n";
212                       MostrarLista(lista);
213                       break;
214             case 5:   cout<<"\n Valor a buscar: ";
215                       cin>> _dato;
216                       buscarElemento(lista, _dato);
217                       break;
218             case 6:   cout<<"\n Valor a eliminar: ";
219                       cin>> _dato;
220                       eliminarElemento(lista, _dato);
221                       break;
222             case 7:   cout<<"\n Valor repetido a eliminar: ";
223                       cin>> _dato;
224                       eliminaRepetidos(lista, _dato);
225                       break;
226         }
```

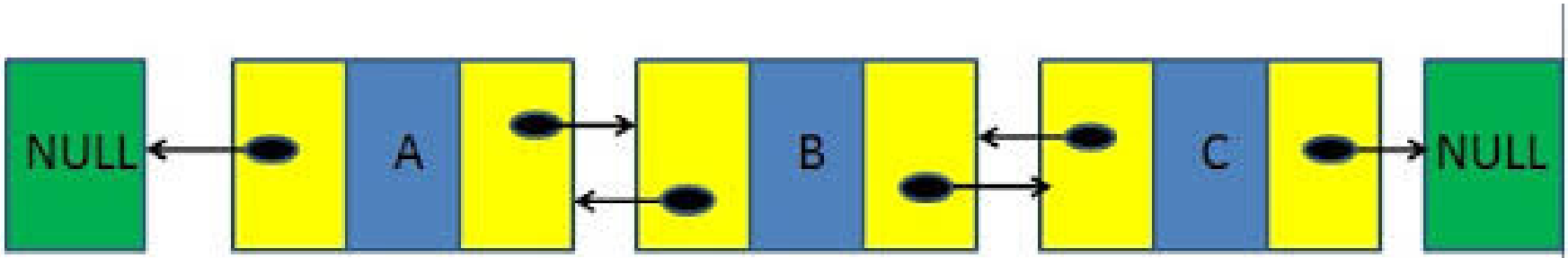
Ejemplo 1

```
227             cout<<endl<<endl;
228             system("pause");
229             system("cls");
230         }while(op!=8);
231         system("pause");
232         return 0;
233     }
```

Listas Circulares Simples



Listas Doblemente Enlazadas



Listas Circulares Dobles