

18-3-2021



# Lista estática

Practica: 2

Materia: Seminario de estructura de datos 1

Sección: D13.

Código: 216584703

Carrera: Ingeniería en computación.

Nombre alumno: Padilla Pérez Jorge Daray

Nombre profesor: Julio Esteban Valdes Lopez

## Índice general

Índice .....	1
Introducción .....	2
Pantallazos .....	3-6
Conclusión .....	7
Código fuente .....	8-13

# Introducción

Mi práctica consiste en la implementación de una lista estática en lenguaje c++, en donde se puede apreciar la integración de la frase: divide y vencerás, además de introducir el tema de las listas y el usar posiciones para agregar y quitar cosas, mediante estos procesos se tomó en cuenta la implementación de una función de insertar, eliminar y mostrar.

Al realizar mi practica obtuve como resultado la correcta integración e implementación de la lista estática dentro del código, en el cual es necesario asegurar que cada llamada a la lista debe estar definida sobre un problema menos complejo que el que dio lugar a la llamada, , como el que yo diseñe, al cual hago referencia en la página 12 logrando obtener un mejor conocimiento del tema.

# Pantallazo

```

1  #include <iostream>
2  #include <string>
3  #include <cstdlib>
4  #include <string>
5
6  #define TAMLISTA 10
7
8  using namespace std;
9
10 void menu();
11
12 typedef int tipo_dato;
13
14 struct Lista{
15     tipo_dato datos[TAMLISTA];
16     int ultimo;
17     void inicializa();
18     bool vacia();
19     bool llena();
20     void imprimir();
21     void insertar(int pos, tipo_dato elem);
22     void elimina(int pos);
23     void consultar(int pos);
24     void anular();
25
26     Lista(){
27         inicializa();
28     }
29 };

```

aquí se aprecia la inicialización de las funciones de forma diferente a como venimos manejando pero que funciona de buena manera también.

Aquí se mira donde se les dan los parámetros adecuados a las funciones para su correcto funcionamiento.

```

65 void Lista::elimina(int pos){
66     if (vacia() || pos < 0 || pos > ultimo){
67
68         cout<<"La lista esta vacia"<<endl;
69         system("pause");
70
71         return;
72     }
73     int i;
74     i = pos;
75     while (i < ultimo){
76         datos[i] = datos[i + 1];
77
78         i--;
79     }
80     ultimo--;
81 }
82
83 void Lista::imprimir(){
84     for(int i = 0 ; i <= ultimo ; i++){
85
86         cout<<"NO."<<i<<" : "<<datos[i]<<" "<<endl;
87
88     }
89 }
90

```

```

32 void Lista::inicializa(){
33     ultimo = -1;
34 }
35
36 bool Lista::vacia(){
37     return ultimo == -1;
38 }
39
40 bool Lista::llena(){
41     return ultimo == TAMLISTA - 1;
42 }
43
44 void Lista::insertar(int pos, tipo_dato elem){
45
46     if (llena() || pos < 0 || pos > ultimo + 1){
47
48         cout<<"Ingresa un elemento consecutivo valido"<<endl;
49         system("pause");
50         return;
51     }
52
53     for(int i = ultimo+1 ; i > pos ; i-- ){
54
55         datos[i] = datos[i - 1];
56     }
57
58     datos[pos] = elem;
59
60     ultimo = ultimo+1;
61 }
62

```

```

92 void Lista::consultar(int pos){
93     cout<<"NO. "<<pos<<" : "<<datos[pos]<<" "<<endl;
94     cout<<endl;
95 }
96
97 void Lista::anular(){
98     ultimo = -1;
99 }
100

```

```

102 struct Lista mi_lista;
103 int main() {
104     int opc=0;
105     do{
106         system("cls");
107         menu();
108         cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;
109     }while(opc!=2);
110     system("pause>>cls");
111     return 0;
112 }

```

Implementación de la variable `mi_lista` derivada de la estructura `Lista` para mejor entendimiento en código después , además de implementar el menú y un sistema para preguntar si se desea salir del programa o continuar para no cerrar el programa de golpe o por error.

```

114 void menu() {
115     int opc=0;
116     int dato,pos,dato,ultimo;
117     cout<<" Practica 2"<<endl;
118     cout<<"1) Insertar elemento (Por posicion)"<<endl;
119     cout<<"2) Eliminar elemento"<<endl;
120     cout<<"3) Consultar elemento"<<endl;
121     cout<<"4) Imprimir Lista"<<endl;
122     cout<<"5) Anular Lista"<<endl;
123     cout<<"6) Salir"<<endl;
124     cout<<"Seleccione opcion:"<<endl;
125     cin>>opc;
126     switch(opc) {

```

Aquí es donde ya se implementa el menú que vera el usuario mas el inicio de un switch para que se mande a llamar mejor lo que ordene el usuario

Aquí como lo dejamos se inicializa el switch con sus respectivas opciones en las cuales dependiendo lo que ingrese el usuario abrirá una opción u otra, en la cual simplemente se mandan a llamar las funciones anteriormente hechas con el simple hecho de mandar a pedir información y guardarla de manera adecuada.

```


126     switch(opc) {
127
128         case 1: {
129             cout<<"En que posicion desea insertar el elemento :"<<endl;cin>>pos;
130             cout<<"Inserte elemento: "<<endl;cin>>dato;
131             mi_lista.insertar(pos,dato);
132
133             }break;
134         case 2: {
135             cout<<"Que posicion deseas eliminar :"<<endl;cin>>pos;
136             mi_lista.elimina(pos);
137             }break;
138         case 3: {
139             cout<<"Que posicion deseas consultar :"<<endl;cin>>pos;
140             mi_lista.consultar(pos);
141             }break;
142
143         case 4: {
144             mi_lista.imprimir();
145             }break;
146
147         case 5: {
148             mi_lista.anular();
149             }break;
150
151     }
152 }

```

 "D:\Trabajos Seda\Lista Estatica\main.exe"


```
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
```

Aquí se aprecia el menú que debe tener el cual ve el usuario final


 "D:\Trabajos Seda\Lista Estatica\main.exe"

```
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
1
En que posicion desea insertar el elemento :
0
Inserte elemento:
1
Continuar 1 salir 2:
```

Aquí insertamos el primer elemento posición 0 y es el elemento llamado 1 ya que si ponemos una posición que no siga con la secuencia da error como el que se muestra a la derecha


 "D:\Trabajos Seda\Lista Estatica\main.exe"

```
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
1
En que posicion desea insertar el elemento :
4
Inserte elemento:
2
Ingresa un elemento consecutivo valido
Presione una tecla para continuar . . .
```

 "D:\Trabajos Seda\Lista Estatica\main.exe"

```
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
1
En que posicion desea insertar el elemento :
1
Inserte elemento:
2
Continuar 1 salir 2:
```

Entonces insertamos varios elementos como los de la izquierda para probar las otras funciones

 "D:\Trabajos Seda\Lista Estatica\main.exe"

```
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
1
En que posicion desea insertar el elemento :
2
Inserte elemento:
3
Continuar 1 salir 2:
```

## | Pantallazos

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
4
NO.0: 1
NO.1: 2
NO.2: 3
Continuar 1 salir 2:
```

En primer lugar imprimimos los resultados con la opcion 4 en la cual se puede ver la posicion y el valor que tiene asignado.

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
3
Que posicion deseas consultar :
1
NO. 1: 2
Continuar 1 salir 2:
```

Ahora consultamos un elemento, el cual se hace mediante posición y dándole la posición te muestra el elemento como se aprecia ahí

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
2
Que posicion deseas eliminar :
2
Continuar 1 salir 2:
```

En este caso usamos la opción 2 eliminar y eliminamos la posición 2 en la cual estaba el 3 entonces al imprimir los elementos ya no aparece como se aprecia a la derecha.

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
4
NO.0: 1
NO.1: 2
Continuar 1 salir 2:
```

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
5
Continuar 1 salir 2:
```

Por ultimo se selecciona la opción 5 que es anular y anula la lista lo que se puede apreciar mejor del lado derecho a la hora de imprimirlo no aparece nada porque ya fue anulada

```
"D:\Trabajos Seda\Lista Estatica\main.exe"
Practica 2
1) Insertar elemento (Por posicion)
2) Eliminar elemento
3) Consultar elemento
4) Imprimir Lista
5) Anular Lista
6) Salir
Seleccione opcion:
4
Continuar 1 salir 2:
```

# Conclusión

Respecto a la realización del código concluyo que logré el entendimiento correcto del tema mediante la correcta implementación de esta, se entendió el tema de la lista estática tanto de su sintaxis, como la lógica para su implementación, también aprendí los fallos que puede dar a la hora de correr el código y por ende sus posibles soluciones.

Además de aprender la teoría necesaria para darme cuenta de lo funcional que es, y poder identificar cuando conviene utilizar la lista estática y si se puede resolver el problema utilizándola.



# 1 Código Fuente

```
2  #include <iostream>
3  #include <cstring>
4  #include <cstdlib>
5  #include <string>
6
7  #define TAMLISTA 10
8
9  using namespace std;
10
11 void menu();
12
13 typedef int tipo_dato;
14
15 struct Lista{
16     tipo_dato datos[TAMLISTA];
17     int ultimo;
18     void inicializa();
19     bool vacia();
20     bool llena();
21     void imprimir();
22     void insertar(int pos, tipo_dato elem);
23     void elimina(int pos);
24     void consultar(int pos);
25     void anular();
26
27     Lista(){
28         inicializa();
```

```
29  }
30  };
31
32
33  void Lista::inicializa(){
34      ultimo = -1;
35  }
36
37  bool Lista::vacía(){
38      return ultimo == -1;
39  }
40
41  bool Lista::llena(){
42      return ultimo == TAMLISTA - 1;
43  }
44
45
46  void Lista::insertar(int pos, tipo_dato elem){
47
48      if (llena() || pos < 0 || pos > ultimo + 1){
49
50          cout<<"Ingresa un elemento consecutivo valido"<<endl;
51          system("pause");
52          return;
53      }
54
55      for(int i = ultimo+1 ; i > pos ; i-- ){
56
57          datos[i] = datos[i - 1];
```

```
58
59 }
60
61 datos[pos] = elem;
62
63 ultimo = ultimo+1;
64 }
65
66
67 void Lista::elimina(int pos){
68     if (vacía() || pos < 0 || pos > ultimo ){
69
70         cout<<"La lista esta vacia"<<endl;
71         system("pause");
72
73         return;
74     }
75     int i;
76     i = pos;
77     while (i < ultimo){
78
79         datos[i] = datos[i + 1];
80
81         i--;
82     }
83     ultimo--;
84 }
85
86 void Lista::imprimir(){
```

```
87
88  for(int i = 0 ; i <= ultimo ; i++){
89
90  cout<<"NO."<<i<<": "<<datos[i]<< " "<<endl;
91
92  }
93  }
94
95  void Lista::consultar(int pos){
96  cout<<"NO. "<<pos<<": "<<datos[pos]<<" "<<endl;
97  cout<<endl;
98  }
99
100 void Lista::anular(){
101  ultimo = -1;
102  }
103
104
105 struct Lista mi_lista;
106 int main(){
107  int opc=0;
108  do{
109      system("cls");
110      menu();
111      cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;
112  }while(opc!=2);
113  system("pause>>cls");
114  return 0;
115  }
```

```
116
117
118 void menu(){
119     int opc=0;
120     int dato,pos,datos,ultimo;
121     cout<<" Practica 2"<<endl;
122     cout<<"1) Insertar elemento (Por posicion)"<<endl;
123     cout<<"2) Eliminar elemento"<<endl;
124     cout<<"3) Consultar elemento"<<endl;
125     cout<<"4) Imprimir Lista"<<endl;
126     cout<<"5) Anular Lista"<<endl;
127     cout<<"6) Salir"<<endl;
128     cout<<"Seleccione opcion:"<<endl;
129     cin>>opc;
130     switch(opc){
131
132         case 1: {
133             cout<<"En que posicion desea insertar el elemento :"<<endl;cin>>pos;
134             cout<<"Inserte elemento: "<<endl;cin>>dato;
135             mi_lista.insertar(pos,dato);
136
137             }break;
138         case 2: {
139             cout<<"Que posicion deseas eliminar :"<<endl;cin>>pos;
140             mi_lista.elimina(pos);
141             }break;
142         case 3: {
143             cout<<"Que posicion deseas consultar :"<<endl;cin>>pos;
144             mi_lista.consultar(pos);
```

```
145
146     }break;
147
148     case 4: {
149         mi_lista.imprimir();
150     }break;
151
152     case 5: {
153         mi_lista.anular();
154     }break;
155
156     }
157 }
```