

ESTRUCTURAS DE DATOS

Arreglo:

```
[*] array.cpp
1  #include <iostream>
2  #include <array>
3
4  using namespace std;
5
6  int main(int argc, char *argv[]){
7      array<int, 3> a = {8, 9, 10};
8
9      cout << a.size() << endl;
10     cout << a.front() << endl;
11     cout << a.back() << endl;
12
13     cout << a[0] << a[1] << a[2] << endl;
14
15     return 0;
16 }
17
```



Vector:

```
array.cpp  [*] vector.cpp  listaEnteros.cpp  lista.h
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int main(int argc, char *argv[]){
7      vector<int> v;
8      v.push_back(2);
9      v.push_back(3);
10     v.push_back(4);
11     v.push_back(5);
12     v.push_back(6);
13     v.push_back(7);
14
15     cout << v.size() << endl;
16     cout << v.front() << endl;
17     cout << v.back() << endl;
18
19     for(int i = 0; i < v.size();i++){
20         cout << v[i] << endl;
21     }
22
23     return 0;
24 }
```



Lista enteros:

```
array.cpp  [*] vector.cpp  lista.h  listaEnteros.cpp
1  #ifndef LISTA_H
2  #define LISTA_H
3  #include <iostream>
4
5  class ListaEnteros {
6  private:
7      int cuenta;
8      int capacidad;
9      int *items;
10     void agrandar();
11 public:
12     ListaEnteros(int capacidad);           // Constructores
13     ListaEnteros();
14     ListaEnteros(const ListaEnteros &otraLista);
15     ~ListaEnteros();                       // Destructor
16     bool estaVacia();
17     int tamano();
18     void insertar(int indice, int item);
19     void adjuntar(int item);
20     int obtener(int indice);
21     bool contiene(int item);
22     int remover(int indice);
23     void imprimirLista();
24 };
25
26 ListaEnteros::ListaEnteros(int capacidad){
27     this->cuenta = 0;
28     this->capacidad = capacidad;
29     this->items = new int[capacidad];
30 }
31
32 ListaEnteros::~ListaEnteros(){
33     delete[] items;
34 }
35
36 bool ListaEnteros::estaVacia(){
37     return this->cuenta == 0;
38 }
39
40 int ListaEnteros::tamano(){
41     return this->cuenta;
42 }
43
44 int ListaEnteros::obtener(int indice){
45     if(indice < 0 || indice >= this->cuenta) throw "Indice fuera de rango";
46     if(this->estaVacia()) throw "No se puede recuperar elementos de una lista vacia";
47     return this->items[indice];
48 }
```



Edificio de Atención al Estudiante, Nivel 3, 55 Av. Sur, Condominio Centro Roosevelt,
entre Av. Olímpica y Alameda Roosevelt, San Salvador, El Salvador, C.A.



jaleman@ufg.edu.sv



(503) 2209-2930

```
array.cpp  [*] vector.cpp  lista.h  listaEnteros.cpp
46         if(this->estaVacia()) throw "No se puede recuperar elementos de una lista vacia";
47         return this->items[indice];
48     }
49
50 void ListaEnteros::insertar(int indice, int item){
51     if(indice < 0 || indice > this->cuenta) throw "Indice fuera de rango";
52     if (this->cuenta >= this->capacidad) this->agrandar();
53
54     for(int i = cuenta - 1; i >= indice; i--){
55         this->items[i+1] = this->items[i];
56     }
57
58     this->items[indice] = item;
59     this->cuenta++;
60 }
61
62 void ListaEnteros::agrandar(){
63     int *temp = this->items;
64     this->capacidad *=2;
65     this->items = new int[capacidad];
66     for(int i =0; i < this->cuenta; i++){
67         this->items[i] = temp[i];
68     }
69     delete[] temp;
70 }
71
72 void ListaEnteros::adjuntar(int item){
73     this->insertar(this->cuenta, item);
74 }
75
76 bool ListaEnteros::contiene(int item){
77     for(int i = 0; i < this->cuenta; i++){
78         if(this->items[i] == item) return true;
79     }
80     return false;
81 }
82
83 int ListaEnteros::remove(int indice){
84     if(this->estaVacia()) throw "No se puede recuperar elementos de una lista vacia";
85     if(indice < 0 || indice >= this->cuenta) throw "Indice fuera de rango";
86
87     int valor = this->items[indice];
88     for(int i = indice; i < cuenta - 1; i++){
89         this->items[i] = this->items[i+1];
90     }
91     this->cuenta--;
92     return valor;
}
```



```
86  
87     int valor = this->items[indice];  
88     for(int i = indice; i < cuenta - 1; i++){  
89         this->items[i] = this->items[i+1];  
90     }  
91     this->cuenta--;  
92     return valor;  
93 }  
94  
95 void ListaEnteros::imprimirLista(){  
96     std::cout << " | ";  
97     for(int i = 0; i < this->cuenta; i++){  
98         std::cout << this->obtener(i) << " | ";  
99     }  
100     std::cout << std::endl;  
101 }  
102  
103 #endif
```

Implementación lista enteros:



Edificio de Atención al Estudiante, Nivel 3, 55 Av. Sur, Condominio Centro Roosevelt,
entre Av. Olímpica y Alameda Roosevelt, San Salvador, El Salvador, C.A.



jaleman@ufg.edu.sv



(503) 2209-2930

array.cpp [*] vector.cpp lista.h listaEnteros.cpp

```
1  #include "lista.h"
2  #include <iostream>
3
4  using namespace std;
5
6  int main(){
7
8      ListaEnteros lint1(10);
9
10     lint1.adjuntar(12);
11     lint1.adjuntar(7);
12     lint1.adjuntar(19);
13     lint1.imprimirLista();
14
15     lint1.adjuntar(30);
16     lint1.imprimirLista();
17
18     lint1.insertar(1,41);
19     lint1.imprimirLista();
20
21     lint1.remover(2);
22     lint1.imprimirLista();
23
24     return 0;
25 }
26
27
```

