Vectores

Rodolfo Christian Catunta Uturunco (Elemental Bolivia)

21 de marzo de 2025

Vectores

- La **clase** vector se encarga de generar estructuras de datos denominadas vectores.
- Los vectores son similares a los arreglos, con la ventaja de que tienen la capacidad de **cambiar de tamaño**.
- Por la razón anterior son usados en ocasiones como reemplazo a los arreglos estáticos.
- Un componente característico de los vectores es que sus elementos pueden ser accedidos de dos formas.
  - Por un indice
  - Por medio de un iterador

Constructores

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main(){
  vector <int> v1; // vector vacio (sin casillas)
  vector <int> v2(8); // vector con 8 casillas vacias
  vector <int> v3(5,10); // vector con 5 casillas cada una
    con el numero 10
  vector <int> v4(v2); // vector copia de v2
}
```

Acceso a Elementos

```cpp
// Sea un vector v
// v = [7, 4, -1, 8, 5, 0, 1]

// Acceso por indice
cout<<v[0]<<endl; // imprimira 7
int suma = v[2] + v[6]; // suma sera igual a 0

// Accesos especiales
int frente = v.front(); // frente sera igual a 7
int ultimo = v.ultimo(); // ultimo sera igual a 1
```

Funciones de Capacidad

```cpp
// Sea el vector v
// v = [8, 4, 1, -7, 10]

int tam = v.size(); // .size devuelve la cantidad de casillas del vector
// entonces tam sera igual a 5

if(v.empty()){// .empty devuelve un booleano
  // true si el vector esta vacio
  cout<<"v esta vacio"<<endl;
}
else{
  // false si el vector NO esta vacio
  cout<<"v no esta vacio"<<endl;
}
```

Iteradores

```
1  // Sea el vector v
2  vector<int>v;
3  // Con contenido
4  // v = [-8, 1, 2, 30, 14, 17]
5
6  //Creacion del iterador
7  vector<int >::iterador it;
8  it=v.begin();// Iterador que apunta al inicio del vector
9  it++;// Iterador que avanza una posicion
10 it--;// Iterador que retrocede una posicion
11 it=v.end();// Iterador que apunta al "final" del vector
```

Funciones Modificadoras

```
1  // Sea el vector v = [5,-1,7]
2  // Insertar al final
3  v.push_back(4); // v = [5,-1,7,4]
4  // Quitar del final
5  v.pop_back(); // v = [5,-1,7]
6  // Insertar en una posicion (iterador)
7  v.insert(v.begin(),3); // v = [3,5,-1,7]
8  // Borrar de una posicion (iterador)
9  v.erase(v.begin()++); // v = [3,-1,7]
10 // Cambiar de tamanio
11 v.resize(5,0); // v = [3,-1,7,0,0]
12
13 v.clear(); // Borra el contenido
```

Ordenamiento

```cpp
#include <iostream>
#include <vector>
#include <algorithm> // contiene a sort

using namespace std;

int main(){
  vector<int> v; // Sea v = [7,-8,1,4,0]
  // Orden creciente
  sort(v.begin(),v.end()); // v = [-8,0,1,4,7]
  // Orden decreciente
  sort(v.rbegin(),v.rend()); // v = [7,4,1,0,-8]
}
```

Búsqueda Binaria

```cpp
#include <iostream>
#include <vector>
#include <algorithm> // contiene a binary_search, lower_bound, upper_bound
using namespace std;
int main(){
  vector<int> v = {7,-8,1,4,0};
  // Ordenar primero
  sort(v.begin(),v.end()); // v = [-8,0,1,4,7]
  binary_search(v.begin(),v.end(),4); // true
  binary_search(v.begin(),v.end(),10); // false
  // lower_bound -> primer elemento no menor (>=)
  int pos = lower_bound(v.begin(),v.end(),1)-v.begin(); // 2
  pos = lower_bound(v.begin(),v.end(),5)-v.begin(); // 4
  // upper_bound -> primer elemento mayor (>)
  pos = upper_bound(v.begin(),v.end(),1)-v.begin(); // 3
}
```