

# Taller de vectores

Nombre: Juan David Torres Avila

Asignatura: Programación II (IS284)

Grupo: 5°

## 1. Inicialización Uniforme de Vector

Problema: Escribe un programa en C++ que cree un vector de enteros y lo inicialice con los Números del 1 al 10 utilizando inicialización uniforme. Imprime el vector resultante.

Entrada: No aplicable.

Salida: Imprime los elementos del vector, separados por espacio.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    vector<int> vec{1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
```

```
    for (const int &num : vec) {
```

```
        cout << num << " ";
```

```
    }
```

```
    return 0;
```

```
}
```

Resultado esperado:

```
1 2 3 4 5 6 7 8 9 10
```

## 2. Añadiendo Elementos con push\_back

Problema: Utiliza un vector vacío e implementa un bucle que utilice la función push\_back para

añadir los Números del 1 al 5. Imprime el vector final.

Entrada: No aplicable.

Salida: Los Números del 1 al 5 en una sola línea, separados por espacios.


Código: #include <iostream>

#include <vector>

using namespace std;

```
int main() {  
    vector<int> vec;  
    for (int i = 1; i <= 5; ++i) {  
        vec.push_back(i);  
    }  
    for (const int& num : vec) {  
        cout << num << " ";  
    }  
    cout << endl;  
  
    return 0;  
}
```

Resultado esperado:



```
1 2 3 4 5
```

### 3. Suma de Elementos de un Vector

Problema: Dado un vector de enteros inicializado, escribe un programa que calcule y muestre la

suma de todos sus elementos.

Entrada: La primera línea contiene N, el Número de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Un entero que representa la suma de los elementos del vector.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    int n;
```

```
    cout << "Ingrese el tamaño del vector: ";
```

```
    cin >> n;
```

```
    vector<int> vec(n);
```

```
    for (int i = 0; i < n; ++i) {
```

```
        cout << "Ingrese el elemento " << i + 1 << ": ";
```

```
        cin >> vec[i];
```

```
}
```

```
int sum = 0;
```

```
for (int i = 0; i < n; ++i) {
```

```
    sum += vec[i];
```

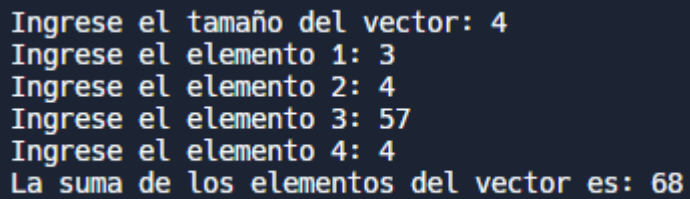
```
}
```

```
cout << "La suma de los elementos del vector es: " << sum << endl;
```

```
return 0;
```

```
}
```

Resultado esperado:



```
Ingrese el tamaño del vector: 4
Ingrese el elemento 1: 3
Ingrese el elemento 2: 4
Ingrese el elemento 3: 57
Ingrese el elemento 4: 4
La suma de los elementos del vector es: 68
```

#### 4. Duplicando Valores con el Operador []

Problema: Crea un vector de 5 elementos enteros. Utiliza un bucle para modificar cada elemento

del vector para que sea el doble de su índice. Imprime el vector resultante.

Entrada: No aplicable.

Salida: Los elementos del vector después de la modificación, separados por espacio.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    vector<int> vec(5);
```

```
    for (int i = 0; i < vec.size(); ++i) {
```

```
        vec[i] = 2 * i;
```

```
    }
```

```
    for (const int& num : vec) {
```

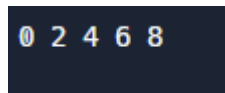
```
        cout << num << " ";
```

```
    }
```

```
    return 0;
```

```
}
```

Resultado esperado:

A screenshot of a terminal window with a dark background. The text "0 2 4 6 8" is displayed in a light blue or cyan monospaced font.

## 6. Impresión Utilizando size()

Problema: Escribe un programa que imprima los elementos de un vector, utilizando un bucle for

que se base en el tamaño del vector obtenido con el método .size().

Entrada: La primera línea contiene N, el Número de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Los elementos del vector en una sola línea, separados por espacios.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    int N;
```

```
    cout << "Ingrese el número de elementos: ";
```

```
    cin >> N;
```

```
    vector<int> vec(N);
```

```
    for (int i = 0; i < N; ++i) {
```

```
        cout << "Ingrese el elemento " << i + 1 << ": ";
```

```
        cin >> vec[i];
```

```
    }
```

```
    cout << "Los elementos del vector son: ";
```

```
    for (int i = 0; i < vec.size(); ++i) {
```

```
        cout << vec[i] << " ";
```

```
    }
```

```
    cout << endl;
```

```
    return 0;
```

```
}
```

Resultado esperado:

```
Ingrese el número de elementos: 5
Ingrese el elemento 1: 3
Ingrese el elemento 2: 44
Ingrese el elemento 3: 6
Ingrese el elemento 4: 7
Ingrese el elemento 5: 1
Los elementos del vector son: 3 44 6 7 1
```

## 7. Observando size() y capacity()

Problema: Crea un vector y añade elementos del 1 al 10 uno a uno. Después de cada adición,

imprime el size() y capacity() del vector para observar cómo cambian.

Entrada: No aplicable.

Salida: Para cada elemento añadido, imprime una línea con el size() y capacity() del vector, separados por un espacio.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    vector<int> vec;
```

```
    for (int i = 1; i <= 10; ++i) {
```

```
        vec.push_back(i);
```

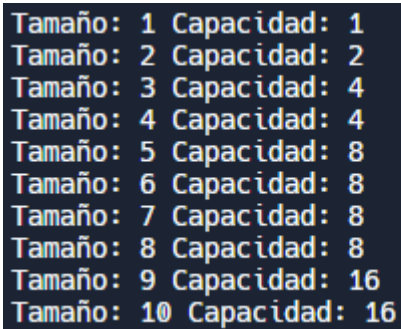
```
        cout << "Tamaño: " << vec.size() << " Capacidad: " << vec.capacity() << endl;
```

```
}
```

```
return 0;
```

```
}
```

Resultado esperado:



```
Tamaño: 1 Capacidad: 1
Tamaño: 2 Capacidad: 2
Tamaño: 3 Capacidad: 4
Tamaño: 4 Capacidad: 4
Tamaño: 5 Capacidad: 8
Tamaño: 6 Capacidad: 8
Tamaño: 7 Capacidad: 8
Tamaño: 8 Capacidad: 8
Tamaño: 9 Capacidad: 16
Tamaño: 10 Capacidad: 16
```

## 8. Buscar el Máximo Elemento

Problema: Dado un vector de enteros, escribe un programa que encuentre y muestre el valor

máximo dentro del vector.

Entrada: La primera línea contiene N, el Número de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Un entero que representa el valor máximo encontrado en el vector.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```



```
int N;

cout << "Ingrese el tamaño del vector: ";

cin >> N;

vector<int> vec(N);

for (int i = 0; i < N; ++i) {

    cout << "Ingrese el elemento " << i + 1 << ": ";

    cin >> vec[i];

}

int max_element = vec[0];

for (int i = 1; i < vec.size(); ++i) {

    if (vec[i] > max_element) {

        max_element = vec[i];

    }

}

cout << "El elemento máximo es: " << max_element << endl;

return 0;

}
```

Resultado esperado:

```
Ingrese el tamaño del vector: 4
Ingrese el elemento 1: 2
Ingrese el elemento 2: 578
Ingrese el elemento 3: 444
Ingrese el elemento 4: 5
El elemento máximo es: 578
```

## 9. Eliminar el Último Elemento con pop\_back

Problema: Crea un vector y añádele algunos elementos. Luego, elimina el último elemento usando

pop\_back. Imprime el vector antes y después de la eliminación.

Entrada: La primera línea contiene N, el Número inicial de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Primero, imprime el vector inicial en una línea. Después, imprime el vector después de

eliminar el último elemento en otra línea. Los elementos deben estar separados por espacios.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    int N;
```

```
    cout << "Ingrese el tamaño del vector: ";
```

```
    cin >> N;
```

```
    vector<int> vec(N);
```

```
    for (int i = 0; i < N; ++i) {
```

```
        cout << "Ingrese el elemento " << i + 1 << ": ";
```

```
        cin >> vec[i];
```

```
    }
```

```
for (const int& num : vec) {  
    cout << num << " ";  
}  
cout << endl;
```

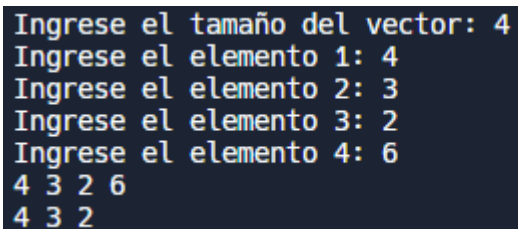
```
vec.pop_back();
```

```
for (const int& num : vec) {  
    cout << num << " ";  
}  
cout << endl;
```

```
return 0;
```

```
}
```

Resultado esperado:



```
Ingrese el tamaño del vector: 4  
Ingrese el elemento 1: 4  
Ingrese el elemento 2: 3  
Ingrese el elemento 3: 2  
Ingrese el elemento 4: 6  
4 3 2 6  
4 3 2
```

## 11. Uso de front y back

Problema: Dado un vector de enteros, imprime el primer y último elemento del vector utilizando

los métodos front() y back().

Entrada: La primera línea contiene N, el Número de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Dos enteros en una línea, el primer y último elemento del vector, separados por un espacio.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    int N;
```

```
    cout << "Ingrese el número de elementos: ";
```

```
    cin >> N;
```

```
    vector<int> vec(N);
```

```
    for (int i = 0; i < N; ++i) {
```

```
        cout << "Ingrese el elemento " << i + 1 << ": ";
```

```
        cin >> vec[i];
```

```
    }
```

```
    cout << "El primer y ultimo elemento son: " << vec.front() << " " << vec.back() << endl;
```

```
    return 0;
```

```
}
```

Resultado esperado:

```
Ingrese el número de elementos: 4
Ingrese el elemento 1: 3
Ingrese el elemento 2: 4
Ingrese el elemento 3: 8998
Ingrese el elemento 4: 3
El primer y ultimo elemento son: 3 3
```

### 13. Calcula la Media de los Elementos de un Vector

Problema: Dado un vector de enteros, calcula y muestra la media de sus elementos.

Entrada: La primera línea contiene N, el Número de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Un Número flotante que representa la media de los elementos del vector, redondeado a

dos decimales.

Código:

```
#include <iostream>
```

```
#include <vector>
```

```
#include <iomanip>
```

```
using namespace std;
```

```
int main() {
```

```
    int N;
```

```
    cout << "Ingrese el número de elementos: ";
```

```
    cin >> N;
```

```
    vector<int> vec(N);
```

```
    for (int i = 0; i < N; ++i) {
```

```

    cout << "Ingrese el elemento " << i + 1 << ": ";
    cin >> vec[i];
}

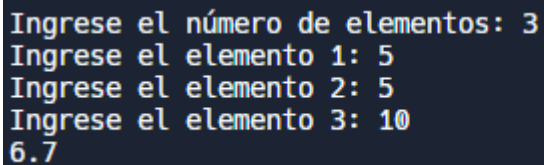
int sum = 0;
for (int i = 0; i < N; ++i) {
    sum += vec[i];
}

float promedio = static_cast<float>(sum) / N;
cout << setprecision(2) << promedio << endl;

return 0;
}

```

Resultado esperado:



```

Ingrese el número de elementos: 3
Ingrese el elemento 1: 5
Ingrese el elemento 2: 5
Ingrese el elemento 3: 10
6.7

```

#### 14. Uso de clear para Vaciar un Vector

Problema: Crea un vector y añádele algunos elementos. Utiliza el método clear para vaciarlo

completamente y muestra el tamaño del vector después de vaciarlo.

Entrada: La primera línea contiene N, el Número inicial de elementos del vector. La siguiente línea

contiene N enteros separados por espacio.

Salida: Un entero que representa el tamaño del vector después de utilizar clear().

Código: #include <iostream>

#include <vector>

```
using namespace std;
```

```
int main() {
```

```
    int N;
```

```
    cout << "Ingerese el tamaño del vector: ";
```

```
    cin >> N;
```

```
    vector<int> vec(N);
```

```
    for (int i = 0; i < N; ++i) {
```

```
        cout << "Ingrese el elemento " << i + 1 << ": ";
```

```
        cin >> vec[i];
```

```
    }
```

```
    vec.clear();
```

```
    cout << "El vector ha sido eliminado." << endl;
```

```
    cout << vec.size() << endl;
```

```
    return 0;
```

```
}
```

Resultado esperado:

```
Ingerese el tamaño del vector: 3
Ingrese el elemento 1: 54
Ingrese el elemento 2: 2
Ingrese el elemento 3: 5
El vector ha sido eliminado.
0
```