

1. Objetivo del código

Este programa en **C++** permite:

1. Pedir un número (x) que se buscará en un vector.
 2. Solicitar el tamaño del vector (n).
 3. Generar tres tipos de vectores:
 - **Vpeor**: Un vector donde todos los valores son x (peor caso para búsqueda).
 - **Vmejor**: Un vector con valores crecientes a partir de x.
 - **Vcualquiera**: Un vector con la mitad inicial llena de x y la otra mitad con valores crecientes desde x.
 4. Contar cuántas veces aparece x en cada uno de los tres vectores.
 5. Mostrar los resultados.
-

2. Explicación detallada del código

2.1. Inclusión de librerías y declaración de espacio de nombres

cpp

CopiarEditar

```
#include <iostream>
```

```
using namespace std;
```

- Se incluye la librería `iostream` para operaciones de entrada y salida (`cin` y `cout`).
 - Se usa `using namespace std;` para evitar escribir `std::` antes de `cout` y `cin`.
-

2.2. Función Contar

cpp

CopiarEditar

```
int Contar(int *V, int n, int x)
```

```
{
```

```
    int i, nveces;
```

```
    nveces=0;
```

```

for(i=1; i<=n; i++)
{
    if(V[i]==x)
        nveces=nveces+1;
}
return nveces;
}

```

- **Propósito:** Contar cuántas veces aparece x en el vector V.
- **Parámetros:**
 - V: Puntero a un array dinámico.
 - n: Tamaño del vector.
 - x: Valor a buscar.
- **Funcionamiento:**
 - Inicializa nveces en 0.
 - Recorre el vector desde la posición **1 hasta n** (evitando la posición 0).
 - Si V[i] == x, incrementa nveces.
 - Retorna nveces.

2.3. Función main

cpp

CopiarEditar

```

int main()
{
    int n, x;

    cout << "Introduce valor a buscar (x): ";
    cin >> x;
}

```

```
cout << "Introduce n.elementos del vector (n): ";
```

```
cin >> n;
```

- **Solicita al usuario:**
 1. El **valor x** a buscar.
 2. El **tamaño n** del vector.
-

2.4. Reserva de memoria dinámica para los vectores

cpp

CopiarEditar

```
int *Vpeor, *Vmejor, *Vcualquiera;
```

```
Vpeor= new int [n+1];
```

```
Vmejor= new int [n+1];
```

```
Vcualquiera= new int [n+1];
```

```
if(Vmejor == NULL || Vpeor == NULL || Vcualquiera == NULL)
```

```
{
```

```
    cout << "Error al reservar memoria, Reviselo" <<endl;
```

```
    return 0;
```

```
}
```

- **Reserva memoria dinámica** para los tres vectores de tamaño n+1 (ya que los índices empiezan en 1).
 - **Verifica si la reserva de memoria fue exitosa.**
 - Si alguna asignación falló (NULL), muestra un mensaje de error y termina el programa.
-

2.5. Generación y visualización del vector Vpeor

cpp

CopiarEditar

```
int i;
```

```
int resultado;

cout << "\nVector peor: " << endl;

for(i=1; i<=n; i++)
{
    Vpeor[i] = x;
}

for(i=1; i<=n; i++)
{
    cout << Vpeor[i] << ' ';
}
```

- **Genera el vector Vpeor:** Llena todas las posiciones con x.
 - **Lo muestra en pantalla.**
-

2.6. Conteo de x en Vpeor

cpp

CopiarEditar

```
resultado = Contar(Vpeor,n,x);

cout << endl;
```

- Llama a Contar(Vpeor, n, x), que devuelve cuántas veces x aparece en Vpeor.
-

2.7. Generación y visualización del vector Vmejor

cpp

CopiarEditar

```
cout << "Vector mejor: " << endl;

for(i=1; i<=n; i++)
{
```

```

    Vmejor[i] = x+i;
}
for(i=1; i<=n; i++)
{
    cout << Vmejor[i] << ' ';
}

```

- **Genera el vector Vmejor:** Cada posición i contiene $x + i$.
 - **Lo muestra en pantalla.**
-

2.8. Conteo de x en Vmejor

cpp

CopiarEditar

```

resultado = Contar(Vmejor,n,x);

cout << endl;

```

- Llama a Contar(Vmejor, n, x) para contar cuántas veces aparece x en Vmejor.
-

2.9. Generación y visualización del vector Vcualquiera

cpp

CopiarEditar

```

cout << "Vector cualquiera: " << endl;

for(i=1; i<=n/2; i++)
{
    Vcualquiera[i] = x;
}

```

```

for(i=n/2+1; i<=n; i++)
{
    Vcualquiera[i] = x+i;
}

```

```
}
```

```
for(i=1; i<=n; i++)
```

```
    cout << Vcualquiera[i] << ' ';
```

- **Genera el vector Vcualquiera:**
 - Llena la **primera mitad** con x.
 - Llena la **segunda mitad** con valores $x + i$.
 - **Lo muestra en pantalla.**
-

2.10. Conteo de x en Vcualquiera

cpp

CopiarEditar

```
resultado = Contar(Vcualquiera,n,x);
```

```
cout << endl;
```

- Llama a Contar(Vcualquiera, n, x) para contar cuántas veces aparece x.
-

2.11. Muestra el resultado final

cpp

CopiarEditar

```
cout << "\nCASO PEOR" << endl;
```

```
cout << "Numero veces: " << Contar(Vpeor, n, x) << endl;
```

```
cout << "CASO MEJOR" << endl;
```

```
cout << "Numero veces: " << Contar(Vmejor, n, x) << endl;
```

```
cout << "OTRO CASO" << endl;
```

```
cout << "Numero veces: " << Contar(Vcualquiera, n, x) << endl;
```

- Imprime cuántas veces x aparece en cada vector.
-

2.12. Finaliza el programa

cpp

CopiarEditar

```
system("pause");  
  
return 0;  
  
}
```

- `system("pause");` se usa en **Windows** para pausar la ejecución hasta que el usuario presione una tecla.
 - `return 0;` indica que el programa finalizó correctamente.
-

3. Resumen de ejecución

1. Se pide x y n .
2. Se crean los vectores V_{peor} , V_{mejor} , y $V_{cualquiera}$.
3. Se llenan y muestran los vectores.
4. Se cuenta cuántas veces aparece x en cada vector.
5. Se imprimen los resultados.

Este código permite analizar diferentes casos de búsqueda de x y podría aplicarse al análisis de eficiencia de algoritmos.

Si necesitas alguna aclaración o modificación, dime. 😊

Tú dijiste:

segun el codigo cual seria el algoritmo?

ChatGPT Plus

Algoritmo basado en el código practica1.cpp

El código en **C++** implementa un **algoritmo** que genera y analiza tres vectores con distintos patrones de llenado y cuenta la cantidad de veces que un número (x) aparece en ellos. Basándonos en el código, el algoritmo en **pseudocódigo** sería el siguiente:

Algoritmo en pseudocódigo

pseudocode

CopiarEditar

INICIO

```

// 1. Solicitar valores al usuario
Escribir("Introduce valor a buscar (x): ")
Leer(x)

Escribir("Introduce n.elementos del vector (n): ")
Leer(n)

// 2. Reservar memoria para los tres vectores
Crear Vpeor[n+1], Vmejor[n+1], Vcualquiera[n+1]

// 3. Verificar si la memoria fue asignada correctamente
Si (Vpeor == NULL o Vmejor == NULL o Vcualquiera == NULL) Entonces
    Escribir("Error al reservar memoria, reviselo")
    Terminar programa
FinSi

// 4. Generar y mostrar el vector "peor caso"
Escribir("Vector peor: ")
Para i desde 1 hasta n Hacer
     $V_{peor}[i] \leftarrow x$ 
FinPara
Escribir(Vpeor)

// Contar ocurrencias de x en Vpeor
resultado  $\leftarrow$  Contar(Vpeor, n, x)

// 5. Generar y mostrar el vector "mejor caso"
Escribir("Vector mejor: ")

```


Para i desde 1 hasta n Hacer

$V_{\text{mejor}}[i] \leftarrow x + i$

FinPara

Escribir(V_{mejor})

// Contar ocurrencias de x en V_{mejor}

resultado \leftarrow Contar(V_{mejor} , n, x)

// 6. Generar y mostrar el vector "cualquiera"

Escribir("Vector cualquiera: ")

Para i desde 1 hasta $n/2$ Hacer

$V_{\text{cualquiera}}[i] \leftarrow x$

FinPara

Para i desde $(n/2) + 1$ hasta n Hacer

$V_{\text{cualquiera}}[i] \leftarrow x + i$

FinPara

Escribir($V_{\text{cualquiera}}$)

// Contar ocurrencias de x en $V_{\text{cualquiera}}$

resultado \leftarrow Contar($V_{\text{cualquiera}}$, n, x)

// 7. Mostrar resultados finales

Escribir("\nCASO PEOR")

Escribir("Numero veces: ", Contar(V_{peor} , n, x))

Escribir("CASO MEJOR")

Escribir("Numero veces: ", Contar(V_{mejor} , n, x))

Escribir("OTRO CASO")

Escribir("Numero veces: ", Contar(Vcualquiera, n, x))

FIN

Explicación del algoritmo

1. **Solicita los datos al usuario:** Se piden x (el número a buscar) y n (tamaño del vector).
2. **Reserva memoria dinámica** para los tres vectores.
3. **Verifica si la memoria fue asignada correctamente.** Si hay error, finaliza el programa.
4. **Genera y muestra el "peor caso":**
 - Llena el vector Vpeor completamente con x.
 - Muestra el vector y cuenta las ocurrencias de x.
5. **Genera y muestra el "mejor caso":**
 - Llena Vmejor con valores desde x + i.
 - Muestra el vector y cuenta las ocurrencias de x.
6. **Genera y muestra el "caso cualquiera":**
 - Llena la primera mitad con x y la segunda mitad con valores crecientes.
 - Muestra el vector y cuenta las ocurrencias de x.
7. **Muestra los resultados finales,** indicando cuántas veces aparece x en cada caso.

Función auxiliar Contar en pseudocódigo

pseudocode

CopiarEditar

Funcion Contar(V, n, x) → Entero

Definir i, nveces Como Entero

nveces ← 0

Para i desde 1 hasta n Hacer

Si $V[i] == x$ Entonces

nveces \leftarrow nveces + 1

FinSi

FinPara

Retornar nveces

FinFuncion

Descripción:

- La función recibe el vector V, su tamaño n y el valor x a buscar.
- Inicializa un contador nveces = 0.
- Recorre el vector sumando 1 cada vez que encuentra x.
- Retorna la cantidad total de veces que x aparece en V.

Conclusión

Este **algoritmo permite evaluar la cantidad de veces que un número x aparece en distintos tipos de arreglos**, lo cual es útil en análisis de eficiencia de búsqueda y algoritmos de ordenación. Se genera un peor caso (donde x siempre aparece), un mejor caso (donde x no está), y un caso intermedio.