Diseñar y escribir "Mejor Pivote"
Juan Felipe Morales Espitia PUJ

Código:

```
/* ------------------------------------------------------------------------------------

  Diseñar y escribir un algoritmo que informe el mejor pivote de un arreglo de números.

NOTA: El mejor pivote es aquella posición que separa el arreglo en dos, tal que la suma  de   los
promedios de ambos suba-rreglos es máxima.


  Programado por: Felipe Morales PUJ - Estructura de Datos 2310  :)


------------------------------------------------------------------------------------

*/


//Inicio Programa
#include <iostream>
#include <vector>
using namespace std;


// Función para encontrar el mejor pivote en un arreglo de números
int EncontrarMejorPivote(vector<int>& arr) {
    int suma_total = 0; // Variable para almacenar la suma total de todos los elementos del arreglo
    for (int num : arr) {
        suma_total += num; // Aca serìa la suma de todos los elementos del arreglo
```

```
    }


    int suma_izquierda = 0; // Variable para almacenar la suma parcial de elementos en el subarreglo izquierdo

    int mejor_pivote = 0; // Variable para almacenar la posición del mejor pivote (inicialmente es 0)

    double max_promedio = static_cast<double>(suma_total) / arr.size(); // Variable para almacenar el promedio máximo (inicialmente el promedio total)


    // Recorrer el arreglo para encontrar el mejor pivote

    for (int i = 0; i < arr.size() - 1; i++) {

        suma_izquierda += arr[i]; // Actualiza la suma parcial de elementos en el sub-arreglo izquierdo

        int suma_derecha = suma_total - suma_izquierda; // Calcula la suma de elementos en el sub-arreglo derecho

        double promedio_izquierda = static_cast<double>(suma_izquierda) / (i + 1); // Calcula el promedio del sub-arreglo izquierdo

        double promedio_derecha = static_cast<double>(suma_derecha) / (arr.size() - (i + 1)); // Calcula el promedio del sub-arreglo derecho

        double promedio_actual = promedio_izquierda + promedio_derecha; // Calcular el promedio de ambos sub-arreglos


// Mostrar el proceso en cada iteración al usuario para entendimiento del mismo


    //Primero muestro el punto de pivote en el indice y como se divide el arreglo en DOS sub-arreglos

        cout << "Punto de pivote en el índice " << i << ": [" << i + 1 << " elementos] | [" << arr.size() - (i + 1) << " elementos]" << endl;

    //Luego muestra la suma ttal del sub-arreglo izquierdo y posteriormente el derecho

        cout << "Suma izquierda: " << suma_izquierda << " | Suma derecha: " << suma_derecha << endl;

    //Luego muestra el promedio total del sub-arreglo izquierdo y posteriormente el derecho

        cout << "Promedio izquierdo: " << promedio_izquierda << " | Promedio derecho: " << promedio_derecha << endl;

    //Luego muestra el promedio actual de los sub-arreglos y el mejor encontrado en esa     iteración
```

```
        cout << "Promedio actual: " << promedio_actual << " | Mejor promedio encontrado hasta ahora: " << max_promedio << endl;
    //Se para el indice de cada iteracion
        cout << "-------------------------------------------------" << endl;

    /* Si el promedio actual es mayor que el máximo promedio encontrado hasta el momento
    se actualiza el máximo promedio y se guarda la posición del mejor pivote*/
        if (promedio_actual > max_promedio) {
            max_promedio = promedio_actual;
            mejor_pivote = i + 1;
        }
    }

    // Devuelve la posición del mejor pivote
    return mejor_pivote;
}

//Main
int main() {
    int n;
  //Le pedimos al usuario que digite el tamaño del arreglo
    cout << "Ingrese el tamano del arreglo: ";
    cin >> n;

  //Inicializamos el arreglo
    vector<int> arr;
  //Pedimos al usuario  los números que contendrà el arreglo (Nota: Separado por espacios)
    cout << "Ingrese " << n << " numeros separados por espacios: ";
  //Aca pues vamos guardando por medio de un ciclo cada número ingresado
```

```
  for (int i = 0; i < n; i++) {
    int num;
    cin >> num;
    arr.push_back(num);
  }

 //LLamar funcion EncontrarMejorPivote
  int mejor_pivote = EncontrarMejorPivote(arr);
 //Muestra el resultado final del "Mejor Pivote Encontrado" en la posicion correspondiente
  cout << "Mejor pivote encontrado en la posicion: " << mejor_pivote << endl;
  return 0;
 //FIN MAIN
}
//FIN PROGRAMA
```

Para realizar el programa me base en el proceso de “Divide y vencerás“ dividiendo el arreglo en dos sub_arreglos y por medio de “recursividad” ir calculando el promedio de los sub-arreglos (Volviendo el problema mayor en sub-problemas más pequeños), también realizando los cálculos del peor caso posible del algoritmo este sería O(n^2) debido a que n sería el arreglo y en algunos ciclos “for” se utiliza el arreglo para realizar operaciones lo que sería un máximo de n^2 en el peor de los casos.

Link_replit: https://replit.com/@ChKmAn/ED001MejorPivoteFelipeMorales