2º Practica - Divide y Vencerás

Generated by Doxygen 1.9.8

1 Respuestas sobre las preguntas de la práctica 1
1.1 Análisis Teórico . . . 1
1.1.1 Máximo y Mínimo . . . 1
1.1.2 SubVector Suma Máxima . . . 1

2 Class Index 3
2.1 Class List . . . 3

3 File Index 5
3.1 File List . . . 5

4 Class Documentation 7
4.1 ResultadoMaxMin Struct Reference . . . 7
4.1.1 Detailed Description . . . 7
4.1.2 Member Data Documentation . . . 7
4.1.2.1 max_idx . . . 7
4.1.2.2 min_idx . . . 8
4.2 ResultadoSubSumMax Struct Reference . . . 8
4.2.1 Detailed Description . . . 8
4.2.2 Member Data Documentation . . . 8
4.2.2.1 fin . . . 8
4.2.2.2 inicio . . . 9
4.2.2.3 suma . . . 9

5 File Documentation 11
5.1 maximo_minimo_DyV.cpp File Reference . . . 11
5.1.1 Detailed Description . . . 12
5.1.2 Function Documentation . . . 12
5.1.2.1 encontrar_maximo_minimo() . . . 12
5.1.2.2 encontrar_maximo_minimo_segmento() . . . 13
5.1.2.3 main() . . . 13
5.2 maximo_minimo_DyV.cpp . . . 14
5.3 maximo_minimo_Scl.cpp File Reference . . . 15
5.3.1 Detailed Description . . . 15
5.3.2 Function Documentation . . . 16
5.3.2.1 encontrar_maximo_minimo() . . . 16
5.3.2.2 main() . . . 17
5.4 maximo_minimo_Scl.cpp . . . 17
5.5 sub_sumaMaxima_DyV.cpp File Reference . . . 18
5.5.1 Detailed Description . . . 18
5.5.2 Function Documentation . . . 19
5.5.2.1 main() . . . 19
5.5.2.2 max_cruce_sum() . . . 19
5.5.2.3 max_subvector() . . . 19

5.6 sub_sumaMaxima_DyV.cpp 20
5.7 sub_sumaMaxima_Scl.cpp File Reference 21
5.7.1 Detailed Description 22
5.7.2 Function Documentation 22
5.7.2.1 main() 22
5.7.2.2 subSumaMaxima() 22
5.8 sub_sumaMaxima_Scl.cpp 23
5.9 respuestas.dox File Reference 24

Index **25**

Chapter 1

Respuestas sobre las preguntas de la práctica

1.1 Análisis Teórico

1.1.1 Máximo y Mínimo

- Secuencial: O(n) en el mejor y en el peor de los casos.
- Divide y Vencerás: O(n) en el mejor caso y O(n log n) en el peor de los casos.

1.1.2 SubVector Suma Máxima

- Secuencial: O(n^2) en el mejor y en el peor de los casos.
- Divide y Vencerás: O(n) en el mejor caso y O(n log n) en el peor de los casos.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ResultadoMaxMin
: Structura para almacenar los índices del máximo y mínimo encontrados en un segmento del vector 7

ResultadoSubSumMax
: Estructura para almacenar el resultado de la suma máxima de un subvector 8

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

maximo_minimo_DyV.cpp
: Programa para encontrar el máximo y mínimo de un vector utilizando la técnica de Divide y Vencerás 11

maximo_minimo_Scl.cpp
: Programa para encontrar el máximo y mínimo de un vector utilizando un algoritmo secuencial 15

sub_sumaMaxima_DyV.cpp
: Algoritmo para encontrar el subvector con la suma máxima utilizando el método de Divide y Vencerás 18

sub_sumaMaxima_Scl.cpp
: Algoritmo para encontrar el subvector con la suma máxima de forma secuencial básica 21

Chapter 4

Class Documentation

4.1 ResultadoMaxMin Struct Reference

Structura para almacenar los índices del máximo y mínimo encontrados en un segmento del vector.

Public Attributes

- int max_idx
- int min_idx

4.1.1 Detailed Description

Structura para almacenar los índices del máximo y mínimo encontrados en un segmento del vector.

Parameters

max_idx	Índice del máximo encontrado.
min_idx	Índice del mínimo encontrado.

Definition at line 23 of file maximo_minimo_DyV.cpp.

4.1.2 Member Data Documentation

4.1.2.1 max_idx

```
int ResultadoMaxMin::max_idx
```

Definition at line 24 of file maximo_minimo_DyV.cpp.

Referenced by encontrar_maximo_minimo(), and encontrar_maximo_minimo_segmento().

4.1.2.2 min_idx

```
int ResultadoMaxMin::min_idx
```

Definition at line 25 of file maximo_minimo_DyV.cpp.

Referenced by encontrar_maximo_minimo(), and encontrar_maximo_minimo_segmento().

The documentation for this struct was generated from the following file:

- maximo_minimo_DyV.cpp

4.2 ResultadoSubSumMax Struct Reference

Estructura para almacenar el resultado de la suma máxima de un subvector.

Public Attributes

- int suma
- int inicio
- int fin

4.2.1 Detailed Description

Estructura para almacenar el resultado de la suma máxima de un subvector.

Parameters

suma	Suma máxima del subvector
inicio	Índice de inicio del subvector
fin	Índice de fin del subvector

Definition at line 22 of file sub_sumaMaxima_DyV.cpp.

4.2.2 Member Data Documentation

4.2.2.1 fin

```
int ResultadoSubSumMax::fin
```

Definition at line 25 of file sub_sumaMaxima_DyV.cpp.

Referenced by main().

4.2.2.2 inicio

```
int ResultadoSubSumMax::inicio
```

Definition at line 24 of file sub_sumaMaxima_DyV.cpp.

Referenced by main().

4.2.2.3 suma

```
int ResultadoSubSumMax::suma
```

Definition at line 23 of file sub_sumaMaxima_DyV.cpp.

Referenced by main(), and max_subvector().

The documentation for this struct was generated from the following file:

- sub_sumaMaxima_DyV.cpp

Chapter 5

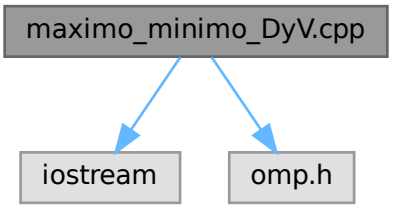
File Documentation

5.1 maximo_minimo_DyV.cpp File Reference

Programa para encontrar el máximo y mínimo de un vector utilizando la técnica de Divide y Vencerás.

```
#include <iostream>
#include <omp.h>
```

Include dependency graph for maximo_minimo_DyV.cpp:



Classes

- struct ResultadoMaxMin

  Structura para almacenar los índices del máximo y mínimo encontrados en un segmento del vector.

Functions

- ResultadoMaxMin encontrar_maximo_minimo_segmento (int ∗v, int inicio, int fin)

  Función para encontrar el máximo y mínimo en un segmento del vector.
- void encontrar_maximo_minimo (int ∗v, int n, int &max_idx, int &min_idx)

  Función recursiva para encontrar el máximo y mínimo en un vector utilizando OpenMP.
- int main (int argc, char const ∗argv[ ])

5.1.1 Detailed Description

Programa para encontrar el máximo y mínimo de un vector utilizando la técnica de Divide y Vencerás.

Este programa utiliza OpenMP para paralelizar la búsqueda del máximo y mínimo en un vector de enteros. La eficiencia teórica en el peor de los casos es O(n log n), y en el mejor de los casos es O(n). El algoritmo divide el vector en dos mitades, encuentra el máximo y mínimo en cada mitad y combina los resultados. Se utiliza la directiva #pragma omp parallel sections para paralelizar la búsqueda en ambas mitades, y se combinan las soluciones para obtener el resultado final. El programa también incluye una estructura ResultadoMaxMin para almacenar los índices del máximo y mínimo encontrados.

Author

Enrique Pinazo Moreno

Date

2025-04-15

Definition in file maximo_minimo_DyV.cpp.

5.1.2 Function Documentation

5.1.2.1 encontrar_maximo_minimo()

```
void encontrar_maximo_minimo (
            int * v,
            int n,
            int & max_idx,
            int & min_idx )
```

Función recursiva para encontrar el máximo y mínimo en un vector utilizando OpenMP.

Parameters

v	Vector de enteros.
n	Tamaño del vector.
max_idx	Índice del máximo encontrado.
min_idx	Índice del mínimo encontrado.

Se plantean los casos base para un elemento y dos elementos, para evitar la recursión innecesaria. Caso base para dos elementos

Dividir el vector en dos mitades y buscar el máximo y mínimo en paralelo

Comparar los resultados de ambas mitades

Definition at line 67 of file maximo_minimo_DyV.cpp.

References encontrar_maximo_minimo_segmento(), ResultadoMaxMin::max_idx, and ResultadoMaxMin::min_idx.

Referenced by main().

5.1.2.2 encontrar_maximo_minimo_segmento()

```
ResultadoMaxMin encontrar_maximo_minimo_segmento (
            int * v,
            int inicio,
            int fin )
```

Función para encontrar el máximo y mínimo en un segmento del vector.

Parameters

v	Vector de enteros.
inicio	Índice de inicio del segmento.
fin	Índice de fin del segmento.

Returns

Los indices del valor maximo y minimo.

Aquí se busca el máximo en la mitad

Aquí se busca el mínimo en la mitad

Almacenar los índices del máximo y mínimo encontrados

Definition at line 35 of file maximo_minimo_DyV.cpp.

References ResultadoMaxMin::max_idx, and ResultadoMaxMin::min_idx.

Referenced by encontrar_maximo_minimo().

5.1.2.3 main()

```
int main (
            int argc,
            char const * argv[] )
```

Tamaño del vector

Declaración de los elementos del vector/

...

Variables para almacenar los índices del máximo y mínimo

Llamada a la función para encontrar el máximo y mínimo

Mostrar el vector

Mostrar los resultados

Liberar la memoria del vector dinámico

Definition at line 126 of file maximo_minimo_DyV.cpp.

References encontrar_maximo_minimo().

5.2 maximo_minimo_DyV.cpp

Go to the documentation of this file.

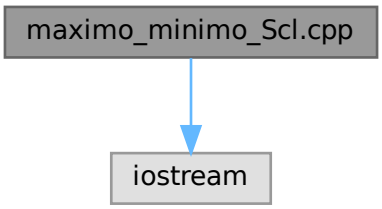
```
00001
00014 #include <iostream>
00015 #include <omp.h>
00016 using namespace std;
00017
00023 struct ResultadoMaxMin {
00024    int max_idx;
00025    int min_idx;
00026 };
00027
00035 ResultadoMaxMin encontrar_maximo_minimo_segmento(int *v, int inicio, int fin) {
00036     ResultadoMaxMin res;
00037     int min = v[0];
00038     int max = v[0];
00039     int idx_max = 0;
00040     int idx_min = 0;
00041     for(int i = inicio; i < fin; i++) {
00043         if(v[i] > max) {
00044             max = v[i];
00045             idx_max = i;
00046         }
00048         if(v[i] < min) {
00049             min = v[i];
00050             idx_min = i;
00051         }
00052     }
00054     res.max_idx = idx_max;
00055     res.min_idx = idx_min;
00056     return res;
00057 }
00058
00067 void encontrar_maximo_minimo(int *v, int n, int &max_idx, int &min_idx) {
00068     if(n <= 1) {
00069         max_idx = 0;
00070         min_idx = 0;
00071         return;
00072     }
00073
00074     if(n == 2) {
00075         if(v[0] > v[1]) {
00076             max_idx = 0;
00077             min_idx = 1;
00078         } else {
00079             max_idx = 1;
00080             min_idx = 0;
00081         }
00082         return;
00083     }
00084
00085     int medio = n/2;
00086     int max_idx_izq, min_idx_izq;
00087     int max_idx_der, min_idx_der;
00088
00090
00091     #pragma omp parallel sections
00092     {
00093         #pragma omp section
00094         {
00095             ResultadoMaxMin res_izq = encontrar_maximo_minimo_segmento(v, 0, medio);
00096             max_idx_izq = res_izq.max_idx;
00097             min_idx_izq = res_izq.min_idx;
00098         }
00099
00100         #pragma omp section
00101         {
00102             ResultadoMaxMin res_der = encontrar_maximo_minimo_segmento(v, medio, n);
00103             max_idx_der = res_der.max_idx;
00104             min_idx_der = res_der.min_idx;
00105         }
00106     }
00107
00108     #pragma omp barrier
00109
00111     #pragma omp single
00112     {
00113         if(v[max_idx_izq] > v[max_idx_der]) {
00114             max_idx = max_idx_izq;
00115         } else {
00116             max_idx = max_idx_der;
00117         }
00118         if(v[min_idx_izq] < v[min_idx_der]) {
00119             min_idx = min_idx_izq;
```

```
00120         } else {
00121             min_idx = min_idx_der;
00122         }
00123     }
00124 }
00125
00126 int main(int argc, char const *argv[]) {
00127     int n = 7;
00128     int *v = new int[n];
00129     v[0] = 5;  v[1] = 904; v[2] = 10;
00130     v[3] = 100; v[4] = 13; v[5] = 54; v[6] = 905;
00131
00132     int max_idx, min_idx;
00133     encontrar_maximo_minimo(v, n, max_idx, min_idx);
00134
00136     cout « "El vector es: [";
00137     for(int i = 0; i < n; i++) {
00138         cout « v[i];
00139         if(i < n-1) cout « ", ";
00140     }
00141     cout «"]"« endl;
00143     cout « "El máximo es: " « v[max_idx] « " en la posición " « max_idx « endl;
00144     cout « "El mínimo es: " « v[min_idx] « " en la posición " « min_idx « endl;
00145
00146     delete[] v;
00147
00148     return 0;
00149 }
```

5.3 maximo_minimo_Scl.cpp File Reference

Programa para encontrar el máximo y mínimo de un vector utilizando un algoritmo secuencial.

`#include <iostream>`
Include dependency graph for maximo_minimo_Scl.cpp:



Functions

- void encontrar_maximo_minimo (int v[], int n, int &maxIdx, int &minIdx, int &maX, int &miN)

  Función para encontrar el máximo y mínimo en un vector.
- int main (int argc, char const ∗argv[])

5.3.1 Detailed Description

Programa para encontrar el máximo y mínimo de un vector utilizando un algoritmo secuencial.

Este programa utiliza un algoritmo secuencial para encontrar el máximo y mínimo de un vector de enteros. La eficiencia teórica en el peor de los casos es O(n), y en el mejor de los casos es O(n). El algoritmo recorre el vector una sola vez, comparando cada elemento con el máximo y mínimo encontrados hasta el momento. Se utiliza un bucle for para recorrer el vector y encontrar el máximo y mínimo. El programa también incluye una función para imprimir el vector y los resultados.

Author

Enrique Pinazo Moreno

Date

2025-04-15

Definition in file maximo_minimo_Scl.cpp.

5.3.2 Function Documentation

5.3.2.1 encontrar_maximo_minimo()

```
void encontrar_maximo_minimo (
            int v[],
            int n,
            int & maxIdx,
            int & minIdx,
            int & maX,
            int & miN )
```

Función para encontrar el máximo y mínimo en un vector.

Parameters

v	Vector de enteros.
n	Tamaño del vector.
maxIdx	Índice del máximo encontrado.
minIdx	Índice del mínimo encontrado.
maX	Valor máximo encontrado.
miN	Valor mínimo encontrado.

Se plantean los casos base para un elemento y dos elementos, para evitar la recursión innecesaria. Se utiliza un bucle para recorrer el vector y encontrar el máximo y mínimo. Caso base para un y dos elementos

Caso genearl n mayor a 2

Se inicializa los maximo y minimo e indices respectivos en la primera posicion

Se recorre el vector empezando desde la segunda posicion

Se encuentra el maximo

Se encuentra el minimo

Definition at line 27 of file maximo_minimo_Scl.cpp.

Referenced by main().

5.3.2.2 main()

```
int main (
            int argc,
            char const * argv[] )
```

Declaración de los elementos del vector

Llamada a la función para encontrar el máximo y mínimo

Definition at line 71 of file maximo_minimo_Scl.cpp.

References encontrar_maximo_minimo().

5.4 maximo_minimo_Scl.cpp

Go to the documentation of this file.

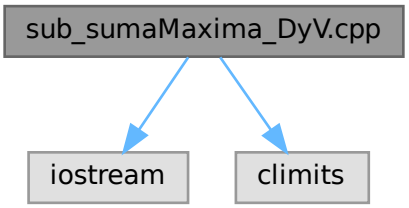
```
00001
00012 #include <iostream>
00013
00014 using namespace std;
00015
00027 void encontrar_maximo_minimo(int v[], int n, int &maxIdx, int &minIdx, int &maX, int &miN){
00029     if(n == 1){
00030         maxIdx = 0;
00031         minIdx = 0;
00032         maX = v[0];
00033         miN = v[0];
00034     }else if (n == 2){
00035         if (v[0] > v[1]){
00036             maxIdx = 0;
00037             minIdx = 1;
00038             maX = v[0];
00039             miN = v[1];
00040         }else{
00041             maxIdx = 1;
00042             minIdx = 0;
00043             maX = v[1];
00044             miN = v[0];
00045         }
00046     }
00048     else{
00050         maX = v[0];
00051         miN = v[0];
00052         maxIdx = 0;
00053         minIdx = 0;
00054
00055         for(int i = 1; i < n; i++){
00057             if(v[i]>maX){
00058                 maX = v[i];
00059                 maxIdx = i;
00060             }
00062             if(v[i]<miN){
00063                 miN = v[i];
00064                 minIdx = i;
00065             }
00066         }
00067     }
00068 };
00069
00070
00071 int main(int argc, char const *argv[])
00072 {
00073     int n = 7;
00074     int v[] = {5,904,10,100,13,54,905};
00075     int idx_max, idx_min, max, min;
00076
00077     encontrar_maximo_minimo(v, n, idx_max, idx_min, max, min);
00078
00079     cout « "Vector: [";
00080     for(int i = 0; i < n; i++){
00081         cout « v[i];
00082         if(i < n-1){cout « ", ";}
00083     }
00084     cout « "]" « endl;
00085     cout « "El maximo es: (" « max « ") en la posicion: [" « idx_max «"]"« endl;
00086     cout « "El minimo es: (" « min « ") en la posicion: [" « idx_min «"]"« endl;
00087     return 0;
00088 }
```

5.5 sub_sumaMaxima_DyV.cpp File Reference

Algoritmo para encontrar el subvector con la suma máxima utilizando el método de Divide y Vencerás.

```
#include <iostream>
#include <climits>
```

Include dependency graph for sub_sumaMaxima_DyV.cpp:



Classes

- struct ResultadoSubSumMax

  Estructura para almacenar el resultado de la suma máxima de un subvector.

Functions

- ResultadoSubSumMax max_cruce_sum (int v[ ], int inicio, int medio, int fin)

  Función para calcular la suma máxima cruzando el medio.
- ResultadoSubSumMax max_subvector (int v[ ], int inicio, int fin)

  Función recursiva para encontrar el subvector con la suma máxima.
- int main (int argc, char const ∗argv[ ])

5.5.1 Detailed Description

Algoritmo para encontrar el subvector con la suma máxima utilizando el método de Divide y Vencerás.

Este algoritmo utiliza la técnica de Divide y Vencerás para encontrar el subvector con la suma máxima en un vector de enteros. Se divide el vector en dos mitades, se calcula la suma máxima en cada mitad y se combina el resultado. La eficiencia teórica en el peor de los casos es O(n log n), y en el mejor de los casos es O(n).

Author

Enrique Pinazo Moreno

Date

2025-04-15

Definition in file sub_sumaMaxima_DyV.cpp.

5.5.2 Function Documentation

5.5.2.1 main()

```
int main (
            int argc,
            char const * argv[] )
```

Vector que cruza el punto medio

Calcular el tamaño del vector

Llamada a la función para encontrar el subvector con la suma máxima

Definition at line 98 of file sub_sumaMaxima_DyV.cpp.

References ResultadoSubSumMax::fin, ResultadoSubSumMax::inicio, max_subvector(), and ResultadoSubSumMax::suma.

5.5.2.2 max_cruce_sum()

```
ResultadoSubSumMax max_cruce_sum (
            int v[],
            int inicio,
            int medio,
            int fin )
```

Función para calcular la suma máxima cruzando el medio.

Parameters

v	Vector de enteros
inicio	Índice de inicio del subvector
medio	Índice medio del subvector
fin	Índice de fin del subvector

Returns

ResultadoSubSumMax Estructura que contiene la suma máxima y los índices de inicio y fin del subvector

Calcular la suma máxima en la parte izquierda

Calcular la suma máxima en la parte derecha

Definition at line 36 of file sub_sumaMaxima_DyV.cpp.

Referenced by max_subvector().

5.5.2.3 max_subvector()

```
ResultadoSubSumMax max_subvector (
            int v[],
            int inicio,
            int fin )
```

Función recursiva para encontrar el subvector con la suma máxima.

Parameters

v	Vector de enteros
inicio	Índice de inicio del subvector
fin	Índice de fin del subvector

Returns

ResultadoSubSumMax Estructura que contiene la suma máxima y los índices de inicio y fin del subvector

Definition at line 73 of file sub_sumaMaxima_DyV.cpp.

References max_cruce_sum(), max_subvector(), and ResultadoSubSumMax::suma.

Referenced by main(), and max_subvector().

5.6 sub_sumaMaxima_DyV.cpp

Go to the documentation of this file.

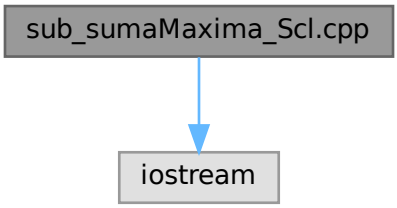
```
00001
00011 #include <iostream>
00012 #include <climits>
00013 using namespace std;
00014
00015
00022 struct ResultadoSubSumMax {
00023     int suma;
00024     int inicio;
00025     int fin;
00026 };
00027
00036 ResultadoSubSumMax max_cruce_sum(int v[], int inicio, int medio, int fin){
00037
00039     int suma_izq = INT_MIN;
00040     int suma_actual = 0;
00041     int inicio_izq = medio;
00042     for(int i = medio; i >= inicio; i--){
00043         suma_actual += v[i];
00044         if(suma_actual > suma_izq){
00045             suma_izq = suma_actual;
00046             inicio_izq = i;
00047         }
00048     }
00049
00051     int suma_der = INT_MIN;
00052     suma_actual = 0;
00053     int fin_der = medio + 1;
00054     for (int i = medio+1; i <= fin; i++)
00055     {
00056         suma_actual += v[i];
00057         if(suma_actual > suma_der){
00058             suma_der = suma_actual;
00059             fin_der = i;
00060         }
00061     }
00062
00063     return {suma_izq + suma_der, inicio_izq, fin_der};
00064 }
00065
00073 ResultadoSubSumMax max_subvector(int v[], int inicio, int fin){
00074     // caso base: vector de un solo elemento
00075     if(inicio == fin){
00076         return {v[inicio], inicio, fin};
00077     }
00078
00079     int medio = (inicio + fin) / 2;
00080
00081     // Resolver recursivamente las dos mitades
00082     ResultadoSubSumMax izq = max_subvector(v, inicio, medio);
00083     ResultadoSubSumMax der = max_subvector(v, medio + 1, fin);
00084
```

```
00085     // Calcular la suma máxima que cruza el punto medio
00086     ResultadoSubSumMax cruzando = max_cruce_sum(v, inicio, medio, fin);
00087
00088     // Determinar qué resultado es mayor
00089     if(izq.suma >= der.suma && izq.suma >= cruzando.suma){
00090         return izq;
00091     } else if(der.suma >= izq.suma && der.suma >= cruzando.suma){
00092         return der;
00093     } else {
00094         return cruzando;
00095     }
00096 }
00097
00098 int main(int argc, char const *argv[])
00099 {
00100     int v[] = {-2, -5, 6, -2, -3, 1, 5, -6};
00101     //int v[] = {8, -5, 6, -2, -3, 1, -6, -6};
00102     //int v[] = {-2, -5, -2, -2, 6, 1, 5, -6};
00103     int n = sizeof(v) / sizeof(v[0]);
00104     ResultadoSubSumMax resultado = max_subvector(v, 0, n - 1);
00105
00106     cout « "Vector original: [";
00107     for(int i = 0; i < n; i++){
00108         cout « v[i];
00109         if(i < n - 1) cout « ", ";
00110     }
00111     cout « "]" « endl;
00112
00113     cout « "Suma máxima: " « resultado.suma « endl;
00114     cout « "Índice inicio: " « resultado.inicio « endl;
00115     cout « "Índice fin: " « resultado.fin « endl;
00116
00117     cout « "Subvector máximo: [";
00118     for(int i = resultado.inicio; i <= resultado.fin; i++){
00119         cout « v[i];
00120         if(i < resultado.fin) cout « ", ";
00121     }
00122     cout « "]" « endl;
00123
00124     return 0;
00125 }
```

5.7 sub_sumaMaxima_Scl.cpp File Reference

Algoritmo para encontrar el subvector con la suma máxima de forma secuencial básica.

`#include <iostream>`
Include dependency graph for sub_sumaMaxima_Scl.cpp:



Functions

- void subSumaMaxima (int *v, int n, int *&vSMax)

  Función para encontrar el subvector con la suma máxima.
- int main (int argc, char const *argv[])

5.7.1 Detailed Description

Algoritmo para encontrar el subvector con la suma máxima de forma secuencial básica.

Este algoritmo utiliza un enfoque secuencial para encontrar el subvector con la suma máxima en un vector de enteros. La eficiencia teórica en el peor de los casos es O(n^2), y en el mejor de los casos es O(n).

Author

Enrique Pinazo Moreno

Date

2025-04-15

Definition in file sub_sumaMaxima_Scl.cpp.

5.7.2 Function Documentation

5.7.2.1 main()

```
int main (
            int argc,
            char const * argv[] )
```

Inicialización del vector

Declaración de un vector donde se guardará la suma máxima

LLamada a la función para encontrar la suma máxima

mostrando el vector

mostrando el vector de suma máxima

Liberar la memoria

Definition at line 65 of file sub_sumaMaxima_Scl.cpp.

References subSumaMaxima().

5.7.2.2 subSumaMaxima()

```
void subSumaMaxima (
            int * v,
            int n,
            int *& vSMax )
```

Función para encontrar el subvector con la suma máxima.

Parameters

v	Vector de enteros
n	Tamaño del vector
vSMax	Vector donde se guardará la suma máxima

Este algoritmo utiliza un enfoque secuencial para encontrar el subvector con la suma máxima. Se inicializa la suma máxima y se recorre el vector para encontrar la suma máxima. Se utiliza un bucle for para recorrer el vector y encontrar la suma máxima. La eficiencia teórica en el peor de los casos es O(n^2), y en el mejor de los casos es O(n). Se utiliza un bucle for para recorrer el vector y encontrar la suma máxima. Si la suma actual más el nuevo elemento es menor que el nuevo elemento, comenzamos un nuevo subvector desde esta posición

Si la suma actual es mayor que la suma global, actualizamos la suma global y los índices de inicio y fin del subvector

Guardamos el subvector de suma máxima

Resto del vector se llena con ceros

Definition at line 25 of file sub_sumaMaxima_Scl.cpp.

Referenced by main().

5.8 sub_sumaMaxima_Scl.cpp

Go to the documentation of this file.

```
00001
00010 #include <iostream>
00011
00012 using namespace std;
00013
00025 void subSumaMaxima(int *v, int n, int *&vSMax){
00026     int maxActual = v[0];
00027     int maxGlobal = v[0];
00028
00029     int inicio = 0;
00030     int fin = 0;
00031     int inicioMax = 0;
00032     int finMax = 0;
00033     for(int i = 1; i < n; i++){
00036         if(maxActual + v[i] <= v[i]){
00037             maxActual = v[i];
00038             inicio = i;
00039         }
00040         else{
00041             maxActual += v[i];
00042             fin = i;
00043         }
00044
00047         if(maxActual > maxGlobal){
00048             maxGlobal = maxActual;
00049             inicioMax = inicio;
00050             finMax = fin;
00051         }
00052         for(int j = 0; j < n; j++){
00053             if(j >= inicioMax && j <= finMax){
00054                 vSMax[j] = v[j];
00055             }
00056             else{
00057                 vSMax[j] = 0;
00058
00059             }
00060         }
00061
00062     }
00063 };
00064
00065 int main(int argc, char const *argv[])
00066 {
00067     int n = 8;
```

```
00068     int vAux[] = {-2,-5,6,-2,-3,1,5,-6};
00069
00070     int *v = new int[n];
00071     for (int i = 0; i < n; i++){
00072         v[i] = vAux[i];
00073     }
00074     int *vSMax = new int[n]{0};
00075
00077     subSumaMaxima(v, n, vSMax);
00079     cout « "Vector: [";
00080     for(int i = 0; i < n; i++){
00081         cout « v[i];
00082         if(i != n-1){cout « ",";}
00083     }
00084     cout « "]" « endl;
00085
00087     cout « "Vector de suma máxima: [";
00088     for(int i = 0; i < n; i++){
00089         cout « vSMax[i];
00090         if(i != n-1){cout « ",";}
00091     }
00092     cout « "]" « endl;
00093
00095     delete[] v;
00096     delete[] vSMax;
00097
00098     return 0;
00099 }
```

5.9 respuestas.dox File Reference

Index

encontrar_maximo_minimo
    maximo_minimo_DyV.cpp, 12
    maximo_minimo_Scl.cpp, 16
encontrar_maximo_minimo_segmento
    maximo_minimo_DyV.cpp, 12

fin
    ResultadoSubSumMax, 8

inicio
    ResultadoSubSumMax, 8

main
    maximo_minimo_DyV.cpp, 13
    maximo_minimo_Scl.cpp, 16
    sub_sumaMaxima_DyV.cpp, 19
    sub_sumaMaxima_Scl.cpp, 22
max_cruce_sum
    sub_sumaMaxima_DyV.cpp, 19
max_idx
    ResultadoMaxMin, 7
max_subvector
    sub_sumaMaxima_DyV.cpp, 19
maximo_minimo_DyV.cpp, 11, 14
    encontrar_maximo_minimo, 12
    encontrar_maximo_minimo_segmento, 12
    main, 13
maximo_minimo_Scl.cpp, 15, 17
    encontrar_maximo_minimo, 16
    main, 16
min_idx
    ResultadoMaxMin, 7

Respuestas sobre las preguntas de la práctica, 1
respuestas.dox, 24
ResultadoMaxMin, 7
    max_idx, 7
    min_idx, 7
ResultadoSubSumMax, 8
    fin, 8
    inicio, 8
    suma, 9

sub_sumaMaxima_DyV.cpp, 18, 20
    main, 19
    max_cruce_sum, 19
    max_subvector, 19
sub_sumaMaxima_Scl.cpp, 21, 23
    main, 22
    subSumaMaxima, 22
subSumaMaxima
    sub_sumaMaxima_Scl.cpp, 22
suma
    ResultadoSubSumMax, 9