

Taller de Programación





Método de ordenación: inserción



Dado un arreglo A y una dimensión lógica (dimL), el algoritmo consiste en ordenar en cada vuelta un elemento a un subconjunto de elementos ya ordenados.

Es decir, en la primera vuelta se toma el subconjunto que contiene el primer elemento del arreglo y obviamente se considera ordenado.

En la segunda vuelta, se toma el segundo elemento del arreglo y se lo inserta de manera que quede ordenado con respecto al subconjunto que ya está ordenado (el primer elemento).

En la tercera vuelta, se toma el tercer elemento del arreglo y se lo inserta de manera que quede ordenado con respecto al subconjunto que ya está ordenado (el primer y segundo elemento).

En la k-ésima vuelta, se toma el k+1 elemento del arreglo y se lo inserta de manera que quede ordenado con respecto al subconjunto de k elementos que ya está ordenados.

En cada vuelta puede ser necesario realizar corrimientos para insertar de manera ordenada cada nuevo elelemnto.

Dado que en la primer vuelta se considera solo el primer elemento y por lo tanto no se debe realizar ningún ordenamiento, en realidad la primera vuelta del algoritmo comienza tomando los dos primeros elementos.

El algoritmo repite (dimension lógica - 1) veces.



5 3 2 1 4 6

VUELTA 1



Tomo ele elemento ubicado en la posición 2 (3) y se compara desde la posición 1 hasta la 1 para ver en qué posición debe insertarse.



5 3 2 1 4 6

Debe insertarse en la posición 1



Se encuentra que el valor debe insertarse en la posición 1, por lo tanto, se realiza un corrimiento desde la posición 1 hasta la 2 para "hacer" lugar en el vector.

5 3 2 1 4 6





dimF = 7dimL= 6

VUELTA 2



Tomo ele elemento ubicado en la posición 3 (2) y se compara desde la posición 1 hasta la 2 para ver en qué posición debe insertarse.



3	5	2	1	4	6	
						_

Debe insertarse en la posición 1



Se encuentra que el valor debe insertarse en la posición 1, por lo tanto, se realiza un corrimiento desde la posición 1 hasta la 3 para "hacer" lugar en el vector.

3 5 2 1 4 6



2 3 5 1 4 6

dimF = 7dimL= 6

VUELTA 3



Tomo ele elemento ubicado en la posición 4 (1) y se compara desde la posición 1 hasta la 3 para ver en qué posición debe insertarse.



2 3 5 1 4 6

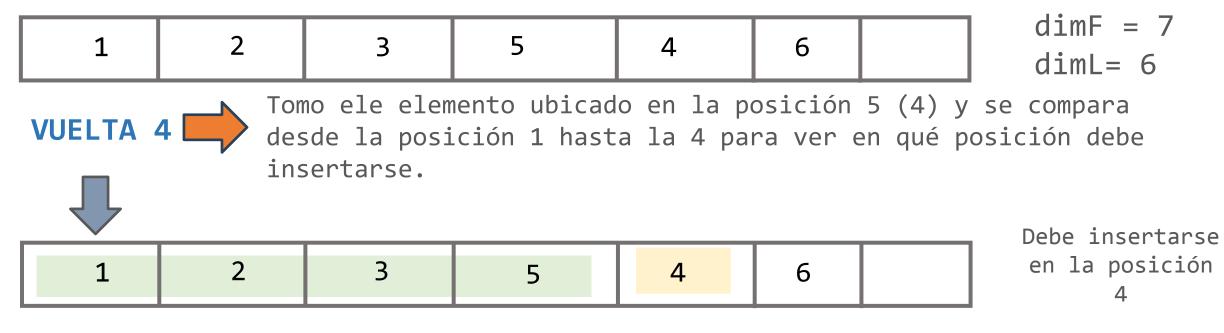
Debe insertarse en la posición 1



Se encuentra que el valor debe insertarse en la posición 1, por lo tanto, se realiza un corrimiento desde la posición 1 hasta la 3 para "hacer" lugar en el vector.

2 3 5 1 4 6







Se encuentra que el valor debe insertarse en la posición 4, por lo tanto, se realiza un corrimiento desde la posición 4 hasta la 5 para "hacer" lugar en el vector.

1 2 3 5 4	6	
-----------	---	--





dimF = 7dimL= 6



Tomo ele elemento ubicado en la posición 6 (6) y se compara desde la posición 1 hasta la 5 para ver en qué posición debe insertarse.



Debe insertarse en la posición 6



Se encuentra que el valor debe insertarse en la posición 6, por lo tanto, se realiza un corrimiento desde la posición 6 hasta la 6 para "hacer" lugar en el vector.

|--|



```
Program ordenar;
Const dimF = ... {máxima longitud del arreglo}
Type
      TipoElem = ... { tipo de datos del vector }
      Indice = 0.. dimF;
      Tvector = array [ 1..dimF] of TipoElem;
Var
  a:Tvector;
  dimL:integer;
Begin
  cargarVector (a, dimL);
  insercion (a, dimL);
End.
```



```
Procedure insercion ( var v: tVector; dimLog: indice );
var i, j: indice; actual: tipoElem;
begin
 for i:= 2 to dimLog do begin
     actual:= v[i];
     j:= i-1;
     while (j > 0) y (v[j] > actual) do
       begin
         v[j+1]:= v[j];
         j:= j - 1;
       end;
     v[j+1]:= actual;
 end;
end;
```



ARREGLOS – Ordenación - insercion

QUE NECESITAMOS CONOCER?

Dimensión lógica del arreglo.

Posición que se debe comparar.

Cuántos elementos ya están ordenados.

CARACTERISTICAS

No tan fácil de implementar.

El tiempo de ejecución es de orden N²

Si los datos están ordenados de menor a mayor el algoritmo solo hace comparaciones, por lo tanto, es de orden (n).

Si los datos están ordenados de mayor a menor el algoritmo hace todas las comparaciones y todos los intercambios, por lo tanto es de orden (N2). comparaciones.

Qué ocurre con las