Facultad de Ingeniería Ingeniería y de Sistemas

Algoritmia y Estructura de Datos: Ordenamiento y búsqueda

Ing. Juan Carlos Sotelo, M.Sc.

Problema base

Problema

Dado un arreglo de n elementos, por ejemplo números enteros, ordenar los elementos de menor a mayor, sin utilizar otro arreglo.

¿Qué se requiere para resolverlo?

La estrategia de solución, requiere que los valores del arreglo cambien de lugar progresivamente hasta que cada uno este en el lugar que le corresponde.

Observaciónes

La estrategia, en cualquier caso, implica que solo dos elementos pueden intercambiar de lugar por vez. Por tanto, se requerirán bucles para realizar todos los intercambios que sean necesarios

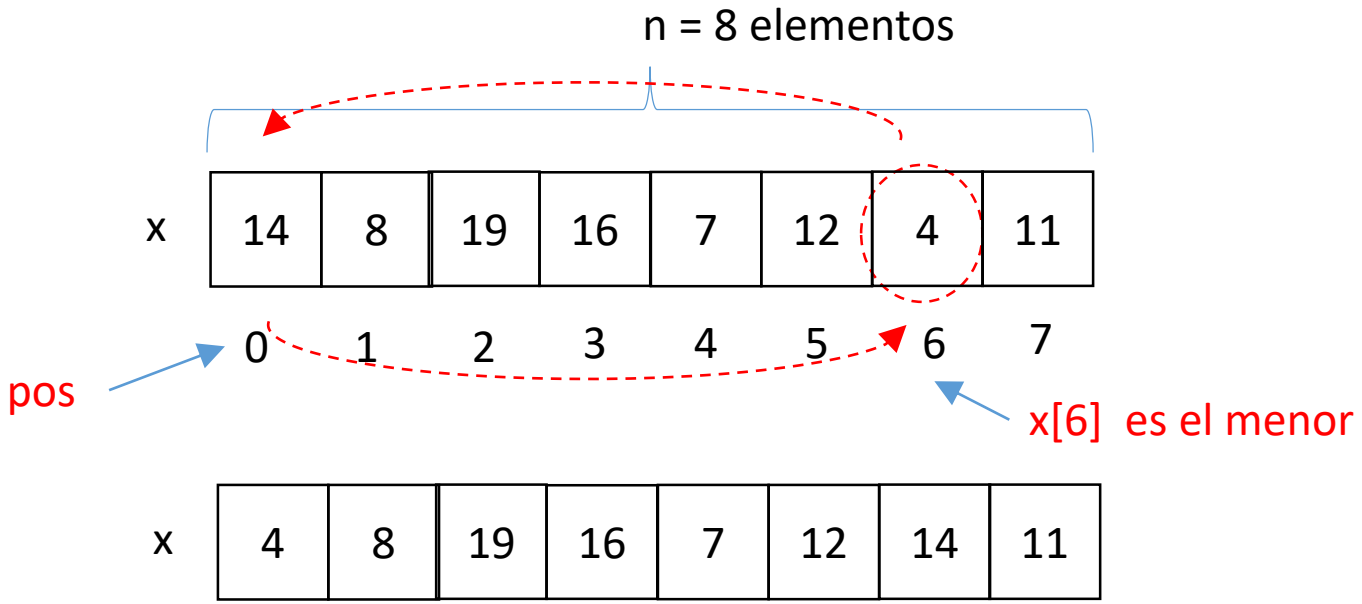
El intercambio de lugar de dos elementos requiere de una variable auxiliar.

Ejemplo:
intercambio de lugar de x[i] con x[j]

```
aux = x[i];
x[i] = x[j];
x[j] = aux;
```

Algoritmo de fuerza bruta para ordenar de menor a mayor

Estamos resolviendo una tarea de clase sobre ordenamiento (ver link en UNIVirtual)




```
pos = 0
Mientras pos < n-1
    ubicar la posición del menor valor desde pos hasta n-1 ( posmenor)
    colocar al menor en la posición pos // intercambiar x[posmenor] con x[pos]
    pos = pos + 1
 finmientras
```

Algoritmo de fuerza bruta para ordenar de menor a mayor

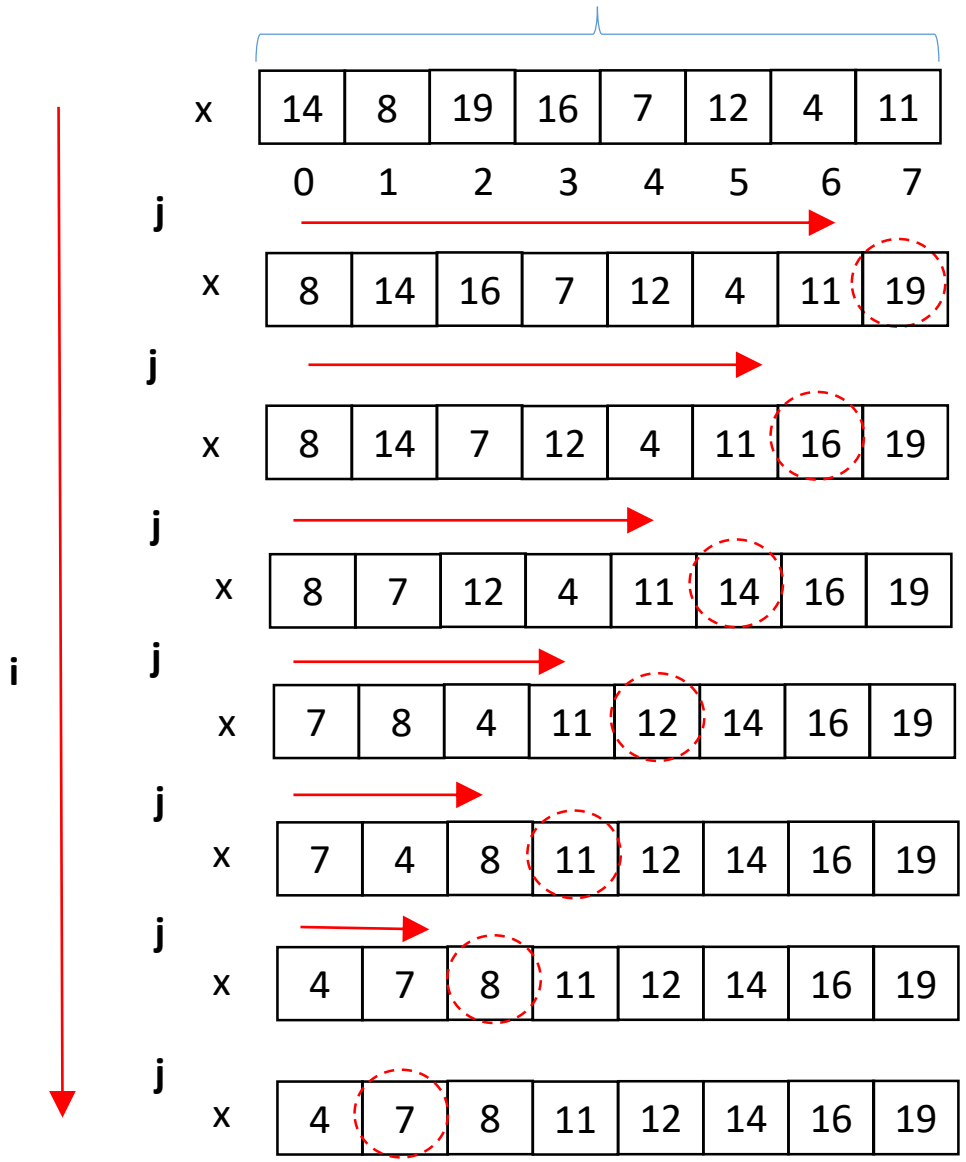
Tarea

Desarrollar el algoritmo de fuerza bruta para ordenar de menor a mayor los elementos de un arreglo de números enteros

Solución

```
#include <iostream>
using namespace std;
main(){
  int n,dato[100],pos=0,posmenor, menor, aux;
  cout<<"Ingrese la cantidad de numeros: "; cin>>n;
  for(int i=0;i<n;i++){
    cout<<"Ingrese numero: "; cin>>dato[i];
  }
  while(pos<n-1){
    for(int i=pos;i<n;i++)
      if(i==pos){
        menor=dato[i]; posmenor=i;
      }
      else
        if(dato[i]<menor){
          menor=dato[i]; posmenor=i;
        }
      aux=dato[pos];      //intercambio de dato[pos] con dato[posmenor]
      dato[pos]=dato[posmenor];
      dato[posmenor]=aux;
      pos++;
    }
    cout<<"Datos ordenados: ";
    for(int i=0;i<n;i++)
      cout<<dato[i]<<" ";
}
```

Algoritmo burbuja para ordenar de menor a mayor




```
for(i=1; i< n;i++)
  for(j=0; j< n-i;j++)
    if (x[j]>x[j+1]){
      aux = x[j];
      x[j] = x[j+1];
      x[j+1] = aux;
    }
```

Problema que requiere ordenamiento de datos

Tarea

Desarrollar un algoritmo que reciba n números enteros no necesariamente ordenados y con algunos valores repetidos 2 o más veces, y determine que valor ocurre más veces y cuantas veces ocurre. Si dos o mas valores empatan en el máximo de ocurrencias, indicar que hay empate y cuantas veces ocurren lo valores empatados.

Sugerencia: Ordenar los valores de menor a mayor aplicando el método de burbuja.

Caso donde 14 es el que mas se repite con 3 ocurrencias

x	14	8	14	16	7	12	14	11
	0	1	2	3	4	5	6	7

Despues de ordenar

x	7	8	11	12	14	14	14	16
	0	1	2	3	4	5	6	7

Caso donde hay empate (7 y 14 son los que más se repiten

x	14	8	14	16	7	12	7	11
	0	1	2	3	4	5	6	7

Despues de ordenar

x	7	7	8	11	12	14	14	16
	0	1	2	3	4	5	6	7

Problema que requiere ordenamiento de datos

Tarea

Desarrollar un algoritmo que reciba n números enteros no necesariamente ordenados y con algunos valores repetidos 2 o más veces, y determine que valor ocurre más veces y cuantas veces ocurre. Si dos o mas valores empatan en el máximo de ocurrencias, indicar que hay empate y cuantas veces ocurren lo valores empatados.

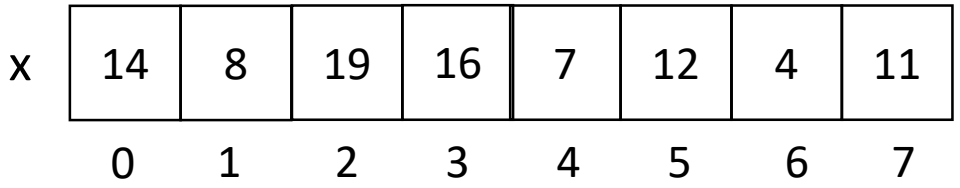
Solución

```
#include<iostream>
using namespace std;
main(){
  int x[100],i,j,n,masrepetido,cont,veces,aux;
  bool empate=false;
  do{
    cout<<"Ingrese la cantidad de numeros: ";cin>>n;
  }while(n<=0);
  for(i=0;i<n;i++) {
    cout<<"numero ";cin>>x[i];
  }
  for(i=1; i< n;i++)
   for(j=0; j< n-i;j++)
     if (x[j]>x[j+1]){
       aux = x[j];
       x[j] = x[j+1];
       x[j+1] = aux;
     }
  masrepetido = x[0];
  veces = 1;
  cont = 1;
  for(i=1; i< n;i++)
    if (x[i] == x[i-1])
      cont++;
    else {
      if (cont>veces){
        empate = false;
        veces = cont:
        masrepetido = x[i-1];
      }
      else
        if (cont==veces)
          empate=true;
      cont = 1;
    }
  if (cont == veces)
    empate = true;
  if (empate)
    cout<<"Hay empate, y los que mas aparecen ocurren "<<veces<<" veces"<<endl;
  else
    cout<<masrepetido<<" es el que mas se repite y aparece " <<veces<<" veces"<<endl;
}
```

Búsqueda secuencial

Problema

Dado un arreglo de n elementos, por ejemplo de números enteros, determinar si cierto dato se encuentra en el arreglo y en que posición aparece por primera vez. Asuma que el arreglo no está ordenado.



dato | 10

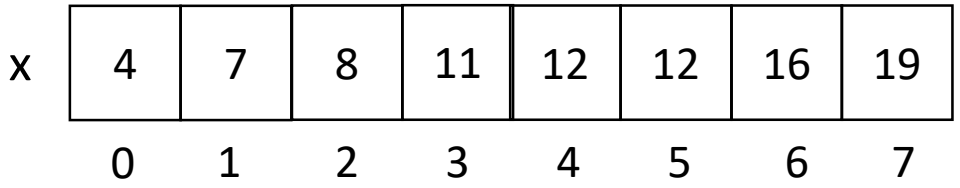
Se requiere determiner si el valor 10 esta en el arreglo y en que posición aparece por primera vez

```
esta = false;
i = 0;
while ((!esta)&&(i<n))
  if (dato == x[i]){
    esta = true;
    pos = i;
  }
  else
    i++;
if (esta)
  cout<<dato<<"esta en la posicion"<<pos;
else
   cout<<dato<<" no esta en el arreglo";
```

Búsqueda secuencial indexada

Problema

Dado un arreglo de n elementos, por ejemplo de números enteros, ordenado de menor a mayor, determinar si cierto dato se encuentra en el arreglo y en que posición aparece por primera vez.



dato | 10

Se requiere determiner si el valor 10 esta en el arreglo y en que posición aparece por primera vez

```
esta = false;
i = 0;
while ((!esta)&&(i<n)&&(dato>=x[i]))
  if (dato == x[i]){
    esta = true;
    pos = i;
  }
  else
   i++;

if (esta)
  cout<<dato<<"esta en la posicion"<<pos;
else
   cout<<dato<<" no esta en el arreglo";
```

Búsqueda binaria

dato 27

x

4	7	8	11	12	12	16	19	23	27	28	31	42	47	57	79
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

inicio

medio= (inicio +fin)/2

fin

Como dato > x[medio]

x

4	7	8	11	12	12	16	19	23	27	28	31	42	47	57	79
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

inicio

Medio

fin

Como dato < x[medio]

x

4	7	8	11	12	12	16	19	23	27	28	31	42	47	57	79
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

inicio

medio

fin

Como dato = x[medio]
termina la búsqueda

Algoritmo de búsqueda binaria

```
inicio = 0;
fin = n-1;
esta = false;
while (!esta && inicio<=fin){
  medio = (inicio+fin)/2;
  if (dato == x[medio]){
    esta = true;
    pos = medio;
  }
  else
    if dato < x[medio]
      fin = medio - 1;
    else
      inicio = medio + 1;
}
if (esta)
  cout<<dato<<" esta en la posicion "<<pos;
else
  cout<<dato<<" no esta en el arreglo";
```