Sortarea unui vector prin interschimbare C++

In continuare, este prezentata sortarea prin interschimbare a unui vector cu N elemente citite de la tastatura. N-ul este, de asemenea, citit de la tastatura.
Sortare crescatoare:

```
#include <iostream>
using namespace std;
int main()
{
int N,i,j, v[100];
cin>>N;
for (i = 1; i <= N; i++)  cin>>v[i];//citire vector
//sortare
for (i = 1; i <= N-1; i++)
       for (j = i+1; j <= N; j++)
             if (v[i] > v[j])
                    {
                       int aux = v[i];
                      v[i] = v[j];
                      v[j] = aux;
                     }
//afisare vector
for (i = 1; i <= N; i++) cout<<v[i]<<" ";
return 0;
}
```

Bubble sort – Vectori C++

In continuare, este prezentata sortarea unui vector de N elemente citite de la tastatura, prin metoda bubble sort. N-ul este, de asemenea, citit de la tastatura.
Principiul dupa care actioneaza bubble sort:
Sortare crescatoare:

```
#include <iostream>
using namespace std;
int main(){
int N, v[100], i,j,sortat;
cin>>N;
for (i = 1; i <= N; i++) cin>>v[i];
do{
     sortat = 1;
     for (i = 1; i <= N-1; i++)
     if (v[i] > v[i+1])
      {
      int aux = v[i];
      v[i] = v[i+1];
       v[i+1] = aux;
        sortat = 0;
       }
}while(sortat == 0);
for (i = 1; i <= N; i++) cout<<v[i]<<" ";
return 0;
}
```

Sortarea prin insertie

Fie un tablou unidimensional care contine n valori intregi.
Realizati un program care ordoneaza crescator elementelor vectorului folosind „algoritmul de insertie”.
Solutia: Elementele vectorului sunt impartite in doua liste: sortata si nesortata.
La fiecare pas primul element al listei nesortate este transferat in lista sortata, exact pe pozitia prin care se respecta ordinea crescatoare a elementelor.
Aceasta operatie se va realize prin deplasarea cu o pozitie spre dreapta a tuturor elementelor mai mari decat el.

Exemplu:
Fie vectorul a=(3,2,1,6,4).
Lista ordonata va fi formata initial din primul element iar lista neordonata de celelalte (n-1) elemente
Pas 1: Se cauta locul lui 2 in lista ordonata (3) si se va deplasa cu o pozitie spre dreapta primul element ==> se obtine vectorul a=(2,3,1,6,4)
Pas 2: Se cauta locul lui 1 in lista ordonata (2,3) si se va deplasa cu o pozitie spre dreapta elementele 2 si 3 ==> se obtine vectorul a=(1,2,3,6,4)
Pas 3: Pentru ca 6>3 nu se vor realiza deplasari, vectorul a ramane acelasi a=(1,2,3,6,4)
Pas 4: Se cauta locul lui 4 in lista ordonata (1,2,3,6) si se va deplasa cu o pozitie spre dreapta elementul 6 ==> se obtine vectorul a=(1,2,3,4,6)

// Sortarea prin insertie – Sortarea unui tablou unidimensional

```
#include<iostream.h>
int v[25],i,j,n,x;
void main()
{
cin>>n;
 for(i=1;i<=n;i++)cin>>v[i];

for(i=1;i<=n;i++) cout<<v[i]<<” ”;

// Sortarea prin INSERTIE
for(i=2;i<=n;i++)    // parcurg vectorul nesortat de la a 2-lea element pana la sfarsit
            if (v[i]<v[i-1])   // primul element din vectorul nesortat se plaseaza pe pozitia corespunzatoare
             { x=v[i];      // valoarea lui v[i] se pierde din vectorul nesortat
              j=i-1;
             while (j>=0 && v[j]>x)
          // mut cu o pozitie spre dreapta toate elementele mai mari decat  x=v[i]
                {
                v[j+1]=v[j];
                j--;
                }
            v[j+1]=x; // insertia primului element pe pozitia corespunzatoare in vectorul sortat
     }
cout<<endl;

for(i=1;i<=n;i++) cout<<v[i]<<” „;}
```

Sortarea prin selectie (selectia minimului)

Fiecare element v[i] se compara cu toate aflate dupa el. Daca se gaseste un element mai mic decat v[i] atunci acestea se vor interschimba. Cand v[i] si-a incheiat rolul de pivot, partea din vector pana la acesta inclusiv, este sortata crescator. */

```
#include<iostream.h>
int v[25],n,i,j,aux,Min, poz;
void main()
{
cin>>n;
for(i=1;i<=n;i++)cin>>v[i];
for(i=1;i<=n;i++) cout<<v[i]<<" ";
for(i=1;i<=n-1;i++)
{
      Min=v[i];poz=i;
         for(j=i+1;j<=n;j++)
         if(v[i]>v[j])
         { Min=v[j];
         poz=j;
         }
      v[poz]=v[i];
      v[i]=Min;
}
cout<<endl;
for(i=1;i<=n;i++)cout<<v[i]<<" ";
}
```

Sortarea prin numarare

Enunt:

Fie un tablou unidimensional care contine n valori intregi.
Realizati un program care ordoneaza crescator elementelor vectorului folosind „algoritmul de numarare”.

Solutia:

Consideram vectorul A. Algorituml de sortare prin numarare consta in gasirea pentru fiecare element A[i] a numarului de elemente din vector mai mici decat el.
Numerele obtinute sunt memorate intr-un alt vector B.
Elementele vectorului A vor fi initial salvate in vectorul auxiliar C.
La finalul algoritmului se vor rescrie in ordine crescatoare elementele vectorului A pe baza valorilor memorate in B si C

Exemplu:

Fie vectorul A=(30,20,1,6,4).
Pas 1: Se realizeaza o copie a vectorului A in C. Se va obtine vectorul C=(30,20,1,6,4).
Pas 2: Se determina elementele vectorului B astfel:
B[i]=cate elemente mai mici decat A[i] sunt in vectorul A Se va obtine vectorul B=(4,3,0,2,1).
Pas 3: Se completeaza elementele vectorului A astfel: A[B[i]]=C[i]. Se va obtine A=(1,4,6,20,30).

```
#include<iostream.h>
int a[25],b[25],c[25],n,i,j;

void main()
{
cin>>n;

for(i=0;i<n;i++) cin>>a[i];
for(i=0;i<n;i++) cout<<a[i]<<” „;
// Sortarea prin numarare
// Pasul 1 fac o copie a vectorului a in vectorul c
for(i=0;i<n;i++)c[i]=a[i];

// Pas 2 crearea vectorului b[i]
// determinam pentru fiecare element cate elemente sunt mai mici

for(i=0;i<n;i++)
        for(j=i+1;j<n;j++)
        if (a[i]<a[j]) b[j]++;
        else b[i]++;

// Pasul 3 Se rescriu elementele lui a in ordine crescatoare
for(i=0;i<n;i++) a[b[i]]=c[i];
cout<<endl;
// Afisarea vectorului sortat
for(i=0;i<n;i++) cout<<a[i]<<” „;
}
```