**Metode de sortare**

1. **Sortare prin interschimbare**

Se ia fiecare element de primul la penultimul si se compara pe rand cu fiecare de dupa el. Daca nu respecta relatia de ordine ceruta se interschimba.

for (i=1; i<n; i++) // a[i] ….a[j]

for (j=**i+1**; j<=n;j++)

if(a[i]>a[j]) // swap(a[i],a[j]);

{aux=a[i];

a[i]=a[j];

a[j]=aux;

}

Exempu: n=4, a=(9, 6, 2, 5)

**i=1** j=2 -> 6 9 2 5

j=3 -> 2 9 6 5

j=4

**i=2** j=3 -> 2 6 9 5

j=4 -> 2 5 9 6

**i=3** j=4 -> 2 5 6 9

**Algoritmul are complexitatea O(n2)**

**Aplicatii**

1. Sortarea el (a[p]…a[q]), cu 1<=p<q<=n

for (i=p; i<q; i++)

for (j=**i+1**; j<=q;j++)

if(a[i]>a[j])

{aux=a[i];

a[i]=a[j];

a[j]=aux;

}

2) Sortarea elementelor pare din vectorul a, cele impare ramanand pe loc.

Ex:

9, 5, **8**, 3, **4, 6 ->** 9, 5, **4**, 3, **6, 8**

for (i=1; i<n; i++)

for (j=**i+1**; j<=n;j++)

if(**a[i]%2==0 && a[j]%2==0** && a[i]>a[j])

{aux=a[i];

a[i]=a[j];

a[j]=aux;

}

**Tema 1**: 510, 618, 511, 512, 514

**Tema 2**: 1608, 180, 183, 185, 186

**1608)**

**Sol 1. //90 p – iese din timp**

int v[1001],a[1001],n,nr,i,j,d,aux;

int main()

{

f>>n;

for(i=1;i<=n;i++)

{ f>>a[i];

**// v[i]= nr. divizori a[i]**

for (d=1; d\*d<a[i]; d++)

if (a[i]%d==0) v[i]+=2;

if (d\*d==a[i]) v[i]++;

}

for (i=1;i<n;i++)

for(j=i+1;j<=n;j++)

if (v[i]<v[j] || v[i]==v[j] && a[i]>a[j])

{

aux=v[i];

v[i]=v[j];

v[j]=aux;

aux=a[i];

a[i]=a[j];

a[j]=aux;

}

for(i=1;i<=n;i++)

g << a[i]<<" ";

return 0;

}

**Sol 2. //100 p - cu formula lui Euler si ciurul lui Eratostene**

int n,i,j,v[1001],c[40001],p[40001],d[1001],k,x,put;

ifstream f("sortare\_divizori.in");

ofstream g("sortare\_divizori.out");

void intsch(int &a, int &b)

{

int aux;

aux=a;

a=b;

b=aux;

}

int main()

{

//Ciurul lui Eratostene pt det nr prime <= 40000, in vectorul p cu k elemente

for(i=2;i\*i<=40000;i++)

if(!c[i])

for(j=i;j\*i<=40000;j++) // marcheaza multiplii lui i

c[j\*i]=1;

for(i=2;i<=40000;i++)

if(!c[i]) p[++k]=i;

//Citire vector

f>>n;

for(i=1;i<=n;i++)

f>>v[i];

/\*

Det in nr numarul de divizoriai lui n prin formul alui Euler

nr=(1+p1)\*(1+p2)\*....\*(1+pk), unde n=f1^p1 \* f2^p2 \*... \*fk^pk

\*/

for(i=1;i<=n;i++)

{

j=1;

x=v[i];

d[i]=1; //d[i]=numarul de divizori ai lui V[i] det. prin Euler

while(p[j]\*p[j]<=v[i] && x>1)

{

put=0;

while(x%p[j]==0)

{

put++;

x/=p[j];

}

d[i]\*=1+put;

j++;

}

if(x>1) d[i]\*=2;

}

//Sortare dupa nr de div

for(i=1;i<n;i++)

for(j=i+1;j<=n;j++)

if(d[i]<d[j] || (d[i]==d[j] && v[i]>v[j]))

{

intsch(v[i],v[j]);

intsch(d[i],d[j]);

}

//Scriere vector

for(i=1;i<=n;i++)

g<<v[i]<<" ";

g.close();

return 0;

}

1. **BubbleSort (Metoda bulelor)**

**Se parcurge vectorul de mai multe ori si se interschimba la fiecare parcurgere doua cate doua elemente alaturate care nu respecta relatia de ordine considerata. Algoritmul se incheie cand intr-o parcurgere nu se mai face nicio interschimbare.**

**Exemplu**

**v=(7, 6, 9, 3, 5)**

* **6, 7, 3, 5, 9**
* **6, 3, 5, 7, 9**
* **3, 5, 6, 7, 9**

do

{ **ok=0;** // pp. ca in parcurgerea curenta nu se mai fac interschimbari

for (i=1; i<n; i++)

if (v[i]>v[i+1])

{aux=v[i]; v[i]=v[i+1]; v[i+1]=aux;

**ok=1;**

}

} while(ok);

1. **Sortarea prin selectie**

**Se identifica indicele/pozitia celui mai mic el. din vector si se interchimba a[1] cu el. de pozitia gasita (cu minimul). Apoi, se det. pozitia celui mai mic el. din vector de la a[2] la a[n] si se interschimba a[2] cu el. de pe pozitia gasita. Alg. Se reia pana la el. de pozitia n-1 inclusiv. Deci, la fiecare pas (i=1,n-1) se interscimba a[i] cu minimul de la el incolo (de la a[i] la a[n]).**

**Exemplu**

**a=(7, 6, 9, 3, 5)**

* i=1 -> imin=4, a[1]⬄a[4] -> **3**, 6, 9, 7, 5
* i=2 -> imin=5, a[2] ⬄ a[5] -> **3, 5**, 9, 7, 6
* i=3 -> imin=5, a[3] ⬄ a[5] -> **3, 5, 6**, 7, 9
* i=4 -> imin=4, -> **3, 5, 6, 7, 9**

for (i=1; i<n; i++)

{ **imin=i**; //imin= indicele minimului de la a[i]….a[n]

for(j=i+1; j<=n; j++)

if (a[j]<a[imin]) imin=j;

if (i!=imin) { aux=a[i]; a[i]=a[imin]; a[imin]=aux]}

}

**Obs.**

1) Alg. are complexiatea O(n2)

1. Pentru sortare descrescatoare se det imax in loc de imin.
2. **Sortarea prin numarare – se aplica doar pe vector cu elemente distincte**

* **metoda are 2 etape**

1. **“etapa de numarare” – in care se obtine vectorul k1, k2,…, kn, unde**

**k[i] = numarul de elemente din vectorul initial < a[i], oricare ar fi i=1,n**

1. **etapa de construire a vectorului sortat, b1, b2,…, bn, in care observam ca a[i] ocupa in vectorul sortat pozitia k[i]+1**

**b[k[i]+1] = a[i], oricare ar fi i=1,n**

**Exemplu**

**a=(7, 6, 9, 3, 5)**

1. **k=(3, 2, 4, 0, 1)**
2. **b=( 3, 5, 6, 7, 9)**

for (i=1; i<n; i++) // I

for (j=i+1; j<=n;j++)

if(a[i]<a[j]) k[j]++;

else k[i]++;

for (i=1; i<=n; i++) // II

b[k[i]+1] = a[i];

// afisarea vectorului sortat

for (i=1; i<=n; i++) cout<<**b[i]**<<” ”;

**Obs.**

1. Alg. are complexiatea O(n2)
2. Are ca ca dezavantaj triplarea zonei de memorie folosita.
3. Pentru sortare descrescatoare, **k[i] = numarul de elemente din vectorul initial > a[i], oricare ar fi i=1,n**
4. **Sortarea prin insertie**

**Pentru fiecare element a[i] (i=2,n) se insereaza in subsirul sortat anterior (a[1]…a[i-1]) inaintea primului element mai mare strict ca el.**

**Exemplu**

a=(7, 6, 9, 3, 5)

i=2 -> a=(**6, 7**, 9, 3, 5)

i=3 -> a=(**6, 7, 9**, 3, 5)

i=4 -> a=(**3, 6, 7, 9**, 5)

i=5 -> a=(**3, 5, 6, 7, 9**)

for (i=2; i<=n; i++)

{

j=1; // **j= pozitia primului el > a[i] ⬄ pozitia pe care-l inseram**

while (a[j]<=a[i] && j<i) j++;

if (j!=i)

// **inseram a[i] pe pozitia j** -> a[1],… a[j],.. a[i-1], a[i],….

aux=a[i];

for(k=i-1; k>=j; k--) a[k+1]=a[k]; // deplasez cu o pozitie la dreapta el. a[j],…a[i-1]

a[j]=aux;

}

**Obs.** Pentru sortare descrescatoare: **while (a[j]>=a[i] && j<i) j++;**

**183) sortare patrate perfecte prin selectie**

#include <iostream>

#include <cmath>

using namespace std;

long n,i,c,j,v[10000],imin,aux;

int main()

{

cin>>n;

for(i=1; i<=n; i++)

{

cin>>v[i];

}

for (i=1; i<n; i++)

if(floor(sqrt(v[i]))==sqrt(v[i]))

{

imin=i;

for(j=i+1; j<=n; j++)

if(floor(sqrt(v[j]))==sqrt(v[j]) && v[j]<v[imin]) imin=j;

if (i!=imin)

{

aux=v[i];

v[i]=v[imin];

v[imin]=aux;

}

}

for(i=1; i<=n; i++)

cout<<v[i]<<" ";

return 0;

}

**511)**

**sortare prin selectie**

#include <iostream>

#include <algorithm>

using namespace std;

int v[100001],imin,imax,j,i,aux;

int main()

{

int n,i,k;

cin>>n>>k;

for(i=1; i<=n; i++)cin>>v[i];

//sortez primele k el. crescator

for (i=1; i<k; i++)

{

imin=i;

for(j=i+1; j<=k; j++)

if (v[j]<v[imin]) imin=j;

if (i!=imin)

{

aux=v[i];

v[i]=v[imin];

v[imin]=aux;

}

}

//sortez ultimele n-k el. descrescator

for (i=k+1; i<n; i++)

{

imax=i;

for(j=i+1; j<=n; j++)

if (v[j]>v[imax]) imax=j;

if (i!=imax)

{

aux=v[i];

v[i]=v[imax];

v[imax]=aux;

}

}

for(i=1;i<=n;i++)cout<<v[i]<<" ";

return 0;

}

**BubbleSort**

#include <iostream>

using namespace std;

int a[1000], n, k, i,ok, aux;

int main()

{

cin >> n >> k;

for (i = 1; i <= n; i++) cin >> a[i];

do

{ok=0;

for (i = 1; i < k; i++)

if (a[i] > a[i+1])

{

aux = a[i];

a[i] = a[i+1];

a[i+1] = aux;

ok=1;

}

} while (ok);

do

{ok=0;

for (i = k+1; i < n; i++)

if (a[i] < a[i+1])

{

aux = a[i];

a[i] = a[i+1];

a[i+1] = aux;

ok=1;

}

} while (ok);

for (i = 1; i <= n; i++)

cout <<a[i] << " ";

return 0;

}

**512) sortare prin numarare**

#include <iostream>

using namespace std;

int n,v[1001],imx,i,j,b[1001],k[1001],mx;

int main()

{

cin>>n;

for(i=1;i<=n;i++)cin>>v[i];

mx=v[1]; imx=1;

for(i=2;i<=n;i++)if(v[i]>mx){mx=v[i];imx=i;}

for(i=1;i<imx-1;i++)

for(j=i+1;j<imx;j++)

if(v[i]<v[j]) k[j]++;

else k[i]++;

for(i=1;i<imx;i++) b[k[i]+1]=v[i];

for(i=imx;i<n;i++)

for(j=i+1;j<=n;j++)

if(v[i]>v[j]) k[j]++;

else k[i]++;

for(i=imx;i<=n;i++) **b[imx+k[i]]=v[i]**;

for(i=1;i<=n;i++)

cout<<b[i]<<" ";

return 0;

}

**Aplicatii - sortare prin interschimbare**

**514)**

int n,a[1001],v[1001],i,j,aux,r,x,y,k;

int main()

{

cin>>n;

for(i=1;i<=n;i++)

cin>>a[i];

for(i=1;i<=n;i++)

{

x=a[i]; y=a[n];

while(y!=0)

{

r=x%y;

x=y;

y=r;

}

if(x==1)

v[++k]=a[i]; // k++; v[k]=a[i];

}

for(i=1;i<k;i++)

for(j=i+1;j<=k;j++)

if(v[i]<v[j])

{

aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for(i=1;i<=k;i++) cout<<v[i]<<" ";

return 0;

}

**180)**

int v[1001],n,i,j,k,imx,imn;

int main()

{

cin>>n;

for(i=1; i<=n; i++) cin >> v[i];

imn=imx=1;

for(i=2; i<=n; i++)

{

if(v[i]>v[imx]) imx=i;

if(v[i]<v[imn]) imn=i;

}

if(imx<imn) swap(imx,imn);

for(i=imn; i<imx; i++)

for(j=i+1; j<=imx; j++)

if(v[i]>v[j]) swap(v[i],v[j]);

for(i=1; i<=n; i++)

cout<<v[i]<<" ";

}

**185)**

**Sol 1 – complexitate O(n)**

int ap[10];

int main()

{

int n,i,j,nr;

cin>>n;

for(i=1; i<=n; i++)

{

cin>>nr;

while(nr>9)

nr=nr/10;

ap[nr]++;

}

for(i=1; i<=9; i++)

for(j=1; j<=ap[i]; j++)

cout<<i;

return 0;

}

**Sol 2 – complexitate O(n2)**

int c[1001],n,i,j,x,aux;

int main()

{

cin>>n;

for(i=1;i<=n;i++)

{

cin>>x;

while(x>9) x/=10;

c[i]=x;

}

for(i=1;i<=n;i++)

for(j=i+1;j<=n;j++)

if(c[i]>c[j])

{

aux=c[i];

c[i]=c[j];

c[j]=aux;

}

for(i=1;i<=n;i++) cout<<c[i];

return 0;

}