Vectori caracteristici și de frecvență

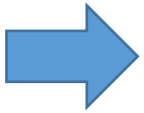
Prof. Mihaela Grindeanu

Colegiul Național “Frații Buzești”, Craiova

Ce vom lucra astăzi?

Probleme de căutare

Probleme de numărare



Vectori de poziție

Vectori de marcare

Vectori de frecvență

Vectori de poziție

Vectori caracteristici

Vectori de frecvență

Ciurul lui Eratostene

Vectori de poziție

Exerciții

Avem un șir de **n numere naturale distincte** (`1<=n<1.000.000`, numerele din șir au maxim `3` cifre).

Simplu:

1. Există `123` în șir? Afișați **poziția lui** `123` în șir, dacă există, și `-1`, în caz contrar.
2. Pe ce poziții se află `15`, `123`, `987` în șir?

Generalizăm:

3. Fiind date m numere naturale de maxim `3` cifre, afișați pozițiile lor în șir sau `-1`, dacă nu există.

1. Există `123` în șir? Afișați poziția lui `123` în șir, dacă există, și `-1`, în caz contrar.

Soluția

Parcurgem șirul de numere și căutăm valoarea `123`, apoi reținem poziția în variabila `poz`.

```
poz = -1; //123 nu se afla in sir
for(i = 1; i <= n; i++)
    if(v[i] == 123)
      poz = i;
fout<<poz;
```

Observație:

Algoritmul parcurge toate cele n elemente ale vectorului, deci avem o complexitate a timpului de execuție **O(n)**, unde `n <= 1.000.000`

2. Pe ce poziții se află 15, 123 și 987 în șir?

Soluția

Avem nevoie de 3 variabile pentru pozițiile numerelor 15, 123, 987.

```
poz15 = poz123 = poz987 = -1;
for(i = 1; i <= n; i++){
    if(v[i] == 15)
        poz15 = i;
    if(v[i] == 123)
        poz123 = i;
    if(v[i] == 987)
        poz987 = i;
}
fout << poz15 <<" " << poz123 << " " << poz987;
```

3. Fiind date m (m < 1.000.000) numere naturale de maxim 3 cifre, afișați pozițiile lor în șir sau –1, dacă nu există

O soluție posibilă

Caută̆m prin vector poziția pentru fiecare din cele m numere.

```
for(j = 1; j <= m; j++){
    cin >> x; //numarul cautat
    poz = -1; //x nu este in vector
    for(i = 1; i <= n; i++)
        if(v[i] == x)
            poz = i;//retin pozitia lui x
    cout << poz << " ";
}
```

Complexitatea algoritmului devine

O(m x n)

Cum m și n sunt mai mici decât 1.000.000, rezultă că numărul de instrucțiuni efectuate pentru găsirea pozițiilor solicitate este de 10^{12}

3. Fiind date m (m < 1.000.000) numere naturale de maxim 3 cifre, afișați pozițiile lor în șir sau −1, dacă nu există

Soluție îmbunătățită

Ne gândim la exercițiul 2 și observăm că am putea păstra complexitatea liniară dacă am avea variabile care să rețină pozițiile celor m valori căutate, lucru imposibil pentru 1.000.000 de valori.

Putem să păstrăm pozițiile cu ajutorul altui vector, **un vector de poziții**:

```
int poz[1000];
```

În care:

poz[0], poz[1], ..., poz[999] vor păstra pozițiile valorilor 0, 1, ..., 999 din vectorul dat. Dacă una din aceste valori nu există în șir, poziția va avea valoarea −1.

3. Fiind date m (m < 1.000.000) numere naturale de maxim 3 cifre, afișați pozițiile lor în șir sau –1, dacă nu există

O soluție imbunatatita

```
cin >> n >> m;
//marchez pozi□iile numerelor de maxim 3 cifre cu -1
for(i = 0;  < 1000 ; i++)
    poz[i] = -1;

for(i = 1 ;i <= n; i++){
    cin >> v[i];
    poz[v[i]] = i;//retin pozitia lui v[i]
}
for(j = 1; j <= m; j++){
    cin >> x; //numarul cautat
    cout << poz[x]<<" "; //afisez pozitia lui x
}
```

Complexitatea algoritmului devine

O(m + n + 1000)

Deci ordinul de complexitate nu depășește

10^6

Vectorul de poziții

- Îl folosim dacă vrem să aflăm rapid poziția unor elemente aflate într-un alt vector.
- Poate fi folosit atunci când caut elemente numere naturale cu o dimensiune rezonabilă pentru a putea construi un vector de poziții

Să aplicăm

pbinfo.ro

- Instabook
- Broscuțe
- Ghirlande

Vectori de poziție

Vectori caracteristici

Vectori de frecvență

Ciurul lui Eratostene

Vectori caracteristici

Exerciții

Avem un număr natural `n` cu maxim `9` cifre.

Simplu:

1. Există cifra `5` în număr? Afișați **DA**, dacă există și **NU**, în caz contrar.
2. Care dintre cifrele `2`, `4`, `6`, sau `8` există în număr?

Generalizăm:

3. Care sunt cifrele care apar în număr? Afișați-le pe ecran separate printr-un spațiu.
4. Care sunt cifrele care nu apar în număr?

Există cifra 5 în număr?
Afișați DA, dacă există și NU, în caz contrar.

```
bool exista5 = false;
while(n != 0){
    if(n%10 == 5)
        exista5 = true;
    n = n/10;
}
if(exista5 == true)
    cout<<"DA";
else
    cout<<"NU";
```

Care dintre cifrele 2, 4, 6, sau 8 există în număr? Afișați-le pe ecran separate printr-un spațiu.

```
bool exista2 = false, exista4 = false,
exista6 = false, exista8 = false;
while(n != 0){
    if(n%10 == 2)
        exista2 = true;
    if(n%10 == 4)
        exista4 = true;
    if(n%10 == 6)
        exista6 = true;
    if(n%10 == 8)
        exista8 = true;
    n = n/10;
}
```

```
if(exista2 == true)
    cout<<2 <<" ";
if(exista4 == true)
    cout<<4 <<" ";
if(exista6 == true)
    cout<<6 <<" ";
if(exista8 == true)
    cout<<8 <<" ";
```

Care sunt cifrele care apar în număr?
Afișați-le pe ecran separate printr-un spațiu.

- Cifrele căutate sunt `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`. Sunt necesare `10` variabile de forma `exista0`, `exista1`, `exista2`, ..., `exista9`.
- Mai ușor de lucrat și de gestionat:

```
bool exista[10];
```

Sunt tot `10` variabile, dar memorate cu ajutorul unui vector:

`exista[0]`, `exista[1]`, `exista[2]`, ..., `exista[9]`

```
cin >> n;
bool exista[10] = {false};
if(n == 0)
    exista[0] = true;
while(n != 0){
    exista[n%10] =  true;
    n = n/10;
}
for(int i = 0; i <= 9; i++)
    if(exista[i] == true)
        cout<< i << " ";
```

Exemplu:

Vectorul `exista` la început:

exista:	false	false	false	false	false	false	false	false	false	alse
indici	0	1	2	3	4	5	6	7	8	9

`n = 348743`

exista:	false	false	false	true	true	false	false	true	true	alse
indici	0	1	2	3	4	5	6	7	8	9

Vectorul `exista` este un vector care marchează prezența unor elemente într-un șir dat

Indicii vectorului sunt termeni ai șirului în care caut

`exista[i] = true`, dacă `i` aparține șirului în care caut

`exista[i] = false`, dacă `i` nu aparține șirului în care caut

Observație importantă!

Termenii șirului trebuie să aibă valori rezonabile astfel încât să poată constitui indicii vectorului de marcare.

Exemplu bun

- Să se afle care sunt numerele cu două cifre care nu se găsesc într-un șir format din n numere naturale (1<=n<=1.000.000, elementele șirului au maxim 7 cifre)

Ce caut? Numere cu două cifre : 10, 11, …, 99 => vector de marcare cu indicii până la 99

=> bool exista[100];

```
for(i=1;i<=n;i++)    //n<=1.000.000 numarul de elem. din sir
{
    cin>>x;           //x termen al sirului=>0<=x<10.000.000
    if(x>=10 && x<=99)
        exista[x]=true;
}
for( i = 10; i<=99; i++ ) //printre numerele cu 2 cifre
    if(exista[i] == false)  //caut valorile nemarcate
        cout << i <<" ";
```

Exemplu în care NU pot aplica vectorul de marcare

- Să se afle care sunt numerele pare care se găsesc într-un șir format din n numere naturale (1<=n<=1.000.000, elementele sirului **maxim 9 cifre**). Afișați numerele în ordine crescătoare fără să se repete.

Ce căutam? Numere pare cu maxim 9 cifre. => un vector de marcare de forma:

```
bool exista[1000000000];
```

Un astfel de vector ocupă aproximativ 1GB de memorie!!!

Nu putem folosi un vector de marcare și căutam altă soluție abordabilă practic.

Util în practică

Afișarea în ordine crescătoare a **numerelor distincte** care apar într-un șir.

Să aplicăm

pbinfo.ro

- Numere8
- Numere1
- NrLipsa
- Nrlipsa1

Vectori de poziție

Vectori caracteristici

Vectori de frecvență

Ciurul lui Eratostene

Vectori de frecvență

Exerciții

Avem un număr natural `n` cu maxim `9` cifre.

Simplu:

1. De câte ori apare cifra `5` în număr?
2. De câte ori apar cifrele `2`, `4`, `6` și `8` în număr?

Generalizam:

3. Care este numărul de apariții al fiecărei cifre din număr?
4. Afișați cifrele care apar o singură dată în număr.

1. De câte ori apare cifra 5 în număr?

```
int ap5 = 0;
while(n != 0){
    if(n%10 == 5)
        ap5++;
    n = n/10;
}

cout <<ap5;
```

2. De câte ori apar cifrele 2, 4, 6 și 8 în număr?

```
int ap2=ap4=ap6=ap8=0;
while(n != 0){
    if(n%10 == 2)
        ap2++;
    if(n%10 == 4)
        ap4++;
    if(n%10 == 6)
        ap6++;
    if(n%10 == 8)
        ap8++;
    n = n/10;
}
```

```
cout << ap2 <<" ";
cout << ap4 <<" ";
cout << ap6 <<" ";
cout << ap8 <<" ";
```

3. Care este numărul de apariții al fiecărei cifre din număr?

- Trebuie să numărăm aparițiile cifrelor `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`. Sunt necesare `10` variabile de forma `ap0`, `ap1`, `ap2`, ..., `ap9`.
- Mai ușor de lucrat și de gestionat:

```
int ap[10];
```

Sunt tot `10` variabile dar memorate cu ajutorul unui vector:

`ap[0], ap[1], ap[2], ..., ap[9]`

```
cin >> n;
int ap[10] ={0};
if(n == 0)
    ap[0] = 1;
while(n != 0){
    ap[n%10]++;
    n = n/10;
}

for(int i = 0; i <= 9; i++)
    cout<< ap[i] << " ";
```

Exemplu:

Vectorul **ap** la început:

ap:	0	0	0	0	0	0	0	0	0	0
indici	0	1	2	3	4	5	6	7	8	9

n = 3487443

ap:	0	0	0	2	3	0	0	1	1	0
indici	0	1	2	3	4	5	6	7	8	9

Vectorul `ap` este un vector care numără elementele dintr-un șir dat

Indicii vectorului sunt termeni ai șirului dat

`ap[i]` conține numărul de apariții al numărului `i` în șir

`ap[i]` se numește frecvența numărului `i`

Observație importantă!

Termenii șirului trebuie să aibă valori rezonabile astfel încât să poată constitui indicii vectorului de frecvență.

Ce inseamnă valori rezonabile?

- Numere naturale;
- Se pot folosi și numere întregi, dar cu translatarea lor către numere naturale;
- Mărimea maximă a valorilor să permită crearea unui vector cu acea dimensiune;

Sortarea numerelor cu ajutorul vectorilor de frecvență.

Exemplu: Să se afișeze în ordine crescătoare cifrele numărului `n`.

Dacă `n = 3487443` și am obținut vectorul de apariții:

ap:	0	0	0	2	3	0	0	1	1	0
indici	0	1	2	3	4	5	6	7	8	9

Parcurgem în ordine crescătoare cifrele de la `0` la `9` și le afișăm de câte ori ne spune numărul de apariții al lor:

```
for(i=0; i<=9;i++){
    nr = ap[i];
    while(nr != 0){            //afisez cifra i de nr ori
        cout << i <<" ";
        nr--;
    }
}
```

Să aplicăm

pbinfo.ro

- Unice
- Subnumar
- sort2
- numere9
- Produs
- 2lap

Vectori de poziție

Vectori caracteristici

Vectori de frecvență

Ciurul lui Eratostene

Ciurul lui Eratostene

Eratostene din Cyrene
(cca 276 - cca 195 î.Hr.)

https://ro.wikipedia.org/wiki/Eratostene

https://ro.wikipedia.org/wiki/Ciurul_lui_Eratostene

Ciurul lui Eratostene

- Vectorul caracteristic: `bool ciur[NMAX];` va avea marcate cu `false` toate numerele prime mai mici decat `NMAX`.
- Multiplii numerelor prime `i` (diferiți de `i`) sunt: `2*i, 3*i ,…, j*i`.
- Acești multiplii nu trebuie să depășească valoarea maximă până la care vrem să găsim numele prime => `j*i < NMAX`.
- Toți multiplii vor fi marcați cu `true`.
- **Numerele 0 și 1 nu sunt numere prime, deci au valoarea `true` .**

Ciurul lui Eratostene
- afișez numerele prime mai mici sau egale cu n -

```
bool ciur[NMAX]={0};
int i,j, n;
cin >> n;
ciur[0] = ciur[1] = true;
for(i = 2; i <= n ; i++)
    if(ciur[i] == 0) //este numar prim
        for(j = 2; j*i <= n; j++) //marchez multiplii lui i cu true
            ciur[j*i] = 1;

for(i = 2; i <= n ; i++)
    if(ciur[i] == false)
        cout<<i<<" ";
```

Ciurul lui Eratostene – îmbunătățire
- afișez numerele prime mai mici sau egale cu n -

```
//observam ca multiplii 2*i,3*i…(i-1)*i au fost marcati anterior
//deci ptentru numarul prim i, vom marca doar multiplii: i*i,(i+1)*i,…j*i <= n
bool ciur[NMAX]={0};
int i,j, n;
cin >> n;
ciur[0] = ciur[1] = true;
for(i = 2; i*i <= n ; i++)
    if(ciur[i] == 0)
        for(j = i; j*i <= n; j++)
            ciur[i*j] = 1;

for(i = 2; i <= n ; i++)
    if(ciur[i] == false)
        cout << i << " ";
```

Să aplicăm

pbinfo.ro

- Eratostene
- Eratostene1
- Divizori4
- Fantastice
- Gcd