Tipuri structurate: tablouri unidimensionale (vectori) – partea a II-a

Verificarea unei proprietăți

Ne întâlnim deseori cu probleme în care trebuie să verificăm dacă toate elementele unui vector au o anumită proprietate, sau dacă există în vector un element care are o anumită proprietate.
Concret, iată cele două situații prezentate separat:

a. Se consideră un vector cu n elemente numere naturale (n≤100). Să se verifice dacă toate elementele vectorului sunt numere pare.

   Vom folosi o variabilă întreagă, numită `ok`, care va avea valoarea 1 dacă toate elementele vectorului sunt numere pare şi 0 în caz contrar. Presupunem inițial că toate elementele vectorului sunt numere pare (`ok=1`); parcurgem vectorul şi dacă găsim un element impar, vom atribui variabilei `ok` valoarea 0.

```
unsigned int a[100];
...
ok=1;
for (i=0;i<n && ok;i++)
     if (a[i]%2!=0) ok=0;
```

b. Se consideră un vector cu n elemente numere întregi (n≤100). Să se verifice dacă există în vector un element negativ.

   Vom folosi o variabilă întreagă numită `gasit`, căreia îi vom atribui valoarea inițială 0 (adică presupunem că toate elementele sunt pozitive). Parcurgem vectorul şi dacă găsim un element negativ, vom atribui variabilei `gasit` valoarea 1.

```
int a[100];
...
gasit=0;
for(i=0;i<n && !gasit; i++)
     if (a[i]<0) gasit=1;
```

Căutarea unui element într-un vector

Se consideră un vector cu n componente întregi (n≤100) şi o valoare întreagă x. Să se verifice dacă x apare sau nu în vector.

Observăm că este o situație similară celei prezentate anterior la punctul b, adică va trebui să verificăm dacă există în vector un element egal cu valoarea x. Această metodă de căutare, în care testăm succesiv elementele vectorului, se numeşte **căutare secvențială**.
Frecvent apar probleme de căutare a unui element într-o mulțime *ordonată*. În această situație, comparăm valoarea x cu elementul din mijlocul vectorului şi avem următoarele posibilități:

- x este egal cu elementul din mijloc, deci am terminat căutarea
- x este mai mic decât elementul din mijloc, deci vom continua căutarea în prima jumătate a vectorului
- x este mai mare decât elementul din mijloc, deci vom continua căutarea în a doua jumătate a vectorului

Această metodă de căutare, în care lucrăm prin înjumătățiri succesive, se numeşte **căutare binară**.

Algoritmul de căutare binară este următorul:

```
st=0;
dr=n-1;
gasit=0;
while (!gasit && st<=dr)
     {mij=(st+dr)/2;
      if (a[mij]==x) gasit=1;
      else if (a[mij]>x) dr=mij-1;
              else st=mij+1;}
if (gasit) cout<<x<<" se gaseste pe pozitia "<<mij;
else cout<<x<<" nu se afla in vector";
```

Interclasare

Fie a un vector cu n elemente şi b un vector cu m elemente, ordonați crescător. Să se construiască un al treilea vector, c, care să conțină atât elementele vectorului a, cât şi elementele vectorului b, în ordine crescătoare.
Exemplu: daca vectorul a are elementele (2,4) iar b are elementele (1,3,5,6,8) se va obține vectorul c cu elementele (1,2,3,4,5,6,8).

O metodă eficientă de rezolvare a acestei probleme este următoarea: parcurgem simultan cei doi vectori, comparând la fiecare pas elementul curent din a cu elementul curent din b. Cel mai mic dintre ele va fi copiat in c şi vom avansa în vectorul din care am copiat şi în c. Când am ajuns la capătul unui vector, copiem în c elementele rămase în celălalt vector.

Algoritmul de interclasare este următorul:

```
i=0; j=0; k=0;
while (i<n && j<m)
        if (a[i]<b[j])
                {c[k]=a[i]; k++; i++;}
        else {c[k]=b[j]; k++; j++;}
while (i<n)
        {c[k]=a[i]; k++; i++;}
while (j<n)
        {c[k]=b[j]; k++; j++;}
```

Bibliografie

- E. Cerchez, M. Serban - Programarea în limbajul C/C++. Volumul I. Editura Polirom
- E. Cerchez –Informatica, Culegere de probleme pentru liceu. Editura Polirom

PROBLEME PROPUSE

1. Fie a un vector cu n(n≤50) componente de tip int.
a. Verificați dacă toate elementele vectorului a sunt numere prime.
b. Verificați dacă există în vectorul a un palindrom.
c. Determinați cea mai lungă secvență formată din elemente egale existentă în vector. Afişați lungimea secvenței, poziția de început şi valoarea care se repetă.

2. Fişierul "numere.in" conține pe prima linie un număr natural n≤255, iar pe a doua linie n numere naturale mai mici sau egale cu 255. Să se construiască fişierul "numere.out" care să conțină:
a. pe primele n linii reprezentările în baza 2 ale numerelor din fişierul "numere.in"
b. pe rândurile următoare, grupele de numere cu proprietatea că orice două numere ale aceleiaşi grupe au în reprezentarea binară un număr egal de cifre de 1. Numerele din aceeaşi grupă se vor afişa pe acelaşi rând cu spațiu între ele.
Exemplu:

numere.in	numere.out
6	1001
9 8 53 5 15 3	1000
	110101
	101
	1111
	11
	53 15
	9 5 3
	8

3. Se consideră un vector a cu n (n≤100) componente numere întregi. Să se şteargă din vectorul a un număr minim de elemente astfel încât la final să se obțină un şir strict crescător de elemente. Primul element din vectorul inițial nu se va şterge.
Exemplu: pentru n=7 şi a=(3,4,8,4,2,1,9) se va afişa (3,4,8,9)

4. Fie un tablou unidimensional cu n elemente valori naturale. Să se determine o submulțime de elemente din tablou, pentru care suma elementelor este divizibilă cu n.
Exemplu: pentru n=7 şi a=(3,6,4,2,11,5,11,6) se va afişa (6,4,2,11,5)

5. petrol
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=141

6. calorii
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=1062

7. livada1
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=1083