Problema 11-E4 – Căutare binară

Accesarea informaţiei digitale pentru volume foarte mari de date necesită algoritmi eficienţi de căutare care să permită localizarea informaţiei utile cât mai rapid şi folosind un minim de resurse de calcul. Un astfel de algoritm este algoritmul de căutare binară.

Cerinţă

Având la dispoziţie un set de valori întregi, distincte, reprezentate sub forma unui vector, **v**[], să se localizeze poziţia unei anumite valori, *x*, specificată de utilizator, folosind un algoritm de căutare binară. Valorile din vector sunt mai întâi ordonate crescător. Dacă valoarea căutată *x* este identică cu valoarea din mijlocul vectorului, atunci algoritmul se încheie şi valoarea a fost găsită. Dacă valoarea căutată este mai mică decât valoarea din mijlocul vectorului, procesul de căutare se repetă pentru sub-vectorul stâng, adică prima jumătate a valorilor din vector, comparând cu noua valoare din mijloc. Dacă valoarea căutată este mai mare decât valoarea din mijlocul vectorului, procesul de căutare se repetă pentru sub-vectorul drept, adică a doua jumătate a valorilor din vector şi aşa mai departe. Vectorul este divizat succesiv în două până când, fie este găsită valoarea căutată, fie sub-vectorul curent are dimensiunea unu. În cazul în care un sub-vector are un număr par de elemente, atunci se consideră valoarea din mijloc a vectorului din care provine, ca fiind implicit parte din acest sub-vector, ceea ce conduce mereu la un sub-vector cu un număr impar de elemente. Programul va afişa pe ecran toate valorile cu care este comparată valoarea *x*, adică valorile din mijloc până la finalizarea procesului de căutare.

Date de intrare

Se vor citi de la tastatură (fluxul *stdin*) următoarele date:

- o valoare întreagă pentru numărul de elemente din vector, *n*, urmată de caracterul *newline* (tasta *Enter*);
- valorile vectorului, **v**[], introduse câte o valoare pe linie urmată de caracterul *newline* (tasta *Enter*);
- o valoare întreagă, *x*, ce reprezintă valoarea care va fi căutată, urmată de caracterul *newline* (tasta *Enter*).

Date de ieşire

Programul va afişa pe ecran la ieşire, toate valorile cu care este comparată valoarea *x* până la finalizarea algoritmului, câte o valoare pe linie, urmată de caracterul *newline* (tasta *Enter*).

ATENŢIE la respectarea cerinţei problemei: afişarea rezultatelor trebuie făcută EXACT în modul în care a fost indicat! Cu alte cuvinte, pe stream-ul standard de ieşire nu se va afişa nimic în plus faţă de cerinţa problemei; ca urmare a evaluării automate, orice caracter suplimentar afişat, sau o afişare diferită de cea indicată, duc la un rezultat eronat şi prin urmare la obţinerea calificativului „Respins”.

Restricţii şi precizări

1. Dimensiunea vectorului **v**[], *n*, este o valoare întreagă, pozitivă şi impară, mai mare sau egală cu 7 şi mai mică sau egală cu 21. Valorile vectorului sunt valori întregi, pozitive, în intervalul [1; 255]. **Se asigură implicit faptul că valorile vectorului sunt diferite între ele**.
2. Valoarea căutată, *x*, este o valoare întreagă, pozitivă, în intervalul [1; 255].
3. **Atenţie**: În funcţie de limbajul de programare ales, fişierul ce conţine codul trebuie să aibă una din extensiile .c, .cpp, .java, sau .m. Editorul web **nu va adăuga automat** aceste extensii şi lipsa lor duce la imposibilitatea de compilare a programului!
4. **Atenţie**: Fişierul sursă trebuie numit de candidat sub forma: <nume>.<ext> unde nume este numele de familie al candidatului şi extensia este cea aleasă conform punctului anterior. Atenţie la restricţiile impuse de limbajul Java legate de numele clasei şi numele fişierului!

Exemple

Intrare	Ieşire
11 20 40 11 54 100 56 67 45 24 21 78 50	45 67 56 54
Explicaţie: **v**=[20 40 11 54 100 56 67 45 24 21 78], x=50, $\mathbf{v}_{sortat}$=[11 20 21 24 40 45 54 56 67 78 100]; Căutare: (pasul 1) $\mathbf{v}_{curent-1}$= $\mathbf{v}_{sortat}$=[11 20 21 24 40 45 54 56 67 78 100], mijloc=45 (index 5), 50>45, căutare în sub-vector dreapta. Valoare comparată=45; (pasul 2) $\mathbf{v}_{curent-2}$=[54 56 67 78 100], mijloc=67 (index 2), 50<67, căutare în sub-vector stânga. Valoare comparată=67; (pasul 3) $\mathbf{v}_{curent-3}$=[54 56 67] (număr par de valori, se adaugă şi centru), mijloc=56 (index 1), 50<56, căutare în sub-vector stânga. Valoare comparată=56; (pasul 4) $\mathbf{v}_{curent-4}$=[54], mijloc=54 (index 0), 50<54, valoarea nu a fost găsită. Valoare comparată=54. Rezultat: 45 67 56 54.	

Intrare	Ieşire
11 20 40 11 54 100 56 67 45 24 21 78 78	45 67 78
Explicaţie: **v**=[20 40 11 54 100 56 67 45 24 21 78], x=78, $\mathbf{v}_{sortat}$=[11 20 21 24 40 45 54 56 67 78 100]. Căutare: (pasul 1) $\mathbf{v}_{curent-1}$= $\mathbf{v}_{sortat}$=[11 20 21 24 40 45 54 56 67 78 100], mijloc=45 (index 5), 78>45, căutare în sub-vector din dreapta. Valoare comparată=45; (pasul 2) $\mathbf{v}_{curent-2}$=[54 56 67 78 100], mijloc=67 (index 2), 78>67, căutare în sub-vector din dreapta. Valoare comparată=67; (pasul 3) $\mathbf{v}_{curent-3}$=[67 78 100] (număr par de valori, se adaugă şi centru), mijloc=78 (index 1), 78==78, valoarea a fost găsită. Valoare comparată=78. Rezultat: 45 67 78.	

Timp de lucru: 120 de minute