Laborator 10 - Mergesort şi Quicksort

10.1. Sortarea prin interclasare (Mergesort)

Aplicând tehnica de programare Divide et Impera, algoritmul de Sortare prin interclasare se realizează astfel:

- dacă şirul curent este netrivial (are cel puţin două elemente), se împarte în două subşiruri care urmează a fi sortate în urma procesului recursiv;
- la pasul de Impera, cele două subşiruri ordonate sunt interclasate.

Interclasarea a două şiruri ordonate este un proces prin care se construieşte un al treilea şir sortat cu elementele celor două şiruri iniţiale.
Interclasarea a două şiruri a şi b ordonate crescător într-un şir c, de asemenea ordonat crescător, presupune:

- cele două şiruri se parcurg simultan de la stânga la dreapta
- cât timp mai sunt elemente de parcurs şi în a şi în b: se compară elementele curente; cel mai mic dintre ele este copiat în c şi în şirul respectiv se trece la elementul următor
- atunci când unul dintre şiruri a fost parcurs complet, elementele rămase în celălalt şir sunt copiate în c.

Interclasarea a 2 vectori, metoda iterativa,a [n], b[m] iar rezultatul in c[n+m]:

1. k=i=j=0;

2. cat timp avem elemente in vectorii a si b

- comparam element cu element a[i] cu b[j];
- plasam elementul mai mic in vectorul c;
- incrementam k, i sau j (dupa caz).

3. Parcurgem vectorii a sau b , daca mai au elemente neprocesate.

Exercitiul 10.1. Astfel, secventa de cod interclasarea doi vectori de intregi este:

$k \leftarrow 0,\ i \leftarrow 0, j \leftarrow 0$

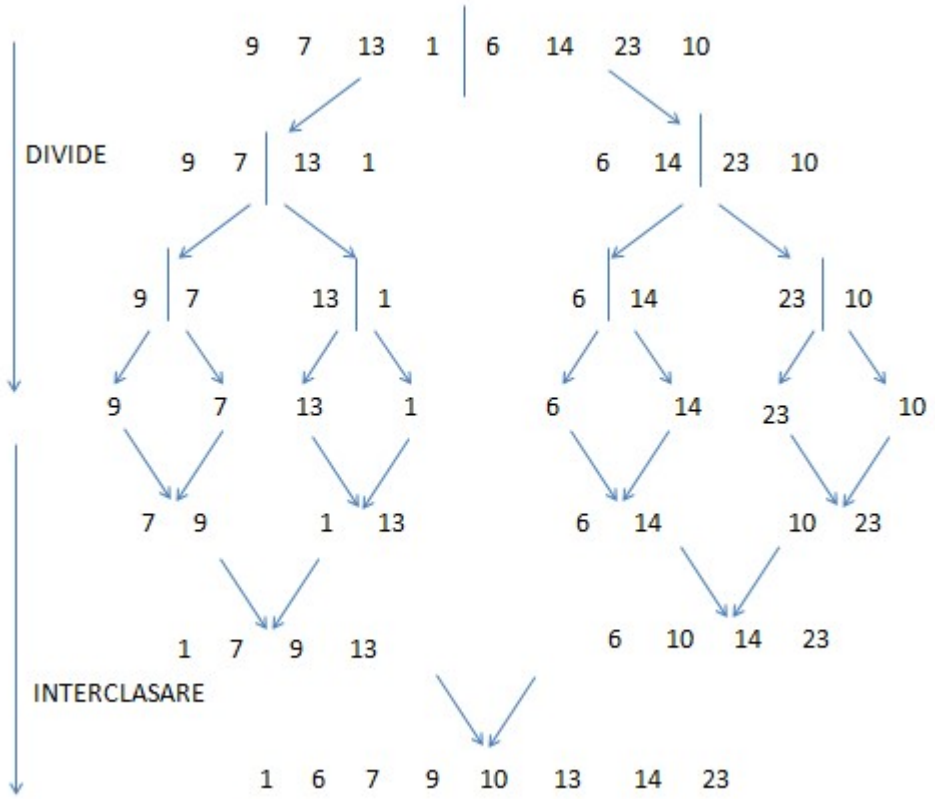
while i< n AND j<m do

```
        if a[i] ≤ b[j] then
            c[k++] ← a[i++]
        else
            c[k++] ← b[j++]
        endif
    endwhile
    if i<n then
        for p ← i,n do
            c[k++] ← a[p]
        endfor
    endif
    if j<m then
        for p ← j,m do
            c[k++] ← b[p]
        endfor
    endif
```

De rezolvat:

Implementati intr-un program C algoritmul de interclasare pentru doua tablouri unidimensionale de n, respectiv m valori intregi.

Exercitiul 10.2. În imaginea de mai jos este exemplificat algoritmul de Sortare prin interclasare (Mergesort) pentru şirul de valori: 9, 7, 13, 1, 6, 14, 23, 10.



Algoritmul presupune execuţia următoarei funcţii recursive:

```
mergesort(v[]; w[]; st; dr)
    if st < dr then
        m ← (st + dr)/2
        mergesort(v; w; st; m)
        mergesort(v; w; m + 1; dr)
        interclasare(v; w; st; m; dr)
    end if
end mergesort
```

Funcţia interclasare() realizează interclasarea şirurilor ordonate v[st]; v[st +1]; ... ; v[m] şi v[m+1];... ; v[dr] în şirul w[0]; w[1]; : : : w[dr-st]. La final, valorile din vectorul w (sortat) se copiază înapoi în v[st]; v[st + 1]; ... v[dr].

```
interclasare(v[]; w[]; st; m; dr)
    i ← st
    j ← m + 1
    k ← 0
    while i <= m AND j <= dr do
        if v[i] <= v[j] then
            w[k] ← v[i]
            i ← i + 1
        else
            w[k] ← v[j]
            j ← j + 1
        end if
        k ← k + 1
    end while
    while i <= m do
        w[k] ← v[i]
        i ← i + 1
        k ← k + 1
    end while
    while j <= dr do
        w[k] ← v[j]
        j ← j + 1
        k ← k + 1
    end while
```

```
    for i ← 0; k - 1 do
        v[i+st] ← w[i]
    end for
end interclasare
```

10.2 Sortarea rapidă (Quicksort)

Ideea acestui algoritm este ca în şirul curent să se aleagă un element pivot (primul sau ultimul element, elementul de la mijloc, un element ales aleator etc.). În urma unei operaţii de pivotare, pivotul va fi mutat pe poziţia pe care ar trebui să se găsească dacă şirul ar fi ordonat. Toate elementele mai mici decât pivotul se vor muta în stânga sa, iar cele mai mari în dreapta. Se obţin astfel două subşiruri, separate de un pivot. Elementele fiecărui subşir se găsesc în "zona" corespunzătoare în ordinea finală, însă nu sunt neapărat sortate. Algoritmul se va aplica în mod recursiv pe fiecare din cele două subşiruri.

Pivotare. Dacă pivotul este ales ca fiind primul element al şirului curent, pentru a îi găsi poziţia în ordinea finală se procedează în felul următor:

- se parcurge simultan şirul de la stânga la dreapta începând cu cea de-a doua poziţie şi de la dreapta la stânga (cu prioritate) până la întâlnire;
- elementele mai mici decât pivotul, aflate la dreapta se vor interschimba cu cele mai mari aflate la stânga. În final se interschimbă valorea pivotului cu valorea pe care s-a oprit parcurgerea.

Exercitiul 10.3. Vom considera şirul 57, 44, 101, 85, 15, 75, 11, 33 şi vom exemplifica operaţia de pivotare.

- se alege 57 ca pivot şi se începe parcurgerea simultană: 57, 44, 101, 85,15, 75, 11,33
- cum 44 < 57, rezultă că 44 trebuie să rămână pe loc, şi se trece la 101: 57, 44, 101, 85, 15, 75, 11, 33
- cum 101 > 57 şi 33 < 57, se interschimbă 101 cu 33 şi se continuă parcurgerea: 57, 44, 33, 85, 15, 75, 11, 101
- are loc o nouă interschimbare, a lui 85 cu 11: 57, 44, 33, 11, 15, 75, 85, 101
- cum 75 > 57, se continuă parcurgerea spre stânga: 57, 44, 33, 11, 15, 75, 85, 101.
- În acest moment cele două parcurgeri s-au întâlnit şi se face interschimbarea lui 57 cu 15, obţinând: 15, 44, 33, 11, 57, 75, 85, 101.

Pivotul este aşezat pe poziţia finală în şirul ordonat şi rămâne să sortăm separat subşirurile 15, 44, 33, 11 şi 75, 85, 101. Se va folosi în mod recursiv aceeaşi tehnică pentru fiecare subşir.

Algorimtul presupune execuţia următoarei funcţii recursive:

```
    quicksort(v[]; st; dr)
        if st < dr then
            poz ← pivotare(v; st; dr)
            quicksort(v; st; poz - 1)
```

```
            quicksort(v; poz + 1; dr)
        end if
    end quicksort
```

Funcţia de pivotare are următorul pseduocod:

```
    pivotare(v[]; st; dr)
        pivot ← v[st]
        s ← st + 1
        d ← dr
        while s <= d do
            while v[d] > pivot do
                d ← d - 1
            end while
            while v[s] < pivot AND s <= d do
                s ← s + 1
            end while
            if s <= d then
                v[s] ↔ v[d]
                d ← d - 1
                s ← s + 1
            end if
        end while
        v[st] ↔ v[d]
        return d
```

end pivotare

De rezolvat:

Implementati intr-un program C secventa de pivotare, prima iteratie, din algoritmul de sortare pentru un tablou unidimensional de n valori intregi: 57, 44, 101, 85, 15, 75, 11, 33. Afisati elementele vectorului dupa executia primei iteratii de pivotare. Observati si explicati efectul acesteia.

Last modified: Wednesday, 18 January 2023, 1:05 PM