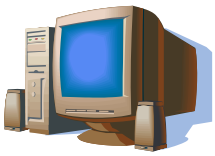
Exerciţii programare

Structuri de control

1. Structuri SIMPLE

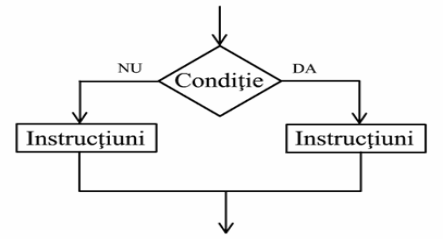


1.1. De atribuire / 1.2. De intrare / 1.3. De ieşire / 1.4. Condiţia (întrebarea, testul)

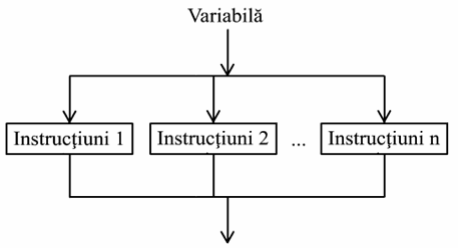
Ia1 - **Metoda paharelor**. Se citesc de la tastatură două numere a şi b, să se interschimbe conţinutul lor.
Ia2 - Se dă temperatura în grade Celsius, să se afişeze în grade Farenheit. F = (C+32) * 1,8.
Ia3 - Se citește un număr întreg n, să se afișeze valoarea expresiei n/3 cu 4 zecimale.
Ia4 - Se dau de la tastatură lungimea şi lăţimea, să se calculeze aria şi perimetrul dreptunghiului.
Ia5 - Se dă x, se înmulțește cu 259 și cu 39. Să se afișeze rezultatul.

2. Structuri DECIZIONALE

2.1. Alternativa



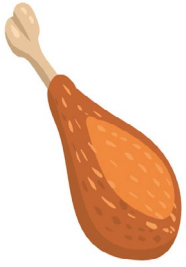
2.2. Selecţia



Id1 - Se citesc de la tastatură două numere, să se afişeze cel mai mare.
Id2 - Se citeşte de la tastatură un număr, să se afişeze dacă este par sau impar.
Id3 - Se citeşte de la tastatură un număr, să se afişeze dacă este pozitiv sau negativ.
Id4 - Se citeşte de la tastatură un număr, să se afişeze dacă este pozitiv, negativ sau nul.
Id5 - Se citesc de la tastatură trei numere, să se afişeze cel mai mare.
Id6 - Se citesc de la tastatură patru numere, să se afişeze cel mai mare.
Id7 - Se citesc de la tastatură trei numere, să se afişeze dacă ele pot fi laturi ale unui triunghi.
şi, dacă da, să se afişeze tipul de triunghi pe care îl pot forma (oarecare, dreptunghic, isoscel, echilateral).
Id8 - Se citesc de la tastatură trei numere, să se ordoneze crescător.
Id9 - Să se scrie un program pentru rezolvarea ecuaţiei de gradul I (ax+b=0).
Id10 - Să se scrie un program pentru rezolvarea ecuaţiei de gradul al II-lea ($ax^2+bx+c=0$).
Id11 - Să se scrie if-ul (condiţia) ca un număr x să aparţină intervalului (4,22].
Id12 - Să se scrie if-ul (condiţia) ca un număr x să NU aparţină intervalului (4,22].
Id13 - Să se scrie if-ul (condiţia) ca un număr x să aparţină intervalului [a,b).
Id14 - Se citeşte un număr, să se afişeze dacă este an bisect sau nu. (Un număr reprezintă un an bisect dacă se împarte la 400 SAU se împarte la 4 şi nu se împarte la 100.)

Is1 - Se citeşte un număr, în funcţie de el să se afişeze ziua corespunzătoare (1=luni, 2=marţi etc, altfel mesaj de eroare)
Is2 - Se citeşte un număr, în funcţie de el să se afişeze luna aferentă (1=ianuarie, 2=februarie etc, altfel mesaj de eroare)
Is3 - Se citeşte un număr care reprezintă o lună din an, să se afişeze ultima zi din luna respectivă.
Is4 - Se citeşte de la tastatură o notă, în funcţie de ea să se afişeze un calificativ (FBine / Bine / Mediu / Slab / FSlab).

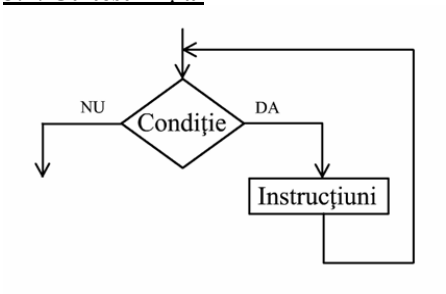
Deliciile C++ :-)



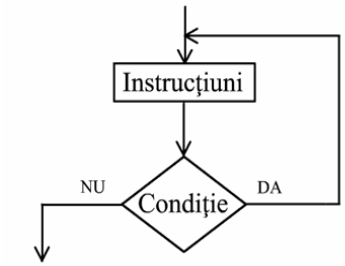
Ix1 - Ce afișează codul: if (-3) cout<<"Da"; else cout <<"Nu";
Ix2 - Ce afișează codul: int x; if (x=3) cout<<"Da"; else cout <<"Nu";
Dar dacă x este declarat astfel: char x[200];
Ix3 - Operatorul condițional: Ce rezultă după instrucțiunea c = (a>b)?a:b;
Ix4 - Ce afișează codul: a=5; cout<<a++; cout<<a;
Dar codul: a=5; cout<<++a; cout<<a;
Ix5 - Ce afișează codul: x=3; y=5; z=--x*y++; cout<<x<<" "<<y<<" "<<z;
Ix6 - Ce afișează codul: i=3; j=5; k=7; u = i < j == j > k; cout<<u;
Ix7 - Ce afișează codul: c=1; u=1; while (u*u<=9) u+=++c; cout<<u;
Ix8 - Se dă x. Dacă x este impar, cu o singură instrucțiune, dar fără a se folosi if, să se scadă 1 din x și să devină par.
Ix9 - Se dă x. Cu o instrucțiune, dar fără a se folosi if, dacă x=1, atunci x devine 0, dacă x=0, atunci x devine 1.
Ix10 - Ce afișează codul: `int a = 3, b; b = a * (a = 2) * a; cout << a << " " << b << endl;`
Ix11 - Ce afișează codul: `a = 0; b = 1; a += b++; cout << a << " " << b << endl;`
Ix12 - Ce afișează codul: `cout << ! (1&&1 || 1&&0);`

3. Structuri REPETITIVE

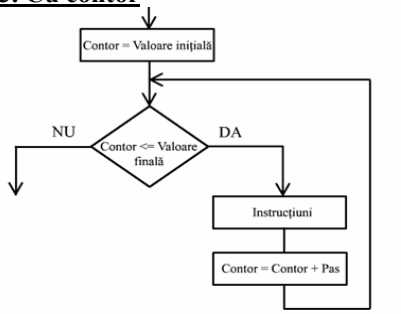
3.1. Cu test inițial



3.2. Cu test final



3.3. Cu contor



Ir1 - Să se afișeze numerele de la 101 la 311, separate cu un spațiu, folosind cele 3 structuri repetitive.
Ir2 - Să se afișeze din 2 în 2 numerele de la 22 la 4, separate cu un spațiu, folosind cele 3 structuri repetitive.
Ir3 - Se citește un număr x, să se afișeze descrescător numerele strict pozitive pare mai mici sau egale cu x.

Ir4 - Se citește x de la tastatură, se calculeze x factorial. x! = 1*2*3*...*x
Ir5 - Se citesc a și n de la tastatură, să se calculeze a la puterea n. a^n
Ir6 - Să se afișeze câte zerouri conține la sfârșit numărul x! (varianta1: x<=12, varianta2: x>10 cifre).

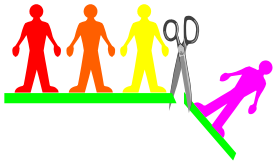
Ir7 - Să se afișeze **divizorii proprii** ai lui x.
Ir8 - Să se afle câți divizori proprii are x.
Ir9 - Să se calculeze suma divizorilor proprii ai lui x.
Ir10 - Să se afișeze dacă x este **prim** sau nu.
Ir11 - Să se afle cel mai mic divizor al lui x (diferit de 1).
Ir12 - Să se afle cel mai mare divizor al lui x (diferit de el însuși).
Ir13 - Să se afișeze toate numerele prime de la 2 până la x. (variantă: numerele neprime) (Ciurul lui Eratostene)
Ir14 - Să se afișeze toate numerele palindrome de la 10 până la x.

Ir15 - Să se afle **cmmdc** a două numere a și b (Algoritmul lui Nicomede. Algoritmul lui Euclid). (exem: 16 și 22)
Ir16 - Să se afișeze dacă a și b sunt prime între ele. (exem: 22 și 25)
Ir17 - Să se afle **cmmmc** a două numere a și b.

Ir18 - Să se afișeze **factorii primi** ai lui x. (exem: 1008, 108, 484, 4004)
Ir19 - Să se afișeze factorii primi ai lui x și puterile lor.
Ir20 - Să se calculeze suma factorilor primi.
Ir21 - Să se calculeze câți factori primi are x.
Ir22 - Să se calculeze suma puterilor factorilor primi.
Ir23 - Să se afișeze factorul prim al lui x cu cea mai mare putere.

Ir24 - Să se verifice dacă un **număr real** are cifre după virgulă și să se afișeze partea zecimală. (exem: 17,2204)
Ir25 - Să se verifice dacă un număr este pătrat perfect.
Ir26 - Să se determine cel mai mare pătrat perfect mai mic decât x.

Exerciții asupra unui număr



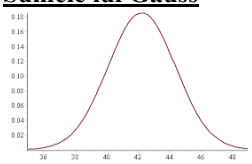
Nf1 - Se citește de la tastatură un număr pozitiv x.
Dacă nu are 4 cifre, să se afișeze un mesaj de eroare. (exem: 1859)
Nf2 - Să se extragă cifrele numărului în 4 variabile și să se afișeze pe ecran una sub alta.
Nf3 - Să se spună dacă numărul este palindrom. (exem: 1991)
Nf4 - Într-o variabilă y să se calculeze inversul numărului.

Nr1 - Se citește de la tastatură un număr întreg x. Să se calculeze și afișeze **suma cifrelor** numărului x.
Nr2 - Să se afișeze **câte** cifre are numărul x.
Nr3 - Să se afișeze cea mai mare cifră din numărul x.
Nr4 - Să se afle **inversul** lui x. (exem: x=1974, inversul=4791)
Nr5 - Să se afișeze dacă numărul este palindrom. (exem: 1991, 12721)
Nr6 - Să se afișeze prima cifră a lui x.
Nr7 - Să se afișeze primele două cifre ale lui x.

Nr8 - Să se elimine prima şi ultima cifră din x.
Nr9 - Să se elimine cifrele pare din x. (exem: x=41859, x devine 159)
Nr10 - Să se înlocuiască cifrele pare cu zerouri. (exem: x=541859, x devine 501059)

Nb1 - Să se afişeze câte cifre are **x în binar**. (exem: 61, 62, 137)
Nb2 - Să se verifice dacă numărul x este rotund (în binar are același număr de 1 şi de 0).
Nb3 - Să se transforme **x în binar**.
Nb4 - Să se transforme x într-un sistem de numerotaţie citit de la tastatură (k). (exem: 147 pt. baza 5, exem: 125 pt. hexa)
Nb5 - Să se verifice dacă x este în binar, să se citească până când este în binar, apoi să se transforme numărul în zecimal.
Nb6 - Să se transforme un număr real x în binar; de la tastatură se introduce şi numărul de zecimale dorite.

Sumele lui Gauss



Ng1 - Să se calculeze 1+2+3+...+n. Formula: n*(n+1)/2.
Ng2 - Suma numerelor impare: 1+3+5+7+ … + (2n -1). Formula: n*n.
Ng3 - Să se calculeze 2+4+6+...+2*n. Formula: n*(primul termen + ultimul termen)/2.
Ng4 - $1^2+2^2+3^2+...+n^2= n(n+1)(2n+1)/6$
Ng5 - $1^3+2^3+3^3+...+n^3= [n(n+1)/2]^2$

Sume diverse

$\sum_{x=1}^{n} \square$

Ns1 - Să se calculeze ∑ x*(x+1), unde x=1,n
Ns2 - Să se calculeze ∑ x*(x+2)/3, unde x=1,n
Ns3 - Să se calculeze $x+x^2+x^3+...+x^n$.
Ns4 - Să se calculeze 1! + 2! + 3! + ... x!

Ns5 - Să se calculeze suma armonică. (exem: pentru **1859**, suma este 1***9** + 2***5** + 3***8** + 4***1**)
Ns6 - Să se calculeze cifra de control a lui x (suma sumei cifrelor până când rămâne o singură cifră).
(Numărul destinului: suma cifrelor care formează data de naştere, până când rămâne o singură cifră).
(Cifra de control din CNP: Ultima cifră este dată în urma unui calcul matematic folosind celelalte 12 cifre ale CNP-ului. Pentru a o afla, se înmulţeşte fiecare cifră din CNP cu cea de pe aceeaşi poziţie din numărul 279.146.358.279. Rezultatele se adună, iar suma obţinută se împarte la 11. Restul din împărţire este cifra de control. Dacă restul este 10, cifra de control este 1. Acest lucru înseamnă că cifra de control este egală cu restul. Cifra de control este cea care validează CNP-ul.)

N numere

Se citeşte n, apoi n numere pozitive.
Ia1 - Să se calculeze suma numerelor. (exem: 8, apoi 22,4,17,5,1,20,25,6)
Ia2 - Să se calculeze produsul numerelor.
Ia3 - Să se afişeze cel mai mare număr citit.
Ia4 - Să se afişeze cel mai mare număr impar citit.
Ia5 - Să se afişeze suma cifrelor pentru fiecare număr.
Ia6 - Să se afişeze suma cifrelor tuturor numerelor.
Ia7 - Să se extragă prima cifră din fiecare număr şi să se calculeze suma lor.

Ib1 - Să se afişeze cea mai mare sumă dintre două numere pozitive citite succesiv.
Ib2 - Să se afişeze perechile de numere impare citite.
Ib3 - Să se afişeze perechile de numere prime citite.
Ib4 - Să se afişeze perechile de numere în care primul citit este mai mare decât al doilea. (Se afişează 14,8 - 10,5 - 12,9)
Ib5 - Să se afişeze perechile de numere prime gemene (diferența dintre ele este 2) până la n. (Exem: 3,5 - 5,7 - 11,13)

Ic1 - Să se afişeze cea mai mare sumă dintre trei numere pozitive citite succesiv.
Ic2 - Să se afişeze perechile de trei numere pare citite.
Ic3 - Să se calculeze termenul n din şirul Fibonacci. f(1)=1, f(2)=1, f(n)=f(n-1)+f(n-2) pentru n>=3.
Ic4 - Se dă s, să se afişeze s ca sumă de termeni ai şirului Fibonacci.

Exerciţii diverse

Ne1 - Să se verifice dacă un număr citit de la tastatură are cifrele ordonate crescător (exem: 1459; verificare pt 1859).
Ne2 - Să se afişeze de câte ori apare cifra x în număr.
Ne3 - Să se afişeze de câte ori apare fiecare cifră în număr.
Ne4 - Să se afişeze o singură dată cifrele care apar în număr. [variantă: care nu apar]
Ne5 - Să se elimine cifrele impare din număr.
Ne6 - Să se afişeze dacă numărul conţine numai cifre pare [variante: impare / identice / prime].
Ne7 - Să se verifice dacă două numere sunt în echer (au același număr de cifre + suma cifrelor de pe aceeaşi poziţie este 9).

Tipuri de număr

Se citeşte de la tastatură un număr întreg x. Să se verifice dacă x este:
superprim - dacă el este prim şi toate prefixele sale sunt toate numere prime (exemplu: 313, 2339).
aproape prim sau semiprim - este produs de două numere prime.
superpalindrom - are pătratul tot palindrom (exemplu: 11).
semipalindrom - cifrele din prima jumătate sunt aceleaşi din a doua jumătate (exemplu: 17817).

perfect - suma divizorilor săi (exceptând numărul însuşi) este egală cu numărul dat.
supraperfect sau abundent - suma divizorilor săi (exceptând numărul însuşi) este mai mare decât numărul dat.
imperfect sau deficient - suma divizorilor săi (exceptând numărul însuşi) este mai mică decât numărul dat.

dublic - are două cifre alăturate egale.
echilibrat - numărul de cifre pare este egal cu numărul de cifre impare.
ciuruit - are minim două zerouri.
corect - suma şi produsul cifrelor sunt pătrate perfecte.
optim - conţine 8 sau suma cifrelor lui este 8.
greu încercat - are două cifre între care diferenţa este 9.
subţire - este mai mic decât suma divizorilor săi fără 1 şi el însuşi.
legat - diferenţa dintre oricare două cifre este 1.
pitic - toate cifrele sunt mai mici decât 4.
generos - suma cifrelor lui e mai mare decat x+2. Să se afle suma numerelor generoase de două cifre.

Vectori

Va1 - Să se citească de la tastatură un vector v cu n elemente, n citit tot de la tastatură.
Va2 - Să se afişeze elementele vectorului.
Va3 - Să se afişeze elementele vectorului, pornind de la ultimul element către primul.
Va4 - Să se afişeze elementele din prima jumătate a vectorului.
Va5 - Să se afişeze elementele de pe poziţii pare.
Va6 - Să se afişeze elementele pare.
Va7 - Să se afişeze elementele pare de pe poziţii impare.
Va8 - Să se afişeze elementele care au cifra zecimalelor egală cu 1.
Va9 - Să se afişeze elementele care au o singură cifră.

Exemplu: n=8, v=(7,4,9,2,6)

Vs1 - Să se calculeze suma elementelor.
Vs2 - Să se calculeze produsul elementelor.
Vs3 - Să se calculeze media aritmetică a elementelor pare.
Vs4 - Să se calculeze norma (radical din suma pătratelor elementelor).
Vs5 - Să se calculeze suma elementelor pare.
Vs6 - Să se calculeze suma elementelor prime.

Vm1 - Să se afle maximul elementelor şi poziţia unde se găseşte prima oară.
Vm2 - Să se afişeze de câte ori există maximul în vector.
Vm3 - Să se afişeze toate poziţiile unde există maximul.
Vm4 - Să se afişeze ultima poziţie unde se găseşte maximul.
Vm5 - Să se afişeze a doua poziţie unde există maximul.
Vm6 - Să se afişeze primele 3 poziţii unde există maximul.
Vm7 - Să se afle minimul elementelor din a doua jumătate a vectorului şi poziţia unde se găseşte prima oară.
Vm8 - Să se afle maximul elementelor pare.

Exerciţii cu folosire contor

Vf1 - Să se afişeze câte elemente din vector sunt pare şi câte sunt impare.
Vf2 - Să se afişeze câte elemente din vector sunt pozitive şi câte sunt negative.
Vf3 - Să se calculeze câte elemente între 2 şi 7 sunt în vector.
Vf4 - Să se afişeze procentual câte elemente sunt peste medie şi câte sunt sub medie.
Vf5 - Să se calculeze câte elemente prime sunt în vector.
Vf6 - Să se calculeze câte elemente din vector au numărul de cifre pare egal cu numărul de cifre impare.
Vf7 - Să se calculeze câte elemente palindrom sunt în vector.

Exerciţii cu folosirea unui singur indice pe vector

Vi1 - Să se şteargă elementul de pe poziţia p din vector. Valoarea lui p se citeşte de la tastatură.
Vi2 - Să se insereze pe poziţia p din vector o valoare x. Valorile lui p şi x se citesc de la tastatură.
Vi3 - Să se mute ultimul element pe prima poziţie în faţa celorlalte.
Vi4 - Elementul de pe poziţia p să se mute la sfârşitul / începutul vectorului.
Vi5 - Să se introducă un număr x citit de la tastatură în poziţia corectă într-un vector ordonat crescător.
Vi6 - Să se şteargă 3 elemente începând cu poziţia p din vector. Valoarea lui p se citeşte de la tastatură.
Vi7 - Să se afişeze diferenţele dintre elementele alăturate.
Vi8 - Să se interschimbe elementele alăturate de pe poziţiile pare cu cele de pe poziţiile impare.

Vectorul de frecvenţe

Vfr1 - Se dă un număr x, să se păstreze într-un vector de câte ori apare fiecare cifră în x (exem: 722772855).
Vfr2 - Să se afişeze cel mai mare număr format din cifrele numărului x luate o singură dată (8752).
Vfr3 - Să se afişeze cel mai mare număr format din cifrele numărului x luate de câte ori apar (877755222).
Vfr4 - Să se afişeze cel mai mic număr format din cifrele numărului x luate o singură dată (2578).
Vfr5 - Să se afişeze cel mai mic număr format din cifrele numărului x luate de câte ori apar (222557778).
Vfr6 - Să se spună dacă se poate forma un palindrom din cifrele numărului x.
Dacă da, să se afişeze cel mai mare număr posibil / cel mai mic număr posibil.
Să se verifice programele pentru valorile 22042017, 404788004.
Vfr7 - Se dă un vector, să se păstreze într-un alt vector de câte ori apare fiecare cifră în elementele primului vector.
Vfr8 - Se dă un vector, să se păstreze într-un alt vector de câte ori apar elementele de două cifre din primul vector.

Exerciţii cu mai mulţi indici pe vector

Vnd1 - Să se inverseze primul element cu ultimul, al doilea cu penultimul şamd (exem: 7,4,9,2,0,6 => 6,0,2,9,4,7).
Vnd2 - Să se elimine duplicatele vecine (elementele egale alăturate) (exem: 7,4,4,4,5,7,4,3,3,7,9 => 7,4,5,7,4,3,7,9).

Folosirea variabilei de tip steag / semafor

Vst1 - Se citeşte de la tastatură un număr (x). Să se afişeze un mesaj dacă x există sau nu în vector.
Vst2 - Se citeşte un număr x, să se verifice dacă numărul divide vreun element al vectorului.
Vst3 - Să se determine dacă vectorul conţine numere negative.
Vst4 - Să se determine dacă vectorul conţine numere pare.
Vst5 - Să se determine dacă vreun element al vectorului are o cifră egală cu trei.
Vst6 - Să se determine dacă vectorul conţine vreun număr prim.

Vst7 - Să se determine dacă toate elementele vectorului sunt strict pozitive.
Vst8 - Să se determine dacă toate elementele vectorului sunt pare.
Vst9 - Să se determine dacă toate elementele vectorului sunt prime.
Vst10 - Să se determine dacă elementele vectorului formează sau nu o progresie aritmetică.
Vst11 - Să se determine dacă elementele vecine dintr-un vector sunt diferite între ele.
Vst12 - Să se determine dacă toate elementele unui vector sunt diferite.
Vst13 - Să se determine dacă elementele unui vector sunt ordonate strict crescător.

Metode de ordonare

Vrd-1 - Metoda simplă (SimpleSort)
Vrd-2 - Metoda bulelor (BubbleSort)
Vrd-3 - Metoda prin aflare minim (SelectionSort)

Vrd-4 - Metoda prin indexare (CountSort) (Într-un alt vector pe poziţie corespunzatoare se stochează numărul de elemente mai mici decât cel curent. Exemplu: Dacă V=(7,4,9,2,6), atunci vectorul Z=(3,1,4,0,2). Explicaţie: z[0]=3 deoarece 7, adică v[0], este mai mare decât trei elemente din V.)
Vrd-5 - Metoda Insert (Să mută elementul spre stânga până găseşte un element mai mic sau egal cu el.)
Vrd-6 - Metoda prin inserţie (InsertionSort) (Să se citească de la tastatură elementele vectorului şi să se introducă în vector pe poziţia necesară astfel încât vectorul să fie ordonat după citirea fiecărui nou element.)
Vrd-7 - HeapSort (Elementele sunt "puse" într-un vector şi gestionate ca într-un arbore binar. Nodul de pe poziţia x are fii pe poziţiile 2x+1 şi 2x+2. Se porneşte algoritmul de la poziţia (n-1)/2 spre 0 şi se verifică "grupuri" de 3 elemente, un tată şi 2 fii. Se află poziţia celui mai mare element dintre cele trei şi, dacă maximul e fiu, se interschimbă cu tatăl.)
Vrd-8 - Metoda rapidă (QuickSort)
Vrd-9 - Metoda prin interclasare.

Ordonare prin dansuri :-)
https://www.youtube.com/user/AlgoRythmics/videos
https://www.youtube.com/watch?v=ulPrQYkd5Oc - HeapSort cu Mario :-)
https://www.youtube.com/watch?v=EreoMaOBTzE - HeapSort

Vrd-10 - Să se ordoneze crescător prima jumătate a vectorului şi descrescător a doua jumătate.
Vrd-11 - Să se ordoneze elementele pare din vector, cele impare ramânând neschimbate.
Vrd-12 - Să se ordoneze elementele din vector după ultima cifră a fiecărui element.
Vrd-13 - Să se ordoneze elementele din vector după prima cifră a fiecărui element.
Vrd-14 - Să se numere câte [comparaţii / interschimbări] se execută la [metoda prin interschimbare / metoda bulelor]. Care este numărul maxim / minim de comparaţii care se execută la [metoda prin interschimbare / metoda bulelor].

Exerciţii cu creare vector nou
VN1 - Se citeşte de la tastatură un număr, să se stocheze cifrele numărului într-un vector (exem: 1859 => 9,5,8,1).
VN2 - Se citeşte de la tastatură un vector, să se stocheze în alt vector elementele pare.
VN3 - Să se stocheze în alt vector elementele prime.
VN4 - Să se stocheze elementele în alt vector, fără a se repeta (exem: 7,2,2,7,7,2,8,5,5 => 7,2,8,5).
VN5 - Să se afişeze de câte ori se repetă fiecare element.
VN6 - Să se stocheze elementele pare şi următorul al acestuia (exem: 7,4,9,2,6 => 4,5,2,3,6,7).
VN7 - Să se afişeze frecvenţele tuturor digramelor din vector. (Digrama este un număr de fix două cifre.)

Exerciţii diverse
Vx1 - Să se afişeze elementele din vector care apar o singură dată.
Vx2 - Să se afişeze elementele din vector care apar de două ori.
Vx3 - Să se afişeze cele mai mari două valori.
Vx4 - Să se afişeze cea mai mare valoare care se repetă.
Vx5 - Să se afle cmmdc al elementelor vectorului.

Vx6 - Să se determine şi să se afişeze primul număr par din vector. Dacă nu există niciun număr par, se afişează mesaj.
Vx7 - Să se determine şi să se afişeze ultimul număr par din vector. Dacă nu există niciun număr par, se afişează mesaj.
Vx8 - Să se aranjeze elementele unui vector astfel încât elementele pare să fie la inceput, iar cele impare la final.
Vx9 - Să se aranjeze elementele unui vector astfel încât zerourile să fie la final (exem: 7,0,0,4,9,0,2,0 => 7,4,9,2,0,0,0,0).
Vx10 - Să se aranjeze elementele unui vector astfel încât elementele pozitive să fie la inceput, iar cele negative la final.

Vx11 - Să se afişeze dacă un vector este **creastă** sau nu.
Vx12 - Să se afişeze câte creste are un vector.
Vx13 - Se dă un vector de cifre, să se calculeze suma numărului format din elementele vectorului parcurgându-l de la stânga la dreapta, cu numărul format parcurgându-l invers. Exemplu: dacă vectorul are valoarea (1, 8, 5, 9), se adună numerele 1859+9581.
Vx14 - Se citesc numere. Într-un vector să se păstreze ultimele 5 numere citite.
Vx15 - Să se afişeze dacă x există sau nu în vector şi, dacă da, de câte ori şi pe ce poziţii.

Vx16 - Să se aranjeze elementele vectorului astfel: pe poziţii pare elementele cele mai mari în ordine descrescătoare, pe poziţii impare elementele cele mai mari în ordine descrescătoare. Exemplu: 8 9 5 3 6 10 15 2 7 devine 15 2 10 3 9 5 8 6 7.
Vx17 - Să se aranjeze elementele vectorului astfel: numerele negative să fie ordonate descrescător, apoi zerourile, apoi numerele pozitive ordonate descrescător. Exemplu: 1 -41 8 -3 0 4 8 0 -1 0 22 7 -20 devine -1 -3 -20 -41 0 0 0 22 8 8 7 4 1.

Exerciţii de 3 stele ***

Vy1 - Să se determine dacă există un indice pentru care suma primelor i elemente este egală cu suma celorlalte n-i.
Vy2 - Să se afişeze cel mai lung şir de numere aflate în ordine crescătoare (exem: 3,9,8,5,2,3,4,5,4,2,9,10,7 => 2,3,4,5).
Vy3 - Se dă un vector, să se afişeze câte grupuri de elemente egale există în vector şi câte elemente conţine fiecare grup.
Vy4 - Se citesc de la tastatură numere, să se stocheze într-un vector ultimele 5 numere citite.

Să se creeze un vector care să conţină urmatoarele numere:
Vy5 - 1,2,2,3,3,3,4,4,4,4,5,5,5,5,5,...
Vy6 - 1,1,2,1,2,3,1,2,3,4,1,2,3,4,5,...
Vy7 - 1,2,2,3,4,4,5,6,6,7,8,8,...
Vy8 - 1,2,3,4,5,6,7,8,9,1,0,1,1,1,2,1,3, ... Să se afişeze pe grupe astfel: 1 - 23 - 456 - 7891 - 01112 ...

Lucru cu 2 vectori

Se dau doi vectori (a şi b) de dimensiune n.
VD1 - Să se creeze un alt vector care să conţină suma elementelor pe linie a celor doi vectori (c[i]=a[i]+b[i]).
VD2 - Să se calculeze produsul cartezian al celor doi vectori (c[i]=a[i]*b[i]).
VD3 - Să se calculeze produsul scalar al celor doi vectori: <a,b>=suma(a[i]*b[i]).
(Norma calculată în funcţie de produsul scalar: ||v||=sqrt(<v,v>)).
VD4 - Să se afişeze dacă cei doi vectori sunt ortogonali (adică produsul lor scalar este zero).

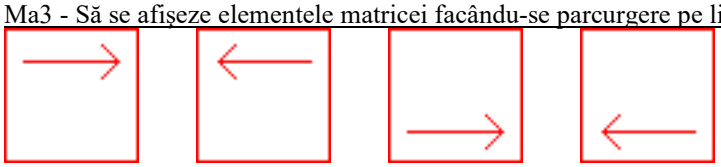
VD5 - Să se schimbe elementele celor doi vectori.
VD6 - Să se adauge vectorul b în coada vectorului a.
VD7 - Să se insereze vectorul b în vectorul a într-o poziţie dată de la tastatură.
VD8 - Să se interclaseze cei doi vectori.

Reuniunea, intersecţia, diferenţa - Exemplu: a=(7,3,7,7,2,1,9,8,2) b=(7,4,9,2,6)
VD9 - Să se afişeze o singură dată toate numerele din vectori (reuniunea).
VD10 - Să se afişeze o singură dată numere comune din vectori (intersecţia).
VD11 - Să se afişeze o singură dată numerele care există în primul vector şi nu există în al doilea (a-b).
VD12 - Să se afişeze o singură dată numerele care există în al doilea vector şi nu există în primul (b-a).

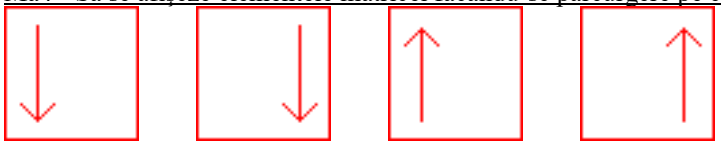
VD13 - Să se afişeze dacă vectorul b este inclus în a.
VD14 - Să se determine dacă un vector este mai mare (maximul unuia este mai mic decât minimul celuilalt).
VD15 - Să se verifice dacă doi vectori sunt proporţionali sau nu şi, dacă da, să se afişeze proporţia.
VD16 - Se dau doi vectori de cifre reprezentând cifrele a două numere mari (de minim 20 cifre). Să adune cele două numere într-un al treilea vector.

Matrici

Ma1 - Să se citească de la tastatură o matrice cu m linii şi n coloane, m şi n citite tot de la tastatură.
Ma2 - Să se afişeze matricea (trecând pe rândul următor după fiecare linie afisata).
Ma3 - Să se afişeze elementele matricei facându-se parcurgere pe linie, pornind din fiecare colţ al matricei.



Ma4 - Să se afişeze elementele matricei facându-se parcurgere pe coloană, pornind din fiecare colţ al matricei.



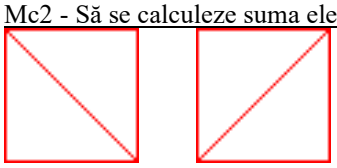
Ma5 - Să se afişeze elementele de pe linia x (unde x se citeşte de la tastatură).
Ma6 - Să se afişeze elementele de pe marginile matricei în sensul acelor de ceas.

Mb1 - Să se calculeze suma elementelor.
Mb2 - Să se calculeze norma unei matrici dreptunghiulare (suma pătratelor elementelor).
Mb3 - Să se calculeze suma elementelor pare aflate pe linii impare.

Mb4 - Să se afişeze câte elemente pare şi câte elemente impare sunt într-o matrice.

Mb5 - Să se afle maximul.
Mb6 - Să se afişeze poziţiile unde se află maximul.
Mb7 - Să se afişeze indicele liniilor unde se află minimul.

Mc1 - Să se calculeze suma elementelor de pe diagonala principală (adică urma unei matrici pătratice).
Mc2 - Să se calculeze suma elementelor de pe diagonala secundară.



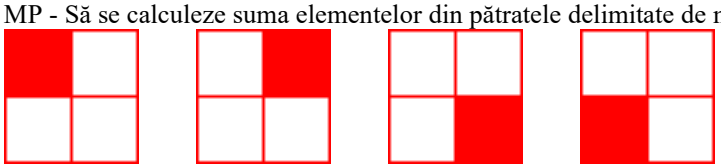
Mc3 - Se dă x de la tastatură, să se afişeze suma pe linia x.
Mc4 - Se dă x de la tastatură, să se afişeze produsul pe coloana x.

Mc5 - Să se calculeze **sumele pe linii**.
Mc6 - Să se calculeze sumele pe coloane.
Mc7 - Maxim / minim pe linii (variante: afişare pe ecran, stocare într-un vector).
Mc8 - Maxim / minim pe coloane (variante: afişare pe ecran, stocare într-un vector).
Mc9 - Să se afle minimul maximelor pe linii.
Mc10 - Căutare număr - Se dă un număr x, să se afle dacă există sau nu în matrice.

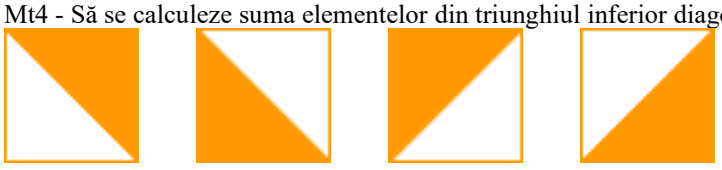
Permutări, transpuneri
Mp1 - Să se interschimbe prima linie cu ultima linie.
Mp2 - Flip vertical - Să se interschimbe liniile astfel: prima se interschimbă cu ultima, a doua cu penultima etc.
Mp3 - Flip orizontal - Să se interschimbe coloanele astfel: prima se interschimbă cu ultima, a doua cu penultima etc.
Mp4 - Rotire verticală - Să se mute ultima linie în faţa celorlalte.
Mp5 - Rotire orizontală - Să se mute ultima coloană în faţa celorlalte.
Mp6 - Să se rotească pe verticală 3 poziţii.
Mp7 - Să se rotească pe orizontală 3 poziţii.

Mp8 - Într-o matrice pătratică să se interschimbe linia 2 cu coloana 3.
Mp9 - Într-o matrice pătratică să se interschimbe linia i cu coloana j.
Mp10 - Să se transpună o matrice.

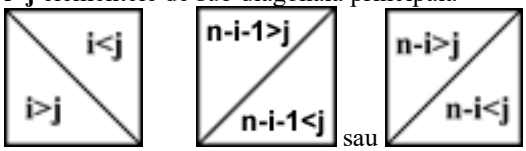
MP - Să se calculeze suma elementelor din pătratele delimitate de mediane.



Triunghiuri mari
Mt1 - Să se calculeze suma elementelor din triunghiul superior diagonalei principale.
Mt2 - Să se calculeze suma elementelor din triunghiul inferior diagonalei principale.
Mt3 - Să se calculeze suma elementelor din triunghiul superior diagonalei secundare.
Mt4 - Să se calculeze suma elementelor din triunghiul inferior diagonalei secundare.



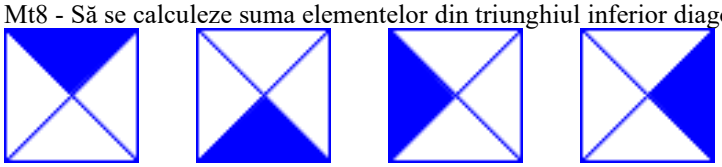
i<j elementele de deasupra diagonalei principale
i>j elementele de sub diagonala principală
n-i>j elementele de deasupra diagonalei secundare
n-i<j elementele de sub diagonala secundară



sau în funcţie de numerotarea matricei, de la 0 sau de la 1

Triunghiuri mici - nord, sud, est, vest

Mt5 - Să se calculeze suma elementelor din triunghiul superior diagonalelor.
Mt6 - Să se calculeze suma elementelor din triunghiul inferior diagonalelor.
Mt7 - Să se calculeze suma elementelor din triunghiul inferior diagonalei principale şi superior diagonalei secundare.
Mt8 - Să se calculeze suma elementelor din triunghiul inferior diagonalei secundare şi superior diagonalei principale.



Mt9 - Să se calculeze suma elementelor din matrice, mai puţin triunghiul inferior diagonalelor.
Mt10 - Să se calculeze suma elementelor benzii diagonal principală de rază 2 a matricei.

Construire matrici

Md0 - Să se construiască matricile pătratice următoare (numărul de linii şi coloane se dă de la tastatură, în exemple n=4)

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	1 2 3 4 8 7 6 5 9 10 11 12 16 15 14 13	1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31	1 1 1 1 3 3 3 3 5 5 5 5 7 7 7 7	1 4 9 16 25 36 49 64 81 100 121 144 169 196 225 256	0 1 0 1 1 0 1 0 0 1 0 1 1 0 1 0	1 11 121 1331	0000 0000 1111 1111	1 2 3 4 1 2 3 4 1 2 3 4 2 3 4 5
16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1	4 3 2 1 4 3 2 1 4 3 2 1 4 3 2 1	1 2 3 4 3 5 7 9 6 9 12 15 10 14 18 22	6 7 8 9 12 14 16 18 18 21 24 27 24 28 32 36	1 1 2 3 5 8 13 21 34 55 89 ... (Şirul Fibonacci)	1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7	14641 (Triun- ghiul lui **Pascal**)	0000 0000 1111 1111	2 3 4 5 2 3 4 5 3 4 5 6 3 4 5 6
1 1 1 1 1 2 2 2 1 2 3 3 1 2 3 4	1 2 3 4 5 6 7 0 8 9 0 0 10 0 0 0	1 0 0 0 2 3 0 0 4 5 6 0 7 8 9 10	22222222 00444444 00006666 00000088	1 2 3 4 2 3 4 5 3 4 5 6 4 5 6 7	1 2 3 4 2 3 4 1 3 4 1 2 4 1 2 3	7 6 5 4 6 5 4 3 5 4 3 2 4 3 2 1		3 4 5 6 4 5 6 7 4 5 6 7 4 5 6 7

Diverse

Md1 - Să se afişeze liniile / coloanele care au toate elementele numere pare / impare / prime / pătrate perfecte.
Md2 - Să se afişeze dacă toate elementele unei matrici sunt diferite între ele.
Md3 - Să se ordoneze elementele de pe diagonala principală / secundară.
Md4 - Să se calculeze produsul cartezian între linia i şi coloana i şi rezultatul să se stocheze într-un vector.
Md5 - Să se afişeze dacă **vreun** element de pe diagonala principală este mai mare decât suma celorlalte elemente din linie.
Md6 - Să se afişeze dacă **fiecare** element de pe diagonala principală este mai mare decât suma celorlalte elemente din linie.
Md7 - Se dă o matrice pătratică, să se afişeze dacă aceasta este un pătrat magic. (Suma pe fiecare linie este egală cu suma pe fiecare coloană şi cu suma pe fiecare diagonală).
Md8 - Să se afişeze maximul minimelor liniilor.

Mx1 - Să se calculeze produsul a două matrici de dimensiuni m*n, respectiv n*p.
Mx2 - Să se calculeze matricea A la puterea n.

Matrici rare

Mr1 - Să se calculeze gradul de umplere al unei matrici.
Mr2 - Să se convertească o matrice în format de memorare cu trei vectori.
Mr3 - Se dă numărul de linii şi coloane al matricei, precum şi numărul de elemente din fiecare vector. Să se convertească o matrice rară memorată în trei vectori într-o matrice normală.
Mr4 - Să se afişeze matricea rară memorată în vectori.
Mr5 - Să se transpună o matrice rară.
Mr6 - Să se adune două matrici rare.
Mr7 - Să se normalizeze o matrice rară (eliminarea elementelor nule din vectorul de valori).

Matricea de permutare - are un singur 1 pe fiecare linie şi pe fiecare coloană, în rest, valori nule. Se stochează în vectori index_linii sau index_coloane.
Matricea triunghiulară - are elemente nenule pe şi deasupra diagonalei principale şi nule dedesupt.

Se dă o matrice şi un vector

Mv1 - Să se treacă elementele de pe linia x într-un vector.
Mv2 - Primele n*n elemente din vector (care are t elemente) să se treacă într-o matrice pătratică.
Mv3 - Să se calculeze produsul dintre matrice şi vector.

Mv4 - Să se determine dacă fiecare linie a unei matrici este proporţională sau nu cu un vector dat.
Mv5 - Să se determine liniile pentru care produsul elementelor este mai mic decât produsul elementelor vectorului obţinut prin reducerea dimensiunii vectorului dat la indicii liniilor.

Şiruri (vectori de caractere)

Funcţii de lucru cu şiruri în C++ ~~cin~~, cin.get(s,50), cin.getline(s,50), scanf, gets - cout, printf str**len**, str**cpy**, str**cat**, str**cmp**, str**n**cpy, str**n**cat, str**n**cmp, str**i**cmp, http://infoscience.3x.ro/c++/siruridecaractere.htm	str**chr**, str**r**chr, str**str**, str**tok**, str**rev**, str**lwr**, str**upr**, i**to**a(x,s,10); x=a**to**i(s); cout<<s[2]; cout<<*(s+2); cout<<s+2; Declarare **vector de şiruri**: char vs[10][255];

Să se citească de la tastatură un şir s. Exemplu: şirul "UNIVERSITATEA CRAIOVA".
Sa1 - Să se afişeze dacă primul caracter din şir este literă sau nu.
Sa2 - Să se afişeze dacă primul caracter din şir este vocală sau nu.
Sa3 - Să se afişeze lungimea şirului.
Sa4 - Să se şteargă din şir 3 caractere, începând din poziţia 4.
Sa5 - Se citesc de la tastatură 2 şiruri, să se introducă al doilea şir în primul şir în poziţia a 4-a.
Sa6 - Se citesc de la tastatură 2 şiruri, să se ordoneze alfabetic.
cin.get();
scanf("%[^\n]s",&s);
*scanf("%*c");*

Parcurgeri caracter cu caracter
Sb1 - Să se afişeze şirul de la coadă la cap.
Sb2 - Să se afişeze de câte ori apare în şir caracterul 'a'.
Sb3 - Să se afişeze câte vocale conţine şirul.
Sb4 - Să se afişeze câte cifre conţine şirul.
Sb5 - Să se elimine primul caracter din şir.
Sb6 - Să se elimine ultimul caracter din şir.
Sb7 - Să se transforme majusculele în litere mici şi literele mici în majuscule.
Sb8 - Să se afişeze a doua jumătate a şirului.
Sb9 - Să se citească de la tastatură două şiruri şi să se determine cel mai lung sufix comun al lor.
Sb10 - Să se citească de la tastatură două şiruri şi să se scrie algoritmul care realizează adunarea acestora în zecimal.

Determinare cuvinte
Se dă un şir de caractere în care cuvintele sunt separate cu un singur spaţiu.
Sc1 - Să se afişeze câte cuvinte conţine şirul.
Sc2 - Să se afişeze lungimile cuvintelor din şir.
Sc3 - Să se afişeze cuvintele din şir.
Sc4 - Să se transforme în majusculă prima literă a fiecărui cuvânt.
Sc5 - Să se numere câte cuvinte din şir conţin litera "a".
Sc6 - Să se ordoneze alfabetic cuvintele din şir.
Aceleaşi enunţuri pentru cazul în care cuvintele sunt separate prin orice combinaţie de caractere.

Sd1 - Să se dubleze vocalele.
Sd2 - Se dă un şir, de exemplu "abcd", să se afişeze "abbcccdddd".
Sd3 - Se dă un şir, de exemplu "abcdef", să se transforme şirul în "defabc". Dacă are număr impar de caractere, cel din mijloc rămâne pe loc, adică "1234567" devine "5674123".
Sd3 - Să se verifice dacă un şir este palindrom. (Exemple: rotitor, reper, cojoc, unu, capac, aerisirea, caiac, tivit, soios.)
Palindrom provine de la cuvintele grecești palin (πάλιν - înapoi) și dromos (δρόμος - drum, direcție)
Sd4 - Să se verifice dacă un şir este palindrom, fără a se ține cont de spații sau alte caractere. (Exemple de șiruri palindrom: Ele fac cafele. O ramă maro. Ele ne seduc cu desenele. Era sa pozez o pasare. Era o tipa rapitoare. Maria bea, ca e bairam.)
Sd5 - Se citesc n şiruri, să se afişeze câte sunt de tip palindrom.
Diferența între palindrom și polindrom: Palindromul este un cuvânt sau o frază care poate fi citită atât de la stânga la dreapta, cât și de la dreapta la stânga, sensul rămânând același. Polindromul este cuvântul care citit de la coadă la cap are un alt înțeles. Exemple: animate - etamina, analog - golana, Roma - amor.
Sd6 - Să se verifice dacă un şir este dublu palindrom. Un șir este dublu palindrom este un șir palindrom care are număr par de caractere, iar fiecare caracter de rang impar este egal cu caracterul alăturat din dreapta lui. Exemplu: aabbaa.

Se1 *** - Să se afişeze astfel şirul "EDITURA":

E	*	EDITURA	*******
ED	**	EDITUR	******
EDI	***	EDITU	*****
EDIT	****	EDIT	****
EDITU	*****	EDI	***
EDITUR	******	ED	**
EDITURA	*******	E	*

Se2 - Să se afişeze şirul astfel:

E E D I E D I T U E D I T U R A	E D I T U R A E D I T U E D I E	T I T U D I T U R E D I T U R A	E D I T U R A D I T U R I T U T

Se3 - Să se afişeze şirul astfel: EDIT, DITU, ITUR, TURA.
Să se testeze algoritmii pentru şirul “UNIVERSITATEA”.

Se4 *** - Să se citească de la tastatură caractere şi, într-un vector, să se stocheze primele p paranteze drepte citite. Să se verifice dacă şirul este corect, adică:
1/ numărul de paranteze deschise este egal cu numărul de paranteze închise;
2/ orice paranteză închisă are în stânga ei un număr mai mare de paranteze deschise decât închise.
Să se afişeze câţi muguri sunt în vector (un mugur e o paranteză deschisă urmată de una închisă). Să se afişeze şirul cu o culoare, iar mugurii să fie afişaţi cu altă culoare.

Fişiere

Fişiere text

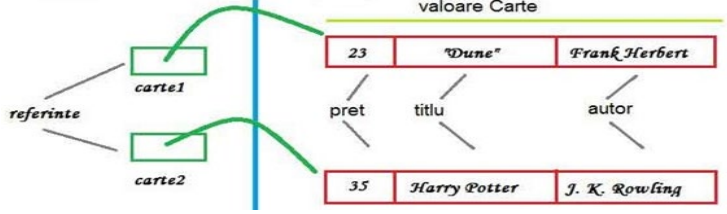
Ft1 - Fişierul text "date.txt" conţine două numere. Să se citească numerele din fişier şi să se afişeze pe ecran media lor.
Ft2 - Fişierul text "date.txt" conţine numere. Să se citească numerele din fişier şi
a) să se afişeze pe ecran numerele citite.
b) să se afişeze pe ecran media lor.
c) să se scrie în fişierul "rezultat.txt" de câte ori apare fiecare cifră din sistemul zecimal.
Ft3 - Fişierul text "date.txt" conţine pe prima linie valoarea n, iar pe a doua linie n numere.
a) Să se citească numerele şi să se afişeze pe ecran produsul şi media lor.
b) Să se citească numerele şi să se scrie ordonate crescător în fişierul "rezultat.txt".
Ft4 - Fişierul text "date.txt" conţine pe prima linie valoarea n şi un număr x, iar pe a doua linie n numere.
a) Să se citească numerele de pe a doua linie şi să se afişeze pe ecran câte numere sunt egale cu x.
b) Să se citească numerele de pe a doua linie şi să se scrie în fişierul "rezultat.txt" câte numere au ultima cifră egală cu x.

Fişiere cu tip

Fp1 - Se dă un fişier cu elemente de tip persoană, să se ordoneze conţinutul acestuia după nume.
Fp2 - Să se realizeze un program de gestiune [a unei agende telefonice / al unui top muzical / al unor facturi].
Operaţii de bază: Introducere / Afişare / Modificare / Ştergere / Căutare.

Tipul de dată articol (înregistrare)

Ar1 - Să se realizeze o structură de tip masina, conţinând câmpurile: cod, marca, model, an_fabricatie, consum.
Ar2 - Să se realizeze o structură de tip carte, conţinând câmpurile: pret, titlu, autor. Să se declare şi să se citească de la tastatură două variabile denumite c1 şi c2, apoi să se calculeze şi să se afişeze media preţurilor lor.



Ar3 - Să se realizeze o structură de tip fracţie. Să se citească de la tastatură şi să se adune două fracţii.
Ar4 - Să se realizeze o structură de tip complex, conţinând câmpuri pentru partea imaginară şi partea reală. Să se citească de la tastatură două astfel de numere şi să se afişeze suma, diferenţa şi produsul lor.

Structură în structură

Ar5 - Să se realizeze o structură de tip elev, conținând câmpurile: cod, nume, prenume, data_inmatriculare și data_nastere, ultimele două câmpuri fiind declarate ca o structură separată numită calendar și având câmpurile zi, luna, an.
5.1 - Să se declare o variabilă e1 de tip elev și să se citească de la tastatură.
5.2 - Să se interschimbe data_înmatriculare cu data_nastere.

Vector de structură
Ar6 - Să se declare un vector v de 10 elemente de tip elev.
6.1 - Să se citească de la tastatură valorile din primul element.
6.2 - Să se afișeze primul caracter din numele celui de al doilea elev.
6.3 - Să se afișeze primul caracter al fiecărui nume din vectorul de elevi.
6.4 - Să se ordoneze elevii după nume / data nașterii și să se afișeze rezultatul.

Structură cu vector
Ar7 - Să se realizeze o structură de tip student, conținând câmpurile: cod, nume, prenume, notele la 20 de materii.
Să se declare o variabilă denumită s1.
7.1 - Să se citească de la tastatură valorile ei.
7.2 - Să se afișeze nota la a doua materie.
7.3 - Să se afișeze notele la toate materiile.

Alte exerciții
Ar8 - Să se realizeze o structură prin care se pot crea trei variabile care să rezolve soluția pentru instrucțiunile:
c.x=a.x+b.x;
c.y=a.y+b.y;
Ar9 - Formula distanței dintre două puncte: $M_1(x_1,y_1), M_2(x_2,y_2)$ este $M_1M_2 = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$
Ar10 - Coordonatele mijlocului segmentului [M_1M_2] sunt: $x = \frac{x_1+x_2}{2}, y = \frac{y_1+y_2}{2}$

Subprograme - funcții și proceduri

Noțiuni: antet, corp, apel, parametri, variabile locale, variabile globale, transmisie prin valoare, transmisie prin adresă.
Moduri de apel al unei funcții void / cu tip. Vizibilitate între două funcții care se apelează una pe cealaltă.
Parametri din antetul unei funcții se numesc parametri formali, iar cei din apelul funcției se numesc parametri efectivi.

Să se realizeze un subprogram care:
Sb1 - primește 4 numere reale și returnează cea mai mare valoare dintre ele.
Sb2 - primește un număr întreg (long) și returnează dacă este prim sau nu (1 sau 0, true sau false).
Sb3a - primește două numere și returnează cmmdc-ul lor.
Sb3b - primește două numere și returnează cmmmc-ul lor.
Sb4a - primește un număr întreg și returnează inversul lui.
Sb4b - primește un număr întreg și verifică dacă este palindrom. Să se verifice dacă un număr este superpalindrom (exemplu: 11, care are pătratul, 121, tot palindrom).
Sb5 - primește ca parametru un număr real și returnează partea zecimală ca număr întreg.
Sb6 - primește trei numere și returnează 1 dacă primul număr este între celelalte două și 0 dacă nu.
Sb7 - primește două numere și afișează numerele pare dintre ele. Dacă primul e mai mare decât al doilea, se interschimbă.
Alte exerciții pentru funcții: cele de la pagina 3 (tipuri de număr).

Transmisie prin referință/pointer/adresă
Sp1 - interschimbă prin metoda paharelor conținutul a două variabile primite ca parametri.
Sp2 - primește două numere și le returnează ordonate crescător.
Sp3 - primește două numere și returnează cel mai mare și cel mai mic dintre ele prin alți parametrii, în această ordine.
Sp4 - primește două numere și returnează prin alți parametri cmmdc-ul și cmmmc-ul lor.
Sp5 - primește două numere și returnează prin alți parametri suma, diferența și produsul lor.
Sp6 - primește un vector de numere întregi și îl ordonează

Recursivitate

Rc1 - Se citește de la tastatură un număr (n). Să se calculeze n factorial.
Rc2 - Folosind funcția factorial,
să se afișeze aranjamente de n luate câte k, A(n,k) și.combinări de n luate câte k, C(n,k).
să se calculeze combinări de n luate câte k, C(n,k).

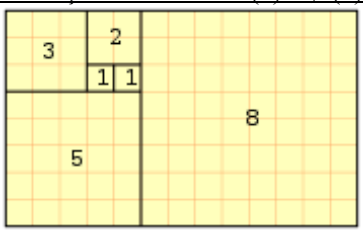
$A_n^k = \frac{n!}{(n-k)!}$ $C_n^k = \frac{n!}{k!(n-k)!}$ $n! = \begin{cases} 1 & , n = 0, \\ (n-1)! \times n & , n > 0. \end{cases}$

Rc3 - Să se calculeze aranjamente de n luate câte k, A(n,k), folosind formula de recurență.
$A_n^k = (n-k+1) \cdot A_n^{k-1}$, unde $A_n^0 = 1$

Rc4 - Să se calculeze combinări de n luate câte k, C(n,k), folosind o formulă de recurență.
$C_n^k = C_n^{k-1} \frac{n-k+1}{k}$ $C_n^k = C_{n-1}^k \frac{n}{n-k}$ $C_n^k = C_{n-1}^{k-1} \frac{n}{k}$ $C_n^k = C_{n-1}^{k-1} + C_{n-1}^k$, unde $C_n^0 = 1$

C(3,2) = ab, ac, bc
A(3,2) = ab, ba, ac, ca, bc, cb
P(3) = abc, acb, bac, bca, cab, cba

Rc5 - Şirul Fibonacci. f(1)=1, f(2)=1, f(n)=f(n-1)+f(n-2) pentru n>=3. Să se afişeze termenul n.



Rc6 - Să se calculeze 1+2+3+4+5...+n.
Rc7 - Să se calculeze 1*2 + 2*3 + 3*4 + 4*5 + ... + n*(n+1).
Rc8 - Să se calculeze 1-2+3-4+5-6+...+n.

Rc9 - Suma cifrelor unui număr.
Rc10 - Să se afişeze divizorii proprii ai unui număr x citit de la tastatură.
Rc11 - Se dau a şi n, să se calculeze a la puterea n.
Rc12 - Cmmdc a două numere.

Rv1 - Suma elementelor unui vector.
Rv2 - Produsul elementelor.
Rv3 - Maximul elementelor.
Rv4 - Căutare număr în vector.
Rv5 - Suma elementelor dintre al 2-lea şi al 5-lea din vector.

Algoritmi

Divide et impera ("D&i")

Di1 - Se dă un vector ordonat v şi un număr x, să se afişeze dacă x există sau nu în vector folosind **căutarea binară**.
Di2 - Se dă un vector, să se calculeze suma elementelor lui.
Di3 - Se dă un vector, să se calculeze produsul elementelor lui.
Di4 - Se dă un vector, să se afle maximul său.

Backtracking

Bk1 - Se dă un număr x, să se afişeze P(x) = permutari de x (Exemplu pt. x=3: 123, 132, 213, 231, 312, 321).
Bk2 - Se dau n şi k, să se afişeze aranjamente de n luate câte k. *Să se afişeze numărul total de soluţii găsite.*
Bk3 - Se dau n şi k, să se afişeze combinari de n luate câte k.
Bk4 - Se dă o cifră x, să se afişeze submulţimile formate din cifrele până la x. (Exemplu pt. x=3: 1, 2, 3, 12, 13, 23, 123).

Bk5 - Într-un vector se dau monedele ordonate descrescător. Se dă o sumă, să se afişeze variantele de plată a sumei.
Să se afişeze numărul total de soluţii găsite.
Bk5b - Să se afişeze variantele de plată a sumei în numărul minim de monede.
Bk6 - Se dă un număr x, să se afişeze permutările cifrelor sale. (Varianta 1: cifrele sunt diferite. Var 2: cifrele pot fi egale.)
Bk7 - Se dă un număr x, să se afişeze submulţimile formate din cifrele sale.

Bg1 - Problema săriturii calului: Să se afişeze poziţiile pe care poate sări un cal pe o tablă de şah de dimensiune n*n astfel încât să sară o singură dată pe toate pătrăţelele. (Indicație: stiva e alcătuită din 2 vectori, nu dintr-o matrice!)
Bg2 - Problema labirintului. Se dă de la tastatură o poziție de plecare (x1 și y1) și alta de ieșire (x2,y2). Trebuie să se găsească drumurile de la x1,y1 la x2,y2. De exemplu, matricea poate fi așa:
0001101
1001000
1100011
1001001
1110000
0011001
0110010
Bg3 - Problema colorării hărţilor.
Bg4 - Problema canibalilor şi a misionarilor.
Bg5 - Problema damelor: Să se aşeze n dame pe o tablă de şah de dimensiune n*n fără ca acestea să se atace între ele. (Indiciu soluţie: să nu fie poz-i == abs(v[poz]-v[i]) , ceea ce ar însemna că sunt pe aceeași diagonală)

Greedy
Gr1 - Problema continuă a rucsacului - cu fracţionare.
Gr2 - Problema discretă a rucsacului - fără fracţionare.
Gr3 - Problema cu utilitatea economică.
Gr4 - Se dau distanţele intermediare între benzinăriile dintre două localităţi, se ştie capacitatea rezervorului maşinii şi consumul ei la suta de km. Să se stabilească drumul cu cele mai puţine opriri pentru alimentare.

Pointeri (adrese / referințe)

Operatorul new poate aloca numai masive unidimensionale. Pentru alocarea masivelor cu mai multe dimensiuni (care să poată fi accesate cu sintaxa obișnuită) se vor utiliza vectorii de pointeri.
Exemplu de alocare pentru matrice:

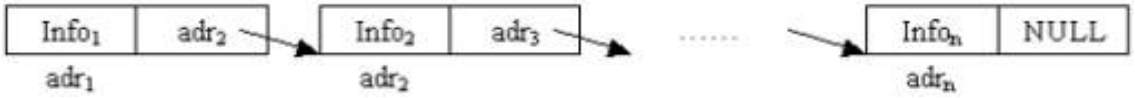
```
int m = 3, n = 4;  // dimensiunile matricei
int **mat;  // declararea matricei ca pointer
mat = new int* [m]; // alocarea vectorului de pointeri
for (int i = 0; i < m; i++)
{ mat[i] = new int[n]; } // alocarea vectorilor pentru fiecare linie
```

Elementele matricei astfel alocate pot fi accesate folosind sintaxa obișnuită: mat[i][j].
Dezalocarea se va face urmand succesiunea pasilor in ordine inversa:

```
for (int i = 0; i < m; i++)
{ delete [] mat[i]; } // dezalocare vectori pentru fiecare linie
delete [] mat; // dezalocare vector de pointeri
```

Liste înlănţuite

LSI (listă simplu înlănțuită)
Tipuri de liste:
- stivă (nodurile noi se introduc în fața celorlalte existente)
- coadă (nodurile noi se introduc după cele existente)



Creare statică LSI de tip coadă / stivă
Lc1 - Să se creeze o listă simplu-înlănțuită din 3 noduri, a,b,c.
Lc2 - Să se adauge un nod d la sfârșitul listei.
Lc3 - Să se insereze un nod e între b și c.
Lc4 - Să se șteargă nodul c.
Lc5 - Să se interschimbe nodul b cu nodul d.
Lc6 - Să se interschimbe informația din nodul 1 cu informația din nodul 3.
Lc7 - Folosind numai adresa nodului 1, să se afișeze informațiile din nodurile 1, 2 și 3.
Lc8 - Să se creeze o listă circulară.

Parcurgere dinamică LSI
Lp1 - Să se afişeze o listă simplu-înlănţuită.
Lp2 - Să se calculeze suma informaţiilor dintr-o listă simplu-înlănţuită.
Lp3 - Să se calculeze câte noduri conţine lista.
Lp4 - Să se calculeze media informaţiilor din listă.
Lp5 - Să se afle cel mai mare număr dintr-o listă simplu-înlănţuită.
Lp6 - Să se afişeze informaţia din ultimul nod / din penultimul nod.
Lp7 - Se dă de la tastatură un număr x, să se verifice dacă există în listă. Dacă nu există, să se adauge la sfârşitul listei.
Lp8 - Să se afişeze informaţiile pare dintr-o listă simplu-înlănţuită.
Lp9 - Să se afişeze informaţiile prime dintr-o listă simplu-înlănţuită.

Creare dinamică LSI de tip coadă
Ld1 - Se citesc numere până se introduce valoarea 0, să se creeze dinamic o listă simplu-înlănţuită.
Ld2 - Să se creeze o listă care să conţină primele 10 numere pare şi strict pozitive.
Ld3 - Să se creeze o listă care să conţină n numere citite de la tastatură.
Ld4 - Se dă un vector v, se dă un număr x, să se caute x în vector şi să se stocheze într-o listă poziţiile unde este găsit.
Ld5 - Să se creeze o listă care să conţină numerele prime până la un număr x citit de la tastatură.
Ld6 - Se citesc numere până se introduce valoarea 0, să se creeze dinamic o listă dublu-înlănţuită.

Inserări
Li1 - Se dă adresa nodului de început al unei LSI, să se adauge un nod la începutul ei.
Li2 - Se dă adresa nodului de început al unei LSI, să se adauge un nod la sfârşitul ei.
Li3 - Se dă adresa nodului de început al unei LSI, să se adauge un nod înainte de ultimul ei nod.
Li4 - Să se insereze un nod într-o listă simplu-înlănţuită într-o poziţie t introdusă de la tastatură.
Li5 - Să se insereze un nod cu informaţia 0 după primul număr par găsit.
Li6 - Să se adauge suma ca informaţie într-un nod la finalul listei.

Ştergeri
Ls1 - Să se şteargă dintr-o listă simplu-înlănţuită nodul de pe o poziţie t citită de la tastatură.
Ls2 - Să se şteargă primul nod dintr-o LSI (dacă e funcţie, să returneze adresa actualului prim nod).
Ls3 - Să se şteargă primul nod cu informaţia cu valoare pară dintr-o LSI.
Ls4 - Să se şteargă nodurile care au informaţii cu valori pare de la începutul unei LSI (până se întâlneşte un număr impar).
Ls5 - Să se şteargă toate nodurile cu valori pare dintr-o LSI.

Alte exerciţii
Lx1 - Să se verifice dacă informaţiile dintr-o LSI sunt în ordine crescătoare.
Lx2 - Să se ordoneze o LSI, folosind metoda simplă.
Lx3 - Să se ordoneze o LDI, folosind metoda bulelor.
Lx4 - Se ştie o adresă din listă, să se numere câte noduri are o listă circulară.
Lx5 - Într-o altă listă să se salveze numerele pare din prima listă.

Tabele de dispersie (HashTable / Listă de liste)
Lh1 - Se dă un fişier text care conţine date despre un graf. Să se creeze listele de adiacenţă ale acestuia.
Lh2 - Se dă dimensiunea (5), se dă n, apoi n numere (22 4 2017 11 8 2002 46 4 2 6 105), să se creeze tabela de dispersie. (Lista 1 conţine numerele care încep cu 1 sau 6, lista 2 pe cele care încep cu 2 sau 7 şamd.)

Stiva (de cărţi)

St1 - Să se simuleze folosirea unei stive cu ajutorul unui vector.
St2 - Să se simuleze folosirea unei stive cu ajutorul unei liste dublu înlănţuite.

Coada (la pâine)

C1 - Să se simuleze folosirea unei cozi cu ajutorul unui vector.
C2 - Să se simuleze folosirea unei cozi cu ajutorul unei liste simplu înlănţuite.

Grafuri

Se dă un graf (X,G), X=mulţimea nodurilor, G=mulţimea relaţiilor dintre noduri.
Exemple de grafuri: Neorientat: 1-2, 2-3, 3-4, 2-5, 4-5, 1-6, 1-7, 3-7, 2-8, 5-8.
Orientat: 1-2, 2-3, 3-2, 3-4, 4-3, 4-5, 6-5, 2-4, 1-4, 1-7, 7-1, 2-8, 4-8, 5-8, 8-6.

Metode de reprezentare:
matricea de adiacenţă
listele de adiacenţă
fişierul text
matricea de incidență (vârfuri-muchii)

Noţiuni: grafuri neorientate, grafuri orientate sau digrafuri, p_graf (poate avea p muchii între aceleaşi două noduri), nod (vârf), muchie (la graf neorientat), arc (la graf orientat), bucla (arc de la nodul x la nodul x), ordin graf (numărul de noduri), **grad de incidenţă** (interior, exterior), nod izolat, nod terminal (sau nod suspendat).

Drum (sau cale), *lanţ*, circuit, *ciclu*, drum/lanţ/circuit/ciclu **elementar**, drum/lanţ/circuit/ciclu **simplu**.
Subgraf, graf parţial, componente conexe (sau simplu conexe), componente tare conexe.
Ciclu hamiltonian este un ciclu elementar care conține toate nodurile unui graf.

Tipuri de grafuri: conex (sau simplu conex), tare conex, complet - câte muchii are graful complet neorientat, complementar, hamiltonian, eulerian,
regulat, aciclic, bipartit (sau bigraf), bipartit complet, planar, orientat transpus, orientat turneu, suport.
Atenție! **Nodul izolat** poate fi graf complet!

Elementar: fiecare **nod** apare o singură dată. Opusul: **neelementar**.
Simplu: fiecare **muchie** apare o singură dată. Opusul: **compus**.

Un **subgraf** se obţine prin eliminarea minim a unui **vârf**.
Un **graf parţial** se obţine prin eliminarea minim a unei **muchii**.

O **componentă** este **(simplu) conexă** dacă oricare două vârfuri sunt conectate printr-un **lanţ**.
O **componentă** este **tare conexă** dacă oricare două vârfuri sunt conectate printr-un **drum**.
Atenție! **Nodul izolat** este considerat componentă conexă!

Graf hamiltonian: are un **ciclu elementar** care trece **prin toate nodurile** sale.
Condiţia de suficienţă: Dacă un graf cu n>=3 vârfuri, astfel încât gradul x al oricărui vârf verifică inegalitatea: gr(x)>=n/2, atunci graful este graf hamiltonian.
Graf eulerian: are un **ciclu simplu** care trece **prin toate muchiile** sale.
Condiţia de suficienţă: Un **graf este eulerian** dacă este conex şi gradele tuturor vârfurilor sunt pare și pozitive.
(Exemplu: graful papion sau fluture)

Graf regulat: are toate gradele nodurilor numere pare (posibil și zero).
Graf aciclic: nu are cicluri (este pădure). (Obs: Poate să nu fie conex.)
Graf bipartit: nodurile se împart în două mulțimi, astfel încât fiecare muchie leagă un nod din prima mulțime cu un nod din a doua. **Graf bipartit complet**: De la fiecare nod dintr-o mulțime există muchie către fiecare nod din cealaltă mulțime.
Graf planar: poate fi desenat în așa fel încât oricare două muchii să nu se intersecteze. Opusul lui: **graf neplanar**.
Graf orientat transpus: arcele care nu sunt dus-întors se inversează
Graf orientat turneu: între oricare două noduri există un singur arc (ori dus, ori întors)
Graful suport: este graful obținut prin înlocuirea muchiilor multiple printr-una simplă

Numărul de grafuri complete neorientate este n(n-1)/2.
Numărul de grafuri complete orientate este 3 la n(n-1)/2.

Exerciţii (variante: pentru graf orientat / neorientat)
Ga1 - Se dau de la tastatură numărul de noduri, numărul de arce, apoi arcele, să se creeze matricea de adiacenţă a grafului.
Ga2 - Se dă matricea de adiacenţă, să se creeze fişierul text.
Ga3 - Se dă matricea de adiacenţă, să se creeze listele de adiacenţă.
Ga4 - Se dă fişierul text, să se afișeze listele de adiacenţă.
Ga5 - Se dă fişierul text, să se creeze matricea de adiacenţă.
Ga6 - Se dă fişierul text, să se creeze listele de adiacenţă.
Ga7 - Se dau listele de adiacenţă, să se creeze matricea de adiacenţă.

Ga8 - Se dau listele de adiacenţă, să se creeze fişierul text.

Gb1 - Să se afişeze gradele de incidenţă [interioare / exterioare] ale nodurilor unui graf orientat.
Gb2 - Să se afişeze nodurile cu gradul de incidenţă [exterior / interior] cel mai mare.
Gb3 - Se dă de la tastatură un nod, să se afişeze toate nodurile către care acesta are arce. [Variantă: de la care ...]
Gb4 - (Pentru graf orientat) Să se afişeze nodurile cu gradul de incidenţă interior egal cu cel exterior.
Gb5 - Să se determine matricea grafului complementar.

Gd1 - Să se realizeze matricea drumurilor.
Gd2 - Să se determine componentele conexe / tare conexe.
Gd3 - Parcurgerea în lăţime (BF - Breadth First)
Gd4 - Parcurgerea în adâncime (DF - Depth First)
Gd5 - Să se determine dacă graful este tare conex.
Gd6 - Să se determine dacă graful este aciclic.
Gd7 - Roy Warshall - determină drumurile cele mai scurte între noduri - lucrează cu matricea de adiacenţă

Grafuri ponderate
Gr1 - Matricea ponderilor
Gr2 - Dijkstra - Să se determine distanţa cea mai mică de la un nod la celelalte.
Exemple de grafuri pentru Dijkstra

6	1 5 30	3 4 30	7	2 3 35	4 6 15	3 6 30	1 6 10
1 3 40	2 3 15	4 6 25	1 2 40	2 4 35	4 7 55	2 6 15	1 7 90
1 2 20	2 4 70	5 6 20	1 3 80	2 5 95	6 7 35	3 5 20	

Gr3 - Roy Floyd - determină drumurile cele mai scurte - e la fel ca Roy-Warshall, dar lucrează cu matricea ponderilor
Gr4 - AdrianT (Se folosește BF. Se crează vectorul b în care se țin nodurile parcurse, vectorul d de distanțe, vectorul t de tați. Se adaugă fii unui nod în b, distanța în d și tatăl în t.)

Gx1 - Se consideră un graf neorientat cu 8 noduri numerotate de la 1 la 8 și următoarele muchii: [1,7], [1,8], [3,4], [3,5], [3,6], [3,7], [4,7], [5,6], [5,8], [6,7], [6,8], [7,8]. Precizați care este numărul minim de culori cu care pot fi colorate nodurile grafului, astfel încât oricare două noduri adiacente să aibă culori diferite.

Arbori

- Definiţie: graf conex (deci neorientat) şi fără cicluri.
- Noţiuni: câte muchii are, rădăcină, tată (ascendent direct), ascendent, descendent direct (fiu), descendent, frunze, nivel (de obicei, numerotat de la 1), înălţime (numărul de muchii de la rădăcină la cea mai îndepărtată frunză), subarbore, pădure (un graf în care fiecare componentă conexă a sa este arbore), ordinul sau gradul arborelui (valoarea maximă luată de gradul unui nod al arborelui). Obs: Orice arbore este graf bipartit.
- Tipuri de arbori: binar, echilibrat (subarborele stâng are același număr de noduri ca subarborele drept), neechilibrat.
- Metode de reprezentare: cele trei de la grafuri + vectorul de "tați", listă de descendenți (direcți).

Parcurgându-se vectorul de taţi, să se afişeze:
Aa1 - frunzele arborelui
Aa2 - gradele nodurilor
Exemplu (vectorul de taţi): 0 1 1 2 2 3 3 4 8 6 7 7 10 5 16 14 12 17

Arbori binari
- Definiție: arbore în care fiecare nod are maxim doi fii.
- Metode de reprezentare: vectorii "stânga" și "dreapta".
- A.b.strict sau complet: în care fiecare nod are fix 2 fii.
- A.b.plin: în care fiecare nod are fix 2 fii și toate nivelurile sunt complete.
- Parcurgeri: **V**SD (**pre**ordine), S**V**D (**in**ordine), SD**V** (**post**ordine).

Parcurgându-se vectorul de taţi
Ab1 - Să se verifice dacă arborele este binar. Dacă da, să se construiască vectorii Stânga şi Dreapta.
Parcurgându-se vectorii stânga+dreapta
Ab2 - Să se afişeze frunzele arborelui.
Ab3 - Să se afişeze gradele de incidenţă ale nodurilor.

Arbore binar de căutare
Ac1 - Să se construiască un arbore binar de căutare pentru numerele: 25, 6, 74, 11, 46, 3, 80, 92, 10, 11, 5, 7, 17, 89.
Ac2 - Se dă arborele binar de căutare 5, 7, 10, 21, 27, 39, cu 10 fiind rădăcina și 5, 21, 39 noduri terminale. Se cere ordinea de intrare a numerelor astfel încât să se creeze arborele.
Ac3 - Să se afişeze cel mai mic element din arbore. Să se afişeze cel mai mare element din arbore.

Arbori care conţin informaţii în noduri
Cheie = Valoare nod.
Heap de maxim: A.b. complet în care orice tată > orice fiu. Heap de minim: orice tată > orice fiu
Notația poloneză: Se dă 2*x+6/(2-z) în format in-fix, adica in-ordine (SVD). Să se obțină notația poloneză (adică SDV).

Clase şi obiecte

- **proprietăţi / atribute** (=variabile)
- **metode** (=funcţii)

- **constructor**
În C++ și **C#**, constructorul are numele clasei.
În VB.NET, se denumeşte New.
În PHP, se denumeşte: public function __construct()
În PHP, nu se pot declara mai mulţi constructori.
În Python, ci se denumeşte: def __init__()

- **destructor**
În C++, destructorul are numele clasei, precedat de o tildă ~
În PHP, se denumeşte: public function __destruct()
În Java, nu există destructor. Se foloseşte:
@Override
public void finalize() { SOP("Păzea! Se distruge obiectul!"); }

- **încapsularea** (public, privat, protected - funcţii set şi get)
- **polimorfismul** (exem: figură - arie, perimetru, volum / carte de vizită - afişare date angajat sau student sau etc)
- **moştenirea** (exem: persoana - angajat, student / maşină - automobil, tir, microbuz, basculantă / animal - pasăre, felină)
Apel constructor din clasa de bază.
În C++, proprietăţile îşi păstrează nivelul de încapsulare în clasa derivată (public rămâne public, protected rămâne la fel).

- **abstract**
O clasă abstract nu poate fi instanţiată.
- **interfaţă**
Conţine numai definiţiile funcţiilor, nu şi corpurile, care vor fi definite de clasele care implementează interfaţa.
- **virtual**
O clasă abstractă servind drept interfață conține doar funcții pur virtuale, și nici un atribut sau metode obișnuite.

- **static** (exem: număr obiecte dintr-o clasă)
- **template** - T
- **friend** - permite unei metode să acceseze membrii privaţi ai unei clase din exteriorul acesteia
- **immutable** - obiecte constante (care nu mai pot fi modificate după ce au fost instanţiate)
- **dangling pointer** - pointer către un obiect care poate dispărea. Exem: a = new pers(); b = a; delete (a); Acum b punctează către nimic. (dangling = bălăbăneală :-))

Relații între obiecte:
- obiecte server, asupra cărora numai se acționează
- obiecte actor, care numai acționează asupra altor obiecte
- obiecte agent, sunt și server, și actor

În C++: operatorul de rezoluţie sau de vizibilitate ::
Definirea funcţiilor dintr-o clasă se face în afara domeniului de definiţie al clasei, de aceea numele funcţiei trebuie să fie însoţit de numele clasei şi să fie separat de aceasta prin operatorul de rezoluţie sau vizibilitate (::).

- override (supraîncărcare, redefinire)
http://en.cppreference.com/w/cpp/language/operators
Exemple:

cout<<elev; // în c++ void operator<<() { cout<<nume<<" "<<prenume; }	**- toString din Java** @Override public String toString() { ... }

- operator cast
Exem: operator Student()
Exem: operator int() { return this->cod_persoana ; }
http://www.cplusplus.com/doc/tutorial/typecasting/

C#
```
public static implicit operator float(clsAngajat obj)
{
  return (float)obj.salariu;
}
```

Câteva reguli POO în Java
La instanţierea unui obiect dintr-o clasă fiu, se execută întâi constructorul fără parametrii al clasei părinte, apoi constructorul clasei fiu.
Subclasele pot să suprascrie metodele claselor de baza. Metodele suprascrise ascund metodele vechi.
O clasa poate să moştenească doar o altă clasă, dar poate implementa mai multe interfeţe.

Modificator acces CLASA

Modificator acces	**Explicaţii**
Public	access nelimitat
Internal	acces permis doar în clasa sau spatiul de nume în care e cuprinsa
Protected	acces în clasa curenta sau în cele derivate
Private	implicit. Doar pentru clasele interioare
protected internal	folosit pentru clasele interioare semnificând accesul în clasa care-l conţine sau în tipurile derivate din clasa care-l conţine

Modificator acces PROPRIETĂŢI ŞI METODE

Modificator acces	**Explicaţie**
Public	membrul accesibil de oriunde
Internal	accesibil doar într-un bloc functional al unei aplicatii .Net
Protected	accesibil oricarui membru al clasei care-l conţine şi al claselor derivate
Private	implicit. acces permis doar pentru clasa care conţine membrul
protected internal	accesibil oricarui membru al al clasei care îl conţine şi al celor derivate, dar şi în blocul functional

1. Definiţi o **clasa Complex** a numerelor complexe care conţine două date membre private, re şi im de tip double şi trei funcţii membre public, init(), set() şi display().
Funcţia init() iniţializează datele membre ale clasei cu valoarea 0.
Funcţia set() modifica valorile datelor membre cu valorile transmise ca parametrii.
Funcţia display() afişează partea reala respectiv partea imaginara a numărului complex.
În funcţia main() a programului declaraţi un obiect cu numele c1 de tipul Complex. Pentru acest obiect apelati fiecare dintre funcţiile membre ale clasei.

2. Definiţi mai multe funcţii constructor pentru clasa Complex (constructor fără argumente, cu un argument şi cu două argumente) astfel încât următoarea secvenţă de program să se execute corect.
```
void main()
{
 Complex c1;
 Complex c2(5);
 Complex c3(1,2);
}
```

3. Definiţi **clasa cerc**, având ca proprietate raza şi ca funcţii membre un constructor, aria şi perimetrul.

4. Definiţi **clasa dreptunghi**, având ca atribute lățimea și lungimea și ca funcţii membre un constructor, aria și perimetrul.

SQL

Persoane - oraşe de naştere - ţări. Să se afişeze oraşele şi ţările din care fac parte.
Persoane - oraşe de naştere - ţări - oraşe vizitate
Trenuri - staţii (ora sosire/plecare planificată, ora sosire/plecare reală)
Actori - filme

Biblioteca
Banca
Gestiune
Reţete

studenti - orase-de-nastere, orase - judete, orase - tari, facturi-clienti, produse - categorii, produse - firme-producatoare, jucatori - tari, jucatori - echipe, echipe - tari
studenti - examene, studenti - profesori, facturi - produse, actori - filme, carti - abonati
spectacole - artisti, opere - spectacole (reprezentatii), opere - roluri, opere - compozitori

' OR '1'='1

HTML

- Tag - folosit pentru a specifica regiuni ale documentului HTML pe care browser-ul le va interpreta ulterior.
- Element - este un tag complet, având un <tag> de deschidere şi unul de inchidere </tag>.
- Atribut - este folosit pentru a stabili valoarea unui element în HTML. Un element poate avea mai multe atribute.

head
body

title
bgcolor
background

p - align
div

b *sau* strong
i *sau* em
u

font face size color

sub sup del

h1, h2, h3, h4, h5, h6
br
hr - width color

ul
ol
li

img - src width height
alt align
style="width:100px;
height:50px;
margin:auto;"

a href title target
base

<!-- comentariu -->

table
tr td th
align width height
bgcolor background

border bordercolor
cellspacing=0
cellpadding=2

form
method name action

input type="text" -
name size maxlength
value **required**
readonly **placeholder**

input type="**number**"
min max step
style="width: 7em"

input type="**password**"
input type="**date**"
input type="**email**"
input type="url"

input type="checkbox"
input type="radio"
input type="**hidden**"
input type="**file**"
input type="range"

input type="**submit**"
input type="reset"

select - name size
multiple - option

textarea - rows cols

embed (audio)
src autostart loop
volume width height
hidden

https://www.youtube.com/watch?v=gaXE0v0bJoE

```
<object width="425" height="344">
<param name="movie" value="http://www.youtube.com/v/UAq8qHNWMNw&hl=en&fs=1"></param>
<param name="allowFullScreen" value="true"></param>
<embed src="http://www.youtube.com/v/UAq8qHNWMNw&hl=en&fs=1" type="application/x-shockwave-flash"
allowfullscreen="true" width="425" height="344">
</embed>
</object>
```

<input type='reset' name='btn_reset' value='Nu' onClick='**javascript:history.back**()' />