

**Lista temelor**, cu toate cerintele, pentru **a doua lucrare practica**.

***Cerinte comune tuturor temelor:***

- toate clase vor conține obligatoriu constructori de inițializare, parametrizați și de copiere, destructor, iar operatorii „=”, „>>”, „<<” să fie supraincarcati.
- ilustrarea conceptelor: upcasting, downcasting, funcții virtuale (pure – unde se considera mai natural)
- utilizarea variabilelor și funcțiilor statice
- citirea informațiilor complete a n obiecte, memorarea și afisarea acestora

***Cerinte specifice fiecărei teme:***

**Tema 1.** liste de numere reale, reprezentate ca tablouri uni- si bi-dimensionale:

Se dau următoarele clase:

- Vector (int dim, float \*a)
- Matrice (Vector \*v)
- Matrice\_oarecare (int lin) : Matrice
- Matrice\_patratice (int dim) : Matrice – nr de linii (dim) trebuie sa coincida cu dimensiunea oricarui vector component.

***Obs. Se considera ca vectorii care intra în componenta matricelor au același numar de elemente.***

Clasele derivate trebuie sa contina cel puțin constructori parametrizați (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructori si o metoda care sa verifice daca o matrice triunghiulara este diagonala. Pentru matricile patratice, functia de afisare sa conțină și determinantul acestora.

**Tema 2.** vectori de numere complexe

Se dau următoarele clase:

- Complex (float re,im)
- Vector (int dim, Complex \*v).
- Stiva : Vector - cu operațiile de adaugare și stergere modificate corespunzator.
- Coada: Vector - cu operațiile de adaugare și stergere modificate corespunzator.

Clasele derivate trebuie sa contina cel puțin constructori parametrizați (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructor si o metoda prin care sa verifice dacă stiva / coada conține elemente care au doar partea imaginara nenula.

### **Tema 3.** Cozi de caractere (implementate dinamic):

Se dau urmatoarele clase:

- Nod (char info, nod\*next)
- Coada (nod \* prim, nod \* ultim, int dim\_max);
- DEQUE : Coada - cu operatiile de adaugare și stergere modificate corespunzator.
- CoadaPrioritati (int \*prioritate) : Coada - cu operatiile de adaugare și stergere modificate corespunzator.

***Obs. Prioritate - vector alocat dinamic care conține prioritatea pentru fiecare element .***

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza) si destructor. Simulați operatiile pe o stiva, respectiv pe o coada, folosind coada cu prioritati.

### **Tema 4.** liste de numere întregi 1 (implementate dinamic)

Se dau urmatoarele clase:

- Nod (int info, nod\*next)
- Nod\_dublu (nod \* ante) :Nod
- LDI (elemente de tip Nod\_dublu); // lista dublu inlantuita;
- LSI : LDI ; // lista simplu inlantuita, obtinuta mostenind caracteristicile unei LDI adaptate la o inlantuire simpla;

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor către constructorul din clasa de baza) si destructor. Să se exemplifice sortarea prin insertie directa utilizand LDI.

### **Tema 5.** liste de numere întregi 2 (implementate dinamic)

Se dau urmatoarele clase:

- Nod (int info, nod\*next)
- Nod\_dublu (nod \* ante) :Nod
- Lista (elemente de tip Nod\_dublu)
- Lista\_Circulara : Lista;

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor către constructorul din clasa de baza) si destructor. Să se rezolve problema lui Josephus folosind liste circulare.

### **Tema 6.** liste de numere întregi 3 (implementate dinamic)

Se dau următoarele clase:

- Nod (int info);
- Nod\_simplu (nod \* next) :Nod
- Nod\_dublu (nod \* ante) :Nod\_simplu
- Nod\_prioritate (int prio) :Nod\_dublu
- Coada\_prioritati (elemente de tip Nod\_Prioritate);
- Coada\_simpla : Coada\_prioritati (elementele au prioritati egale).

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor către constructorul din clasa de baza) si destructori. Să se exemplifice Heapsort utilizand coada cu prioritati și Insertionsort utilizand coada simpla.

### **Tema 7.** multimi finite de numere întregi reprezentate ca tablouri unidimensionale

Se dau următoarele clase:

- Pereche(int prim,doi)
- Multime\_pereche ( int n, Pereche \* p)
- Stiva\_pereche : Multime\_pereche – cu operatiile de adaugare și stergere modificate corespunzător;
- Coada\_pereche : Multime\_pereche – cu operatiile de adaugare și stergere modificate corespunzător;

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza) si destructori. Să se implementeze o functie prin care se simuleaza operatiile pe o stiva de perechi folosind 2 cozi de perechi.

### **Tema 8.** polinoame reprezentate ca tablouri unidimensionale (prin gradul polinomului si vectorul coeficientilor.

Se dau următoarele clase:

- Clasa Monom(int grad, float coef)
- Clasa Polinom(int nr\_monoame, Monom \*m)
- Polinom\_ireductibil : Polinom;
- Polinom\_reductibil : Polinom.

Clasele derivate trebuie sa contina constructoi parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructor si o metoda care sa aplice criteriul lui Eisenstein de verificare a ireductibilitatii polinoamelor.

(Webografie ajutatoare: <http://www.profesoronline.ro/teorie-3140-1.html>)

Afisarea unui polinom reductibil să fie făcută și ca produs de 2 polinoame.

**Tema 9.** matrice de numere complexe reprezentate ca tablouri bidimensionale

Se dau urmatoarele clase:

- Complex(float re,im)
- Matrice( Complex \*\*v)
- Matrice\_oarecare (int lin, int col) : Matrice
- Matrice\_patratice (int dim): Matrice

Clasele derivate trebuie sa contina cel puțin constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructori si o metoda care sa verifice daca o matrice triunghiulara este diagonala. Pentru matricile patratice, functia de afisare sa conțină și determinantul acestora.

**Tema 10.** matrice de numere complexe reprezentate ca structuri inlantuite (ca matrici rare si nu neaparat patratice)

- Aceleași cerinte ca la Tema 9, fără calcularea determinantului.

**Tema 11.** arbori binari de cautare in reprezentare inlantuita:

Se dau urmatoarele clase:

- Nod (int info, nod\*st, nod \*dr)
- Nod\_rosu\_negru (char\* culoare) :Nod
- Arbore(int nr\_noduri)
- ABC (Nod \*rad):Arbore
- Arbore\_bicolor (Nod\_rosu\_negru \*rad) : Arbore

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructori si o functie virtuala prin care inaltimea arborelui bicolor sa fie identificata prin Adancimea neagra a radacinii.

Webografie: [https://writer.zoho.com/public/miha\\_cio/Arbori-bicolori1/noband](https://writer.zoho.com/public/miha_cio/Arbori-bicolori1/noband)

**Tema 12.** arbori binari de cautare AVL in reprezentare inlantuita:

Se dau urmatoarele clase:

- Nod (int info, nod\*st, nod \*dr)
- Nod\_AVL (int echilibru) :Nod
- Arbore(int nr\_noduri)
- ABC (Nod \*rad):Arbore
- Arbore\_AVL(Nod\_AVL \*rad) : Arbore

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructori si o functie virtuala prin care să se afiseze factorii de echilibru pentru fiecare nod.

**Tema 13.** arbori oarecare, in reprezentare inlantuita, prin legaturile fiu (catre fiul cel mai din stanga) si frate (catre urmatorul fiu al tatalui, in ordinea fiilor tatalui de la stanga la dreapta).

Se dau urmatoarele clase:

- Nod (int info)
- Nod\_ABC: Nod( Nod\*st, Nod\*dr)
- Nod\_fiu\_frate (int copii, Nod \*\* leg) :Nod
- Arbore(int nr\_noduri)
- ABC (Nod \*rad):Arbore
- AB\_oarecare(Nod\_fiu\_frate \*rad) : Arbore

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructori si o functie virtuala prin care afisarea elementelor unui arbore binar oarecare sa fie data de parcurgerea în latime, cu mentiunea listei fiilor pentru fiecare nod, iar afisarea unui ABC sa conțină cele 3 parcurgeri simultan.

#### **Tema 14.** grafuri

Se dau urmatoarele clase:

- Matrice (int \*\*a) - matrice de adiacenta
- Vector (int \*v, int dim)
- Lista (Vector \*l) - lista de adiacenta
- Graf (int nr\_noduri) - clasa abstracta
- Graf\_Neorientat(Lista L) : Graf
- Graf\_Orientat (Matrice A) :Graf

Clasele derivate trebuie sa contina constructor parametrizat (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza), destructor si o metoda care sa afiseze vectorul de tati, dacă pentru un Graf Orientat se verifica conexitatea lui și se da un nod de plecare pe post de rădăcina.

#### **Tema 15.** grafuri orientate,

Se dau urmatoarele clase:

- Matrice (int \*\*a) - matrice de adiacenta
- Graf (int nr\_noduri) - clasa abstracta
- Graf\_complet (int nr\_muchii) : Graf
- Graf\_antisimetric( Matrice M) : Graf
- Graf\_turneu: Graf\_complet, Graf\_antisimetric

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa/clasele de baza), destructori si o metoda virtuala care sa afiseze arcele unui Graf\_antisimetric, sau a unui Graf\_turneu.

**Obs: graf turneu - orientat complet si antisimetric. Un graf orientat se numește antisimetric dacă pentru oricare două vârfuri din graf  $x$  și  $y$  dacă există arcul  $(x,y)$ , atunci nu există arcul  $(y,x)$ . Un graf orientat sau neorientat se numește complet dacă oricare două vârfuri din graf sunt adiacente.**

**Tema 16:** Din clasa Nr\_Natural\_Mare să se deriveze clasa

Se dau următoarele clase:

- Vector (int \*a)
- Nr\_Natural\_Mare(int\_nrcif, Vector V)
- Numar\_Intreg\_mare(char semn) : Nr\_Natural\_Mare

Să se implementeze operațiile uzuale (+,-,\*). Clasele derivate trebuie să conțină constructori parametrizați (prin care să se evedentieze transmiterea parametrilor către constructorul din clasa de bază) și destructori.

**Tema 17.**

Se dau următoarele clase:

- Vector (int \*a)
- Nr\_Natural\_Mare(int\_nrcif, Vector V)
- Numar\_Fibonacci\_mare:Nr\_Natural\_Mare

Să se implementeze ierarhia de clase mai sus menționată, iar clasa derivată trebuie să conțină constructori parametrizați (prin care să se evedentieze transmiterea parametrilor către constructorul din clasa bază), destructor și o funcție virtuală care să afișeze numărul Fibonacci, precum și descompunerea lui în suma de numere Fibonacci.

**Tema 18.**

Departamentul „Salarizare” al firmei X actualizează fișele întregului personal utilizând o aplicație OOP. În acest scop se vor implementa clasele:

- Data (int zi, char\* luna, int an);
- Angajat (char \*nume, prenume ; float salariu, Data data\_angajarii);
- Part\_Time (int nr\_ore\_zi, Data final\_contract) : Angajat
- Permanent (int nr\_minori\_intretinere) : Angajat

De sărbători fiecare angajat va primi o primă. Angajatul permanent va primi un spor calculat ca un procent din primă fixă, egal cu vechimea lui în firmă, calculată până la 31 decembrie 2018 (de ex., dacă vechimea în firmă este de 12 ani, atunci va primi pentru fiecare copil minor 12% \* suma standard stabilită de firmă etc), iar angajații part time vor primi suma standard

stabilită, dacă contractul lor nu se termina la sfârșitul lunii curente, caz în care primesc doar 75% din suma standard prevazuta.

Realizați programul OOP astfel încât departamentul de salarizare sa aibă informații complete despre toți angajații.

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza) si destructori.

### **Tema 19.**

Se dau urmatoarele clase:

- Persoana(int id, char\* nume)
- Abonat(char \*nr\_telefon) : Persoana
- Abonat\_Skype(id\_skype) : Abonat
- Abonat\_Skype\_Romania (adresa\_mail) : Abonat\_Skype
- Abonat\_Skype\_Extern(tara) : Abonat\_Skype

Sa se construiasca clasa Agenda ce contina o lista de abonati si sa se suprincarce operatorul [ ](indexare)care returneaza abonatul cu numele precizat .

Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza) si destructori.

### **Tema 20**

Se dau urmatoarele clase:

- Punct (float x, float y)
- Patrat (Punct stanga\_jos, float latura)
- Dreptunghi (float latura2) : Patrat
- Romb (Punct colt\_opus) :Patrat
- Paralelogram : Dreptunghi, Romb
- Trapez (float baza2) : Paralelogram

Toate figurile au 2 laturi paralele cu axa Ox. Clasele derivate trebuie sa contina constructori parametrizati (prin care sa se evidentieze transmiterea parametrilor catre constructorul din clasa de baza) si destructori. Functiile membre conțin și metode de calcul pentru arie și perimetru.

O data membra **valid** are valoarea 1 dacă figura este specificata corect.

Constructorii verifica paralelismul laturilor.

Definiti și implementati ierarhia de clase.