**Lista temelor**, cu toate cerintele pentru **prima lucrare practica**.

***Cerinte comune tuturor temelor:***

* implementare in C++ folosind clase
* datele membre private
* constructori de initializare (cu si fara parametrii), constructor de copiere
* get, set pentru toate datele membre
* **ATENTIE:** functiile pe care le-am numit mai jos *metode* (fie ca sunt supraincarcari de operatori, fie altfel de functii), **pot fi implementate ca functii prieten** in loc de metode ale claselor respective, daca se considera ca aceasta alegere este mai naturala;
* supraincarcarea operatorilor << si >> pentru iesiri si intrari de obiecte, dupa indicatiile de mai jos, astfel incat sa fie permise (si ilustrate in program):
* sa existe o metoda publica prin care se realizeaza citirea informațiilor complete a n obiecte, memorarea și afisarea acestora.
* programul sa aiba un meniu interactiv

Necesar: programul sa nu contina erori de compilare si sa ruleze in CodeBlocks

***Cerinte specifice fiecarei teme:***

**Tema 0. Clasa ”Numar\_Complex” //se va implementa in timpul orelor de laborator**

- membrii privati pentru partea reala si partea imaginara (double);

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunea vectorului la zero,

iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- metode publice pentru setat/furnizat partea reala si partea imaginara;

- metoda publica de afisare (sub forma "a", "i\*a", "-i\*a", "a+i\*b", "a-i\*b");

- metoda publica pentru determinarea modulului unui numar complex;

- suma a doua numere complexe, implementata prin supraincarcarea op +;

- produsul a doua numere complexe, implementat prin supraincarcarea op \*;

- împărțirea a doua numere complexe, implementata prin supraincarcarea op /.

**Tema 1. Clasa ”Vector\_Complex”**

- clasa este prietena a clasei **Numar\_Complex** definita în **Tema 0**

- membri privati pentru vectorul propriuzis si numarul de elemente;

- constructor pentru initializarea cu un numar dat pe toate componentele (primeste ca parametru numarul respectiv si numarul componentelor);

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunea vectorului la zero, iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- supraincarcarea operatorilor << și >> sa utilizeze supraincarea acestora în cadrul clasei Numar\_Complex;

- metoda publica pentru determinarea vectorului modulelor, folosind metoda de determinare a modulului din clasa numar complex;

- metoda publica pentru sortarea crescatoare dupa module a vectorului;

- metoda publica pentru calculul sumei tuturor elementelor vectorului, care sa utilizeze operatorul + din clasa de numere complexe;

**Tema 2. Clasa ”Matrice\_Complexa”**

- clasa este prietena a clasei **Numar\_Complex** definita în **Tema 0**;

- membri privati pentru matricea propriuzisa, nr linii și nr coloane;

- constructor pentru initializarea cu un numar dat pe toate componentele (primeste ca parametru numarul respectiv, numarul de linii și numărul de coloane);

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunile matricei la zero, iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- supraincarcarea operatorilor << și >> sa utilizeze supraincarea acestora în cadrul clasei Numar\_Complex;

- metoda publica pentru calculul sumei a doua matrici, implementata prin supraincarcarea operatorului + si pe baza functiei de supraincarcare a lui + din clasa numar complex;

- metoda publica pentru calculul produsului a doua matrici, implementat prin supraincarcarea operatorului \* si pe baza functiilor de supraincarcare ale lui + si \* din clasa numar complex.

**Tema 3. Clasa ”Vector” (vector de numere întregi)**

- membri privati pentru vectorul propriuzis si numarul de elemente;

- constructor pentru initializarea cu un numar dat pe toate componentele

(primeste ca parametru numarul respectiv si numarul componentelor);

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunea vectorului la zero, iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- metoda-operator public de atribuire =;

- metoda publica pentru reactualizarea numarului de componente si initializarea componentelor cu un numar dat (primeste ca parametru numarul respectiv si numarul componentelor);

- metoda publica pentru calculul sumei tuturor elementelor vectorului;

- metoda publica pentru găsirea maximului și a pozitiei lui;

- metoda publica pentru sortarea crescătoare a vectorului;

**Tema 4. Clasa ”Stiva\_de\_caractere” (implementata dinamic)**

Se considera ***Class Nod{ char info; Nod\* next;}***

- constructori de inițializare și parametrizati pentru clasa Nod;

- destructor pentru clasa Nod;

Clasa Stiva\_de\_caractere are:

- membru privat, un "Nod\*" (varful stivei);

- un constructor care initializeaza varful stivei cu NULL;

- un destructor care dezaloca toate elementele stivei;

- metode publice de adaugare a unui element în stiva (***push***), de stergere a unui element (***pop***), de afisare a varfului stivei (***top***) și pentru a testa daca stiva e vida (***isempty***);

- metoda publica de fisarea stivei, concomitent cu golirea ei, realizata prin supraincarcarea operatorului <<;

- supraincarcarea operatorului >>, realizata prin push-uri succesive;

- metoda publica pentru inversarea unui sir de caractere folosind o stiva;

- metoda publica, realizata prin supraincarcarea operatorului -, care sa considere doua stive și sa elimine, concomitent, elementele din ambele stive adaugand caracterul ce are codul ASCII mai mare într-o noua stiva, ca în exemplul de mai jos:

Stiva\_de\_caractere S1,S2;

S1 = {E,X,A,M,E,N}; S2 = {P,O,O,L,A,B,O,R,A,T,O,R} S1 - S2 = {R,O,T,A,X,O}.

**Tema 5. Clasa ”Coada\_de\_caractere” (implementata dinamic)**

Se considera ***Class Nod{ char info; Nod\* next;}***

- constructori de inițializare și parametrizati pentru clasa Nod;

- destructor pentru clasa Nod;

Clasa Coada\_de\_caractere are:

- membri privati, "Nod\*, Nod\*" (primul și ultimul element al cozii);

- un constructor care initializeaza coada cu NULL;

- un destructor care dezaloca toate elementele cozii;

- metode publice de adaugare a unui element în stiva (***push***), de stergere a unui element (***pop***) și pentru a testa daca e vida (***isempty***);

- metoda publica de fisarea a cozii, concomitent cu golirea ei, realizata prin supraincarcarea operatorului <<;

- supraincarcarea operatorului >>, realizata prin push-uri succesive;

- metoda publica pentru concatenarea a doua cozi de caractere, obtinand o alta coada de caractere, implementata prin supraincarcarea operatorului +;

- metoda publica, realizata prin supraincarcarea operatorului -, care sa considere doua cozi și sa elimine, concomitent, elementele din ambele cozi adaugand caracterul ce are codul ASCII mai mare într-o noua coada, ca în exemplul de mai jos:

Coada\_de\_caractere C1,C2;

C1 = {E,X,A,M,E,N}; C2 = {P,O,O,L,A,B,O,R,A,T,O,R} C1 - C2 = {P,X,O,M,E,N}.

**Tema 6. Clasa ”Lista\_circulara” (implementata dinamic)**

Se considera ***Class Nod{ int info; Nod\* next;}***

- constructori de inițializare și parametrizati pentru clasa Nod;

- destructor pentru clasa Nod;

Clasa Lista\_circulara are:

- cel puțin un membru privat „Nod\*” reprezentând nodul considerat prim;

- metoda publica de adaugare a unui element pe o poziție;

- supraincarcare a operatorului >>, realizata prin utilizarea succesiva a metodei decarata anterior;

- metoda publica de stergere a unui element de pe o poziție;

- metoda publica pentru inversarea legaturilor listei;

- metoda publica care primește parametrul întreg k și realizatea eliminarea elementelor listei circulare din k în k pana la golirea acesteia (elementele vor fi afisate în ordinea în care sunt eliminate);

- supraincarcarea operatorului +, care sa efectueze concatenarea a doua liste circulare, rezultand într-o noua lista circulara (ca în exemplul de mai jos).

Lista\_circulara L1 = { 1 → 2 → 3 → 1} , L2 = {4 → 5 → 6 → 4}

L1 + L2 = { 1 → 2 → 3 → 4 → 5 → 6 → 1}

**Tema 7. Clasa ”Lista\_dublu\_inlantuita”**

Se considera ***Class Nod{ int info; Nod\* prev, next;}***

- constructori de inițializare și parametrizati pentru clasa Nod;

- destructor pentru clasa Nod;

Clasa Lista\_dublu\_inlantuita are:

- doi membri privati „Nod\*” reprezentând primul și ultimul element;

- metoda publica de adaugare a unui element pe o poziție;

- supraincarcare a operatorului >>, realizata prin utilizarea succesiva a metodei decarata anterior;

- supraincarcare a operatorului << pentru afisarea listei in ambele sensuri, in aceeasi funcție;

- metoda publica de stergere a unui element de pe o poziție;

- supraincarcarea operatorului +, care sa efectueze concatenarea a doua liste dublu inlantuite, rezultand într-o noua lista dublu inlantuita.

**Tema 8. Clasa ”Multime”** (multimi finite de numere intregi reprezentate ca tablouri unidimensionale)

- membri privati pentru vectorul propriuzis si numarul de elemente;

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunea vectorului la zero, iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- metoda publica pentru transformare a unui vector in multime, prin eliminarea duplicatelor din respectivul vector;

- reuniune a doua multimi, implementata prin supraincarcarea operatorului +;

- intersectie a doua multimi, implementata prin supraincarcarea operatorului \*;

- diferenta a doua multimi, implementata prin supraincarcarea operatorului -.

**Tema 9. Clasa ”Multime\_dinamic”** (multimi finite de numere intregi reprezentate ca liste inlantuite). Cerintele sunt aceleași ca la Tema 8, adaptate pentru liste alocate dinamic.

**Tema 10. Clasa ”Polinom”** - reprezentat ca tablou unidimensional (prin gradul polinomului si vectorul coeficientilor (coeficientii vor apartine la monoame de grade consecutive), de dimensiune egala cu gradul plus 1, de la cel al termenului liber la cel de grad maxim.

- fiecare coeficient va fi de tip double;

- membri privati pentru vectorul propriuzis si numarul de elemente;

- constructori pentru initializare si copiere;

- destructor (în cazul alocarii statice, se seteaza dimensiunea vectorului la zero, iar în cazul alocarii dinamice, se dezaloca zona de memorie utilizata);

- metoda publica pentru calculul valorii unui polinom intr-un punct;

- suma a doua polinoame, implementata prin supraincarcarea operatorului +;

- diferenta a doua polinoame, implementata prin supraincarcarea operatorului -;

- produsul a doua polinoame, implementat prin supraincarcarea operatorului \*;

**Tema 11. Clasa ”Polinom\_dinamic”** - reprezentat ca lista inlantuita (ca polinoame rare, prin lista perechilor (coeficient,exponent), ordonata crescator dupa exponent, si nu neaparat cu exponentii consecutivi.

Cerintele sunt aceleași ca la Tema 10, adaptate pentru liste alocate dinamic.

**Tema 12. Clasa „student” avand**

* membri privati pentru nume(string), anul nasterii (intreg), numar\_credite (intreg), medie\_generala (double)
* constructor de initializare, destructor
* metode publice pentru setare/furnizare informatii pentru fiecare atribut
* supraincarcare pe operator >> si << pentru citirea si afisarea studentilor

Definiti clasa „grupa” avand

* ca membri privati un vector la „student” declarat dinamic, un intreg reprezentand numarul studentilor si un double reprezentand media generala a intregii grupe
* constructor de initializare, destructor care dezaloca toti studentii pointati in componentele vectorului
* eliminare student din grupa
* adaugare student in grupa
* verificare daca exista un student dupa nume

Cele doua clase de mai sus pot avea si alti membri/metode, dar se va respecta principiul incapsularii datelor (orice setare/furnizare a informatiilor continute in membrii privati se va face doar prin intermediul unor metode).

**Tema 13.**

**D1. Definiti clasa "persoana", avand**:

- membri privati pentru nume (string), anul nasterii (intreg), sex (caracter);

- sase metode publice pentru setarea/furnizarea informatiilor din cei trei membri privati.

Definiti clasa "baza\_de\_date" avand

- ca membri privati un vector la "persoana" declarat dinamic si un intreg reprezentand numarul persoanelor;

- ca metode publice un constructor care initializeaza vectorul cu NULL pe fiecare componenta, un destructor care dezaloca toate "persoanele" pointate de componentele vectorului, o metoda pentru adaugat o "persoana", trei metode pentru eliminat persoanele avand un anumit nume, respectiv an al nasterii, respectiv sex (cele trei metode vor avea acelasi nume, prin supraincarcare), doua metode pentru afisat persoanele continute in baza de date in ordinea alfabetica a numelor, respectiv crescatoare a varstelor.

Cele doua clase de mai sus pot avea si alti membri/metode, dar se va respecta principiul incapsularii datelor (orice setare/furnizare a informatiilor continute in membrii privati se va face doar prin intermediul unor metode).

Scrieti un program care sa gestioneze o baza de date folosind clasele de mai sus. El va afisa un meniu (in mod text), care va oferi functii de adaugat o persoana, eliminat persoanele dupa nume, varsta, sex, afisat

persoanele in ordine alfabetica sau dupa varsta.

**Tema 14.**

**D5 Clasa "vector" (vector de double), avand:**

- membri privati pentru vectorul propriuzis si numarul de elemente;

- constructor pentru initializarea cu un numar dat pe toate componentele (primeste ca parametru numarul respectiv si numarul componentelor);

- constructori pentru initializare si copiere;

- metode publice pentru citire si afisare;

- metoda-operator public de atribuire =;

- metoda publica pentru reactualizarea numarului de componente si initializarea componentelor cu un numar dat (primeste ca parametru numarul respectiv si numarul componentelor);

- operatori friend: + (suma element cu element), - (diferenta element cu element), +=, -= (cu efectul cunoscut), == (testarea egalitatii), != (testarea neegalitatii),

- functie friend "length" care furnizeaza numarul de elemente.

Actiunile vor fi implementate astfel ca atunci cand nu este posibila operatia (de exemplu adunarea unor vectori de dimensiune diferita) sa se afiseze un mesaj de eroare.

Program care citeste mai multi vectori si calculeaza suma lor si maximul lor.

**Tema 15.**

**D6 Clasa "matrice" (matrice de double), avand:**

- membri privati pentru matricea propriuzisa, numarul de linii si numarul de coloane;

- constructor pentru initializarea cu un numar dat pe toate componentele (primeste ca parametru numarul respectiv si numarul de linii si de coloane);

- constructori pentru initializare si copiere;

- metode publice pentru citire si afisare;

- metoda-operator public de atribuire =;

- metoda publica pentru reactualizarea numarului de linii si coloane si initializarea componentelor cu un numar dat (primeste ca parametru numarul respectiv, numarul liniilor si al coloanelor);

- operatori friend: + (suma), - (diferenta), \* (produsul dintre doua matrici si dintre o matrice si un intreg), == (testarea egalitatii),

- functii friend "nrlinii", "nrcoloane", "nrelemente" care furnizeaza numarul liniilor, coloanelor, respectiv elementelor matricii;

Program pentru rezolvarea unei ecuatii matriciale de forma "A\*X+B=0" (A,B,X,0 sunt matrici).

**Tema 16.**

**D10 Clasa "individ", avand:**

- membrii privati: i (intreg) = pozitia (dintr-un tablou unidimensional de 30 de elemente declarat static in main);

tip (char) = „numele” speciei ('+' sau '0');

varsta (intreg) = varsta (de la 0 la o valoare maxima fixa aleasa de programator);

energie (double) = energia;

viu (unsigned char) = este 1 daca e viu si 0 daca e mort;

- constructor care primeste ca parametrii pozitia si tipul si initializeaza corespunzator membrii respectivi; varsta se initializeaza cu 0, energia cu o valoare fixa aleasa de programator, viu cu 1;

- metode private:

***hraneste*** = creste energia proprie cu o valoare random intre 1 si 10;

***inmulteste*** = adauga un fiu in stanga sau in dreapta individului daca pozitia este libera (fiul are acelasi tip, varsta este 0, iar energia este de doua ori mai mare fata de cea a parintelui)

***ataca*** = pentru fiecare individ x invecinat si de alt tip, daca energia proprie este mai mare decat energia lui x, din energia proprie se scade energia lui x iar x este omorat (i se aplica metoda "***moare***");

***imbatraneste*** = creste varsta cu 1; scade energie cu o valoare constanta aleasa de programator; daca a atins o varsta maxima aleasa de programator sau daca energia sa a devenit <=0, individul curent este omorat (i se aplica metoda "***moare***");

***moare*** = viu devine 0;

- metode publice:

***actualizare*** = aplica succesiv metodele: hraneste, inmulteste, ataca, imbatraneste;

***esteviu*** = returneaza 1 daca viu==1 si 0 altfel;

***gettip*** = returneaza tipul.

Se citesc maxim 30 de indivizi (supraincarcare pentru >> si <<).

Realizare meniu cu functia de: citire indivizi (se citeste si numarul de indivizi aici), actualizare, este viu, get tip.

**Tema 17. Clasa „carte”**

-membri privati: denumire carte, autor principal, al doilea autor, numar de pagini, pret, rating (goodreads).

-metode: constructori, destructor, supraincarcare pe >> si <<, constructor de copiere, supraincarcare pe =, supraincarcare pe operatorii specifici de comparare (compararea se face pe baza ratingului).

Sa se citeasca de la tastatura un vector declarat dinamic de carti, sa se poata compara doua carti, sa se poata modifica informatiile unei carti din meniu si sa se afiseze toate cartile.

**Tema 18. Clasa „film”**

La fel ca tema 17 doar ca avem membri privati: denumire film, gen, regizor, rating (imdb).

**Tema 19. Clasa „teatru”**

La fel ca tema 17 doar ca avem membri privati: denumire\_piesa, numar actori, actori, rating.

Membrul „actori” este un vector dinamic de obiecte din clasa actor ce contine ca membri privati: nume actor (string), varsta(int).

**Tema 20. Clasa „Magazin”**

Exista un anumit stoc din mai multe tipuri de bunuri (unele masurate la bucata altele la greutate, altele la volum). Fiecare produs are un pret pe unitatea de masura si un cost. Clientul soseste cu o lista de cumparaturi cu bunuri. Magazionerul ii comunica clientului pretul total iar acesta decide daca cumpara sau nu. Daca cumpara se realizeaza, se produce schimbul bunuri contra bani. La finalul zilei se realizeaza inchiderea zilei, afisand totaluri de bunuri vandute si banii obtinuti. Scrieti un program care sa permita aceste operatiuni.

Precizari:

* Tipurile de Produsele pot fi cel putin: Varza (la buc), Faina (la kg) – cal. I, II sau III, Bere la doza (buc) – cu Brand si tip (blonda/bruna), **Vin** varsat (rosu sec si alb sec) – la volum, **Vin** de soi - la sticla (buc) – cu soi (Cabernet Savignon (rosu sec), Merlot (rosu dulce), Savignon Blanc (alb dulce), Chardonnay (alb dulce) ), an culegere si tara de origine (Franta, Argentina, Chile, Australia); Cartofi (la Kg) – rosii sau albi si Jucarii (la buc) – unicat (cu nume text).
* Magazinul detine Articole de stoc.
* Clientul poate specifica vag produsele, omitand anumite trasaturi (eg. Vin, Vin rosu; Faina) sau detaliat (vin rosu sec din 1996 din Franta). Magazionerul propune cele mai profitabile variante avand in vedere precizarile clientului si disponibilul de stoc.
* In lista de cumparaturi a clientului se pot repeta tipuri de produs.

**Tema 21. Clasa „Restaurant”**

La un restaurant exista mai multe mese. La fiecare masa se pot aseza 1 sau mai multi clienti (maxim in functie de capacitatea mesei). Clientii sosesc in grupuri si se aseaza la o masa suficient de mare. Apoi comanda din meniul care contine mai multe feluri de mancare si bauturi fiecare avand un pret. Fiecare client comanda pentru el. Dupa ce mananca parasesc masa (eventual pe rand). Cand un client paraseste masa el doreste nota de plata pentru ce a consumat el. La final se elibereaza nota de plata pentru clientii ramasi. Scrieti un program care sa permita aceste operatiuni.

Precizari:

* Feluri de mancare pot fi cel putin: Ciorba – de burta, de legume (ambele cu sau fara smantana si eventual ardei) si de ciuperci; Fel principal – de pui (Frigarui sau Tocana) sau de vita (Chateaubriand – precizat cat de bine facut sa fie, stroganoff sau file – la 100g), Garnitura (doar la fel principal) – cartofi (prajiti sau piure) sau orez sarbesc si Desert – Inghetata la cupa (intre 3 si 5 cupe) sau tort krantz.
* Suplimentar exista bauturi: Vin varsat (rosu sau alb) – la 100ml, Sticle de vin (Bordeux din 1996 sau Castel Bolovanu 2004), Sticle de bere (cu/fara alcool) sau apa plata (la sticla).
* Interactiunea cu chelnerul se desfasoara sub forma unui dialog in care aceasta afla progresiv dorintele clientului (e.g. De baut va aduc ceva? Da. Vin, bere sau apa? Vin. La pahar sau la sticla? La pahar. Cati ml? 200ml. Multumesc; doriti ciorba? Da. De care? De legume. Cu sau fara smantana? Cu. Cu sau fara Ardei? Fara. Fel principal doriti? Nu. Desert doriti? Da. Inghetata sau tort kranz? Inghetata. Cate cupe? 4).
* Chelnerul intreaba pe rand fiecare persoana de la masa ce doreste.
* Chelnerul poate fi chemat de un client la o masa si poate sa i se ceara Nota individuala sau nota colectiva (a tuturor clientilor ramasi la masa).

**Tema 22. Clasa „Atelier auto”**

Exista mai multe categorii de vehicule (motocicleta, bicicleta, automobil) care pot sosi la un garaj pentru a descoperi ce probleme tehnice au si a obtine un deviz de reparatii. Fiecare vehicul poate avea probleme din diferite categorii: frane, directie, noxe, motor, caroserie. Nu toate vehiculele pot avea toate tipurile de probleme. Fiecare tip de problema are o solutie tehnica ce presupune un anumit consum de materiale (de diferite tipuri) si un anumit efort uman (in ore-om). Fiecare material are un cost la fel ca si ora-om de efort uman. Pentru fiecare vehicul ce soseste in garaj trebuie calculat devizul estimativ.

Precizari:

* Tipurile de probleme vor include cel putin: Frane – placute uzate (fata/spate la un auto/moto), discuri uzate (oricare din 4 la un auto, oricare din 2 la moto), lant tocit (bicicleta), lant lipsa (bicicleta), defectiune capitala; Motor - Nivel ulei scazut (la moto si auto), carburator murdar (la auto), motor topit (la moto si auto); Directie – ghidon stramb (bicicleta, moto), roata stramba (toate), defectiune capitala (toate); Noxe – Vehiculul arde ulei (moto si auto), Vehiculul este anterior anului 2000 (moto si auto); Caroserie (auto) – Caroserie corodata de rugina, Caroserie stramba (aripi stanga/dreapta, fata/spate, bara, capota) - foarte stramba sau asa-si-asa (pe fiecare componenta in parte).
* Solutiile presupun un consum de materiale. Materialele sunt toate la bucata. Placutele de frana de pe spate difera de cele de pe fata. Placutele de frana sunt aferente fiecarei roti. Nivel ulei scazut, motor topit si vehiculul arde ulei toate presupun schimb de ulei.
* Fiecare problema are o solutie (mai putin defectiunea capitala). Pot exista tipuri de materiale comune mai multor solutii. Toate solutii consuma suruburi (minim 5).
* Vehiculul suporta operatiuni de traumatizare (e.g. Uzeaza discul de frana de pe spate dreapta sau Incepe sa arzi ulei).
* Vehiculul odata conectat la tester (in atelier) se va putea autodiagnostica verificand fiecare componenta a sa si returnand lista de probleme.

**Tema 23. Clasa „agentie de turism”**

Exista turisti. Ei au anumite caracteristici (barbati/femei, grupa de varsta si activitatile preferate). Exista destinatii turistice (Amsterdam, Thassos, etc.). Ele permit o serie de activitati (plaja, mers pe munte, vizitat muzee, etc.) dintre care unele pot fi potrivite doar pentru anumite sexe sau grupe de varsta. Unele activitati necesita un autovehicul personal. O activitate dureaza un numar de zile. O activitate poate fi si de grup (pot lua mai multi parte la ea). Pentru a ajunge la o destinatie se foloseste fie transport cu avionul fie transport individual cu autovehiculul. Pentru un grup de turisti (cu caracterstici si preferinte variate) sa se aleaga o destinatie potrivita si sa se realizeze un plan de activitati (o lista) pe zile (maxim 14) astfel incat intreg grupul sa fie satisfacut.

Precizari:

* O activitate se desfasoara intr-un anumit loc la destinatie. Daca distanta de la cazare pana la acel loc este mai mare de 2 Km, este necesar un autovehicul.
* Unele destinatii permit inchirierea de autovehicule, altele nu.
* Orice activitate dureaza minim jumatate de zi.
* O destinatie poate avea mai multe plaje. O plaje poate fi acoperita cu nisip sau pietricele mici (mai ales in Grecia).
* Unele plaje sunt de nudisti. Unele plaje de nudisti nu accepta minori.
* Mersul pe munte se poate efectua in grup. La fel si plaja.
* Mersul pe munte necesita un munte. Muntele prezinta diferite trasee de dificultate usoara/medie/grea. Trasele de dificultate grea sunt interzise copiilor sub 16 ani sau adultilor peste 90. Unele trasee pe munte necesita mai multe zile. Dintre acestea unele presupun opriri la o manastire. Unele manastiri nu accepta vizitatori femei.
* Exista mai multe tipuri de muzee (de arta, de istorie, stiintifice). In anumite destinatii este interzisa vizitarea muzeelor stiintifice de catre femei.
* Acelasi tip de activitati poate fi comun mai multor destinatii.
* Un turist care exprima preferinte pentru un anumit tip de activitate poate omite anumite detalii, fiind indiferent la ele sau le poate preciza (e.g. plaja; plaja cu nisip; mers pe munte minim 2 zile traseu mediu).

**Tema 24. Clasa „Firma de training”**

O firma de Traning ofera mai multe Programe de Training (dezvoltand o anumita Competenta) pentru cursantii sai. Exista o multime fixa maximala de cursanti. Programele sunt alcatuite dintr-o serie de cursuri si eventuale alte programe de Training (incluse). Pentru a termina un program de training, cursantul trebuie sa promoveze toate cursurile ce fac parte din acel program (direct sau indirect). Un curs (e.g. „Programare OOP in C++”) este urmat in cadrul unui Program de un anumit cursant o singura data, chiar daca face parte din mai multe sub-Programe. Daca un cursant a picat un Program, el il poate reface daca a luat minim 3 la toate prezentarile de pana atunci. Cursurile sunt de mai multe feluri (Programare elementara (OOP/non-OOP, ce limbaj), Limba straina (ce limba), Matematica (algebra/geometrie/analiza), Resurse Umane, Financiar, Legislatie auto, Sofat auto practic (tipul vehicului), Comunicarea NLP). Tutorul va completa pentru fiecare cursant al fiecarei prezentari a unui Program o Fisa de Evaluare. Ea va contine o portiune generala (ce cursant, ce program, ce prezentare) apoi notele la fiecare Lucrare Practica a fiecarei prezentari de curs continuta. Cursurile au un numar diferit de lucrari practice. Unele cursuri (Programare, Matematica, Comunicare NLP) au evaluare scrisa/orala de final. Pentru fiecare curs rezulta o nota finala. Pentru unele cursuri la calculul mediei se elimina cea mai slaba nota, pentru altele atat cea mai slaba cat si cea mai buna (alegeti voi la care!). Unele cursuri pot acorda ponderi diferite diferitelor LP (ramase in calcul). Nota la cursul de NLP este nota de la examenul oral daca media celorlalte LP este peste 5 si 1 altfel. Nota unui program se calculeaza ca media notelor programelor continute *direct*. Scopul este ca pe baza fiselor individuale sa se obtina o situatie statistica generala: per cursant (ce competente are si la ce nivel) si per competenta (cat cursanti au o anumita competenta peste un anumit nivel).

Precizari:

* Finalizarea cu promovare a unui Program de Training genereaza o Competenta.
* In functie de nota integului Program, Competenta poate fi la nivel mic (5-6), mediu (7-8) si mare (9-10).
* Exista programe de Finantist: Matematica (algebra) si Financiar; Manager: Resurse Umane, Financiar, Limba(engleza); Programator: Matematica (toate trei), Programare, Limba(engleza); Manager Echipa Programatori: Manager, Programare si Comunicare NLP; Sofer: Legislatie auto, Sofat auto practic; Manager echipa de tiristi: Limba(turca), Manager si Legislatie auto.
* Functionalitatea programului este de urmatorul fel: Sosesc un grup de cursanti care doresc sa urmeze un anumit Program. De la tasatura se citesc corespunzator datele din fisa de evaluare, apoi se afiseaza nota finala si nivelul de competenta obinut pentru fiecare. Fiecare cursant opteaza daca doreste sa refaca cursurile pe care le-a promovat anterior sau nu (caz in care se preia nota).
* La cererea utilizatorului se afiseaza statistica cu numarul de cursanti (fosti) avand o anumita competenta la un anumit nivel.

**Tema 25. Clasa „Curierat rapid”**

O firma de curierat are mai multe vehicule in teritoriu (scutere, masini, dube). Fiecare are o capacitate (o masa maxima admisa si un volum total posibil). Unele dube sunt frigorifice si pot transporta bunuri reci. Fiecare vehicul se afla intr-un anumit punct (x,y). Firma (dispeceratul) primeste comenzi de livrare a unor colete (mai multe articole cu masa / volum diferite, unele dintre ele putand fi reci) de la un client (aflat la alte coordonate x,y) catre o destinatie (din nou x,y). Intreg coletul este preluat de un singur livrator. Soferii se deplaseaza cu viteze diferite (scutere – 20 Km/s, masini – 10 Km/s, dube – 5 Km/s) catre **Tema 19. Clasa „teatru”**

destinatie, in linie dreapta (scutere) sau manhattan (masini, dube). Odata livrat un colet, un vehiculul ramane la destinatie daca nu mai are alte colete de livrat. Coletele trebuie sa ajunga „la timp”. Pentru diferite colete la timp poate insemna: pana la o anumita secunda din zi, intr-un interval de timp maxim (de la preluarea comenzii/preluarea coletului), cat mai repede (fara limita de timp). Firma poate aplica 3 strategii de alocare a coletelor (e.g. vehiculul cel mai apropiat de client, vehiculul care – conform strategiei sale curente – poate livra cel mai repede coletul, vehiculul cel mai putin incarcat). Vehiculele se aleg dintre cele capabile sa transporte coletul (au spatiu) si ar reusi conform strategiei curente sa il livreze la timp. Daca nu exista nici un astfel de vehicul, clientul este refuzat. De asemenea, soferul vehiculului poate opta pentru 3 strategii de livrare/ridicare (coletul cel mai urgent primul, coletul cel mai apropiat de pozitia curenta, first-come-first-served). Sa se simuleze operatiunile firmei de curierat intr-o zi.

Precizari:

* Un vehicul nu isi schimba destinatia pe parcursul unui drum (doar in momentele in care ridica/livreaza un colet).
* Strategia soferului poate trata unitar (la fel) ambele tipuri de operatiuni (ridicare si livrare) sau poate sa faca distinctie intre ele (e.g. mai intai livrari apoi ridicari).
* Strategiile de alocare pot fi schimbate de manager pe parcursul unei zile. La fel si strategiile fiecarui sofer (tot de manager) – insa fara ca un colet sa ajunga sa nu mai poata fi livrat la timp.
* Performanta algoritmului de alocare nu este cruciala pentru aceasta cerinta la acest curs.

**Tema 26. Clasa „Harti”**

O firma de software a primit cerinta de a scrie o aplicatie care determina drumul minim intre doua orase alese dintr-o regiune, folosind sistemul de strazi existent. Diferite regiuni au diferete sisteme de strazi, unele putand forma un graf linie, altele un arbore, altele un dag, altele un graf general si altele un graf complet. Fiecare oras are coordonatele (x,y) si drumul dintre doua orase (daca exista strada intre ele) este distanta plana (euclidiana) intre ele. Firma de software ruleaza algoritmul pe propriile servere, tinand datele pentru o serie de regiuni, pentru care primeste cereri de la clienti (drum minim de la A la B regiunea R; introduce drum intre A si B in regiunea R; creaza o regiune). Algoritmul trebuie sa ruleze cat de cat rapid avand in vedere tipul de graf asociat.

Precizari:

* Drumul minim returnat consta doar in distanta totala de parcurs.
* Observatii: Intr-un graf complet drumul minim de la A la B este fix strada AB. Intr-un arbore exista un singur drum intre doua noduri (care se poate calcula tinand cont de stramosul comun). Intr-un graf linie costul drumului minim se poate calcula in O(1) usor.
* Strazile sunt bidirectionale, mai putin in dag.

**Tema 27. Clasa „Biblioteca”**

O biblioteca ofera suplimentar clientilor sai un serviciu de stocare si regasire a informatiilor. Astfel, fiecare client poate solicita ca bibilioteca sa memoreze informatii pe care el le-a furnizat si pe care le-a numit intr-un anumit fel. Informatiile de memorat pot fi de tip text (siruri de caractere), numeric (numere naturale), matematic (numere complexe) si de tip adresa (tara, judet, oras, strada, numar). De asemenea, informatiile pot avea un nume (e.g. “Adresa Mariei”, “Numarul meu preferat”). Odata introduse in bibilioteca, informatiile capata un ID. Biblioteca ofera urmatoarele servicii: Adaugare informatii, Stergere informatii dupa nume/ID, Regasire informatii (dupa Nume sau ID) si Cautare Informatii (dupa valoare – e.g. 27).

Precizari:

* Daca se incearca adaugarea de informatii cu un nume deja existent, se vor produce urmatoarele: Pentru tipul text se va alipi (apenda) noua valoare la informatia stocata; Pentru tipul numeric si matematic se vor insuma valorile; Pentru tipul adresa se va returna eroare.
* Performanta cautarii nu este cruciala pentru aceasta cerinta.

**Tema 28. Clasa „Teatru Modern”**

Teatru modern: Un teatru modern ofera mai multe spectacole pentru public. Spectacolele pot fi de diferite tipuri (e.g. opera, teatru, circ). Unele tipuri sunt adecvate doar pentru adulti. Fiecare spectacol are un gen (comedie, drama, etc.). Spectacolele de circ sunt alcatuite din mai multe numere (prestatii). Unele numere pot face uz de animale vii. Spectatorii au anumite trasaturi (nume, varsta) si anumite Pofte (e.g. “opera comedie”) si anumite exigente (e.g. nu doresc animale vii). Un spectator poate avea mai multe pofte la un moment dat, el fiind multumit cu satisfacerea oricareia dintre ele. Spectacolele au mai multe prezentari pe parcursul unei luni (momente cand ruleaza). Fiecare prezentare dispune de un numar fix de locuri. Lucratorul de la ghiseu al teatrului modern are sarcina de a sugerea fiecarui client individual care soseste lista de prezentari cu spectacole care l-ar multumi pe client. Cerinta presupune redactarea unui program in acest scop.

Precizari:

* Toate spectacolele au o durata si un nume.
* Un act de opera are un solist principal. Solistul principal are un nume si poate fi Tenor, Bariton sau Bass. Unele opere pot avea mai multe acte (2-7), cu durate diferite, specificate pentru fiecare act in parte. Intre acte se ia o pauza fixa specifica operei.
* Spectacolele de circ sunt toate comedii. Intre numerele unui spectacol de circ nu se ia pauza. Unele numere de circ pot fi adecvate doar pentru adulti caz in care intreg spectacolul este astfel adecvat. Numerele care presupun animale vii precizeaza ce animale sunt folosite (lei, ursi, pisici, pinguini).
* Teatrul are un singur act. Unele piese de teatru sunt interactive (presupun participarea publicului). In acestea, pe langa durata fixa exista si o durata variabila dependenta de numarul de spectatori (un minut pentru fiecare spectator adult si doua pentru fiecare copil sub 16 ani).
* O pofta a clientilor cuprinde una sau mai multe din urmatoarele cerinte: (i) spectacolul sa fie de un anumit gen (comedie, drama, actiune); (ii) sa fie de un anumit tip (teatru, opera, circ); (iii) sa dureze minim un anumit timp (cu/fara pauze); (iv) sa fie circ care sa includa un anumit gen de animale vii (e.g. pinguini); (v) sa fie spectacol de teatru interactiv; (vi) daca este opera sa inglobeze un anumit tip de voce (e.g. Tenor).
* Exigentele clientilor sunt unele dintre urmatoarele: (i) sa nu fie o anumita combinatie de (tip, gen) – e.g. opera dramatica; (ii) sa dureze (cu tot cu eventualele pauze) maxim un anumit timp; (iii) sa nu presupuna animale vii periculoase; (iv) sa fie maxim un numar (e.g. 20) de spectatori. Pot exista mai multe exigente de acelasi tip. Suplimentar, spectacolul trebuie sa fie potrivit pentru categoria de varsta a clientului.
* Pe langa operatia la ghiseu, programul va permite si introducerea spectacolelor si prezentarilor.

**Tema 29. Clasa „Criptografie”**

Criptografie: In Criptografie una din activitatile cel mai adesea efectuate presupune diverse operatii aritmetice (+,-,\*,/) in diferite Corpuri (nu Grupuri). Scopul programului aferent acestei cerinte este sa permiteti operatii cat mai generice peste Corpuri. Astfel, un Corp va putea fi definit de grupul aditiv si grupul multiplicativ. Un grup va putea fi definit de un tip elementar (R, Z, **Q**, Z/nZ, numere complexe) sau de explicitarea regulilor de compunere (pentru grupuri finite). Dandu-se o specificatie pentru un Corp de catre utilizator sa se: Determine daca este valida, Implementeze operatiile aritmetice in acel Corp. Suplimentar, dandu-se doua Corpuri, sa se determine daca sunt izomorfe.

Precizari:

* Grupul aditiv si cel multiplicativ au aceeasi multime.
* Pentru doua elemente dintr-un corp trebuie implementata si comparatia pentru egalitate (sunt sau nu egale).
* Programul trebuie sa proceseze comenzi (de la tastatura) odata ce a citit corpurile de intrare.

**Tema 30. Clasa „Hotel”**

Hotel: Un hotel are mai multe tipuri de incaperi (camere, apartamente, sali de conferinta, restaurant, etc.). Restaurantul are o capacitate maxima fixa, mai mica decat capacitatea totala de cazare. Unele sali de conferinta sunt echipate cu proiector. Unele sali sunt dotate cu mese individuale pentru participanti (4-8 persoane) iar altele sunt organizate tip scena. La Hotel sosesc diferite cereri pentru organizarea de evenimente in grup. Fiecare grup are un numar de participanti (ce trebuiesc cazati in camere si apartamente) si o perioada de sedere (1-10 zile). Unii dintre participanti vor dori sa ia micul dejun la restaurant, altii in camera (aleg la inceput). Unele grupuri vor dori sa rezerve si o sala de un anumit tip (scena/individuala) cu capacitate suficienta pe parcursul sederii lor. Cererile sosesc pe rand. Sa se determine pentru fiecare cerere cea mai apropiata perioada de inceputul de an (ziua 1) cand ea poate fi onorata si sa se tipareasca detaliile (ce camere au fost rezervate, ce sala, etc.). De asemenea pot sosi cereri de anulare a unei rezervari.

Precizari:

* Camerele obisnuite au capacitate maxima 2.
* Apartamentele au capacitatea 4.
* Nu se cazeaza participanti din grupuri diferite in aceeasi camera.
* Rezervarea include pentru fiecare participant (nume) camera in care este cazat (numarul). Participantii nu au preferinte la cuplajul in camere.
* In cazul anularii unei rezervari anuleaza rezervarea tuturor elementelor (camere, sala, etc.).
* Ar fi dragutz daca rezervarile ar fi anulate rapid – bonus!

**Tema 31. Clasa „Spalatorie”**

Spalatorie: Exista mai multe tipuri de haine (Pantaloni, Rochii, Camasi, Paltoane, Gegci, Costume) cu greutati diferite, printre altele. La o Spalatorie, aceste tipuri de haine sunt Supuse unui proces tehnologic de curatatorie realizat cu niste echipamente (Masini). El presupune o serie de etape: Spalare, Stoarcere, Uscare si Calcare. Hainele usoare (pantaloni, rochii, camasi) se pot spala intr-o masina obisnuita. Hainele grele (Paltoane, Gegci) necesita alt echipament special la Spalare. Paltoanele si Gegicile nu se Storc si nu se Calca. Unele haine necesita Spalare peste o anumita temperatura, altele dimpotriva (sub o anumita temperatura). Hainele au culori (deschise/inchise). Ele nu se amesteca la Spalare. Pentru a urma intregul proces, hainele trebuie trecute prin mai multe masini. Este recomandat ca Masinile sa lucreze cand sunt umplute peste jumatate din la capacitatea maxima. Scopul programului este implementarea fluxului tehnologic: (i) preluarea de haine de la clienti; (ii) plasarea lor in masini; (iii) pornirea masinilor cand sunt suficient de pline; (iv) preluarea hainelor de la o masina si mutarea catre urmatoarea (Spalat, Stors, Uscat, Calcat). Hainele sunt preluate in ordinea sosirii clientilor.

Precizari:

* Spalatoria poate prelua mai multe articole de la un client la un moment dat (e.g. 2 pantaloni, 3 gegci, etc.).
* Capacitatea masinilor de Spalat si Stors este determinata de greutate, a celor de Uscat de numarul de articole iar Calcatul se realizeaza manual (avand capacitate 1 articol).
* Costumele se spala separat de alte categorii de haine in masinile obisnuite dar se pot spala impreuna cu Gegci sau Paltoane.
* La Spalare se utilizeaza o cantitate de detergent determinata astfel: Pentru camasi si rochii – o valoare fixa unica tuturor articolelor; pentru Pantaloni – dublul cantitatii fixe dar doar daca sunt de culoare inchisa; pentru restul se folosesc 100g detergent / Kg de articol.
* Fiecare masina are o durata de functionare a programului (de spalat, stors, uscat) fixata.
* Durata calcatului este dependenta de greutatea obietului si de tipul sau. Pentru camasi este 120 secunde / Kg, pentru pantaloni 90 secunde / Kg, iar pentru costume suma dintre durata pentru sacou (150 s/Kg) si pentru pantaloni (idem mai sus).
* Clientii au rabdare oricat de multa.
* O haina „tine minte” prin ce a trecut, astfel incat, la finalul procesului tehnologic sa poata afisa operatiunile (detaliat) la care a fost supusa.
* Hainele se distribuie in prima Masina potrivita disponibila (fara un algoritm special).
* Cand toate hainele aferente comenzii unui client au fost procesate, este afisat pentru acea comanda totalul de detergent folosit si timpul total petrecut in Masini in timp ce lucrau (adica nu cat erau plasate acolo fara ca masina sa functioneze).