Universitatea “Constantin Brâncuşi” din Târgu-Jiu
Facultatea de Inginerie si Dezvoltare Durabila
Departamentul de Automatică, Energie, Mediu si Dezvoltare Durabila

Tematica la disciplina: Practica de specialitate
Anul II, specializarea Ingineria Sistemelor

Coordonator practică:
Lect.dr. Runceanu Adrian

Grup de lucru nr.1

1. Programare în limbajul C++

1.1. Pentru codificarea unui text format din litere mici se folosesc două tablouri bidimensionale 5x5, generate prin program.

a	*b*	*c*	*d*	*e*
f	*g*	*h*	*i*	*j*
k	*l*	*m*	*n*	*p*
o	*r*	*s*	*t*	*u*
v	*w*	*x*	*y*	*z*

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Codificarea se realizează caracter după caracter astfel:

- caracterul din linia i şi coloana j se codifică prin elementul corespunzător din tabloul de numere întregi
- după efectuarea acestei codificări, linia i din tabloul de numere se permută circular spre dreapta cu i poziţii, apoi coloana j din acelaşi tabel se permută circular în sus cu j poziţii

Să se scrie un program care, pentru orice text dat conţinând doar caractere mici ale alfabetului englez, să afişeze succesiunea de numere obţinută prin codificare. Textul poate să conţină şi spaţii care se vor ignora. Între două codificări succesive se va lăsa exact un spaţiu.
Exemplu: Codificarea şirului ‘exemplu’ este 5 23 4 2 12 11 20.

1.2. Se citeşte de la tastatură o matrice A cu m linii şi n coloane şi elemente numere întregi. Să se determine linia (liniile) din matrice care conţine cele mai multe elemente nenule. Se va folosi o funcţie care returnează numărul elementelor nenule de pe o linie a cărui indice i se transmite ca parametru.

1.3. Dându-se n numere întregi să se decidă dacă există un număr majoritar în această secvenţă. Un număr este majoritar dacă numărul său de apariţii în vector este mai mare decât n/2.
Exemplu : În vectorul x={2,4,3,2,2,2,6} numărul 2 este element majoritar.

1.4. Se citesc n numere naturale. Aceste numere se împart în grupe astfel încât în cadrul fiecarei grupe toate numerele au acelaşi număr de cifre 1 în reprezentarea în baza 2. Se cere să se afişeze mediile aritmetice a numerelor din fiecare grupă.
Exemplu: Pentru n=10 şi vectorul {92,60,47,16,52,45,65,7,8,87} se obţin următoarele medii aritmetice: 12, 65, 29.5, 65.55, 67.

Exemplu: Pentru matricea $\begin{pmatrix} 1 & 3 & 2 & 0 \\ 5 & 0 & 0 & 9 \\ 7 & 0 & 10 & 1 \end{pmatrix}$ liniile cerute sunt 1 şi 3.

1.5. Să se scrie un program care tipăreşte distribuţia frecvenţelor lungimii cuvintelor aflate într-un text citit de la tastatură. Cuvintele sunt separate prin spaţii.

Exemplu: Pentru textul ‘Toamna aceasta ploua foarte tare desi nu prea imi place acest lucru’ se va afişa:
1 cuvânt de lungime 2
1 cuvânt de lungime 3
3 cuvinte de lungime 4
4 cuvinte de lungime 5
2 cuvinte de lungime 6
1 cuvânt de lungime 7

1.6. Să se scrie un program care va realiza începerea fiecărei propoziţii dintr-un text cu literă mare. Spaţiile inutile din text se vor elimina. Dacă nu s-a facut nici o modificare se va da un mesaj corespunzător. Propoziţiile sunt despărţite între ele de un caracter punct urmat de zero, unul sau mai multe spaţii.

Exemplu: Textul ‘toamna a venit devreme. ploua si e frig, soarele nu mai arde asa de tare.’ se transforma in ‘Toamna a venit devreme. Ploua si e frig. Soarele nu mai arde asa de tare.’

2. Baze de date

Să se creeze tabelele ***Angajati_GRN***, ***Departamente_GRN*** (în şirul de caractere "**GRN**", ***GR*** reprezintă denumirea *grup*, iar ***N*** reprezintă numarul grupului de lucru la practica informatica, din care faceti parte). In cele doua tabele se vor introduce minim 20 linii in fiecare tabel.

DEPARTAMENTE_GRN
Id_dept number(3) cheie primara (PK)
Den_dept varchar2(20)
Id_manager varchar2(3)
Locatie varchar2(100)

ANGAJATI_GRN
Id_angajat number(3) cheie primara (PK)
Id_dept number(3) referinta (FK) la tabela **DEPARTAMENTE_GRN**
Nume varchar2(40)
Prenume varchar2(40)
Functie varchar2(25)
Salariu number(7)
Id_manager varchar2(3)
Data_ang date
Comision number(5)

2.1. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:

a) Sa se afiseze media aritmetica a salariilor angajatilor din departamentul cu numarul 80.
b) Sa se afiseze numele salariatilor al caror salariu este mai mare decat salariul lui Gheorghe Popescu.
c) Sa se afiseze numele salariatilor din departamentul Vanzari.
d) Sa se afiseze numele salariatilor impreuna cu numele managerului lor.

2.2. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:

a) Sa se afiseze numele angajatului care nu au manager.
b) Sa se afiseze numele si salariul angajatilor care-l au ca manager pe Gheorghe Popescu.
c) Sa se afiseze numele angajatilor impreuna cu denumirea departamentului din care fac parte, pentru salariatii al caror comision este nenul.
d) Sa se afiseze denumirile departamentelor la care sunt asignati angajati.

2.3. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:

a) Afişaţi numele şi prenumele tuturor angajaţilor care au salariul multiplu de 3.
b) Afişaţi numele şi prenumele tuturor angajaţilor din departamentul 30, mărind fiecăruia salariul cu 5,60% afişând rezultatul cu două zecimale.
c) Afisati numele, prenumele şi salariul tuturor angajaţilor unde locaţia departamentului este Tg-Jiu.

2.4. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:

a) Să se afişeze id-ul departamentelor care au media salariilor mai mare decât media salariilor din departamentul 60.

b) Să se introducă următoarea linie de valori în tabela ANGAJATI_PNU. (90, 13, 'Ionescu', 'Maria', 'IT programmer', 12000, 101, '05-29-2012', NULL) si apoi sa se afiseze informatiile din tabela.
c) Să se afişeze numele şi prenumele tuturor angajaţilor, id-ul şi numele departamentului acestora, chiar dacă nu le-a fost desemnat un departament.

2.5. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:

a) Să se afişeze numele, prenumele şi salariile mărite cu 25 % a tuturor angajaţilor din departamentele 12, 20 si 23 care au fost angajaţi după data de 07-06-2004 .
b) Să se mărească salariul tuturor angajaţilor din departamentul 20 al căror prenume conţine litera ‘e’ cu 55%.

3. Programare orientată pe obiecte

3.1. Se consideră un program care descrie organizarea personalului unei instituţii folosind claselor derivate. O clasă numită **Angajat** deţine date şi funcţii referitoare la un angajat al instituţiei:

```
class Angajat{
        char * nume;
        float salariu;
public:
        Angajat();
        Angajat(char *n, int s, float sal);
        Angajat(Angajat& r);
        void display();
        float getSalariu();
        void setSalariu(float s);
};
```

Diferite categorii de angajaţi necesită date suplimentare faţă de cele definite în clasa **Angajat**, corespunzătoare postului pe care îl deţin. De exemplu, un şef de secţie (administator) este un angajat (deci sunt necesare toate datele care descriu această calitate) dar mai sunt necesare şi alte informaţii, de exemplu precizare secţiei pe care o conduce.
Un administrator este un angajat, de aceea clasa **Administrator** se poate construi prin derivare din clasa **Angajat** astfel:

```
class Administrator : public Angajat
{
        int sectie;
public:
        Administrator(const char *n, float sal, int sec);
        Administrator(Administrator& r);
        ~Administrator();
};
```

Sa se implementeze aceasta ierarhie de clase.

3.2. Sa se implementeze tipul abstact **DataCalendaristica** care sa cuprinda:

- Constructor de initializare
- Functie statica pentru validare a unei date calendaristice
- Functie pentru citire si pentru afisarea unei date calendaristice cu un anumit format (zz/ll/aaaa sau ll/zz/aaaa sau aaaa/ll/zz)
- Functie membru statica care primeste un obiect **DataCalendaristica** si intoarce un obiect de acelasi tip dar care contine ziua urmatoare

3.3. Sa se implementeze tipul abstact **Triunghi**. Clasa va contine:

- Constructor de initializare
- O functie membru statica care verifica daca lungimile laturilor pot forma un triunghi.
- Functii pentru calculul ariei si perimetrului

- O functie care verifica daca triunghiul este sau nu dreptunghic.

3.4. Se considera o sala de cinematograf in care scaunele sunt asezate pe linii si coloane sub forma unei
matrici.Un loc in sala este identificat prin rand si numarul de ordine al scaunului in cadrul randului.
Sa se implementeze o clasa care sa retina diagrama locurilor pentru bilete vandute.
Indicatie: Clasa va avea metode pentru :

- Vanzare a unui billet (care va furniza un loc in sala).
- Functie pentru anularea unei rezervari
- Afisarea ocupari salii

3.5. Sa se implementeze tipul abstact **Polinom**. Clasa va contine:

- Constructor de copiere
- Destructor
- Metode pentru citire si afisare.

Sa se realizeze o functie friend care sa realizeze suma a doua polonoame.

4. Proiectarea algoritmilor

4.1. Fişierul text „adiacenta.in” conţine pe prima linie un număr natural n reprezentând numărul de noduri ale unui graf orientat, iar pe fiecare din următoarele n rânduri câte n valori de 0 şi 1 separate prin spaţii, reprezentând elementele unei linii a matricei de adiacenţă corespunzătoare grafului.
a) Să se scrie o funcție grad_intern ce primeşte ca parametru un număr natural x şi returnează gradul intern al nodului x.
Funcţia grad_extern primeşte ca parametru un număr natural x şi returnează gradul extern al nodului x.
b) Să se scrie programul C++ care citeşte datele din fişier, şi care afişează în fişirul text „noduri.out”, separate prin câte un spaţiu nodurile grafului care au gradul intern egal cu gradul extern, folosind apeluri utile ale subprogramelor grad_intern şi grad_extern.
Exemplu: Dacă fişierul ”adiacenta.in” are forma:
5
0 1 1 1 0
0 0 0 0 1
0 1 0 0 0
0 0 0 0 0
1 0 0 0 0
atunci fişierul „noduri.out” va conţine numerele 3 şi 5.

4.2. Se dă graful un graf neorientat pentru care datele se citesc din fişierul „graf.txt”. Să se verifice dacă un şir de m numere citite de la tastatură este lanţ în graf şi să se afişeze un mesaj adecvat. Dacă formează lanţ, să se verifice dacă lanţul este elementar.
Exemplu: Dacă fişierul „graf.txt” are următorul conţinut:
5
1 3

4.3. Intr-un grup de n persoane s-a stabilit o relatie de cunostinta: persoana x este in relatie cu persoana y daca o cunoaste pe aceasta. Relatia de cunostinta nu este reciproca. O celebritate este o persoana care este cunoscuta de toate persoanele din grup, dar care nu cunoaste nici o persoana (este un nod destinatie al grafului). Un necunoscut este o persoana care cunoaste toate persoanele din grup dar nu este cunoscuta de nici o persoana din grup (este un nod sursa al grafului). Un singuratic este o persoana care cunoaste o singura persoana din grup sau este cunoscuta de o singura persoana din grup (este un nod terminal al grafului). Un strain de grup este o persoana care nu cunoaste nici o persoana din grup si nu este cunoscuta de nici o persoana din grup (este un nod izolat al grafului). Demonstrati ca in grup nu poate exista decat o singura celebritate si un singur necunoscut. Scrieti un program care sa citeasca dintr-un fisier matricea de adiacenta a grafului si care sa afiseze:
a) Daca exista o celebritate, sa se precizeze persoana, iar daca nu exista, sa se precizeze: persoana care cunoaste cele mai putine persoane si persoana care este cunoscuta de cele mai multe persoane;
b) Daca exista un necunoscut, sa se precizeze persoana, iar daca nu exista, sa se precizeze: persoana care este cunoscuta de cele mai putine persoane si persoana care cunoaste cele mai multe persoane;
c) Daca exista singuratici, sa se precizeze persoanele;
d) Daca exista straini de grup, sa se precizeze persoanele;
e) Cate persoane cunosc doar doua persoane din grup si sunt cunoscute la randul lor de trei persoane din grup (nodurile care au gradul extern egal cu 2, iar gradul intern egal cu 3).
f) Cate persoane sunt cunoscute de un numar de membri egali cu numarul de membri pe care ii cunosc (nodurile care au gradul extern egal cu cel intern);
g) Care sunt persoanele care se cunosc reciproc (perechile de noduri intre care exista arce duble).

Universitatea “Constantin Brâncuşi” din Târgu-Jiu
Facultatea de Inginerie si Dezvoltare Durabila
Departamentul de Automatică, Energie, Mediu si Dezvoltare Durabila

Tematica la disciplina: Practica de specialitate
Anul II, specializarea Ingineria Sistemelor

Coordonator practică:
Lect.dr. Runceanu Adrian

Grup de lucru nr.2

1. Programare în limbajul C++

1.1. Se citesc n numere naturale. Aceste numere se împart în grupe astfel încât în cadrul fiecarei grupe toate numerele au acelaşi număr de cifre 1 în reprezentarea în baza 2. Se cere să se afişeze mediile aritmetice a numerelor din fiecare grupă.
Exemplu : Pentru n=10 şi vectorul {92,60,47,16,52,45,65,7,8,87} se obţin următoarele medii aritmetice : 12, 65, 29.5, 65.55, 67.

1.2. Simulaţi jocul de Bingo. Jucătorul introduce o variantă jucată, iar calculatorul generează aleator varianta câştigătoare şi afiăează rezultatul (linie, bingo, nimic). Varianta jucată şi cea câştigătoare se vor afişa pe ecran colorate, punându-se în evidenţă cu culori intermitente linia sau cartonul câştigător, dacă varianta jucată este câştigătoare.

1.3. Să se determine punctele şa dintr-o matrice A cu m linii şi n coloane folosind un subprogram care determină maximul dintr-un tablou liniar şi o funcţie care determină minimul dintr-un tablou liniar. (Observaţie : elementul aij se numeşte element şa a matricei A dacă el este minim pe linia i şi maxim pe coloana j sau invers).

1.4. Spunem despre un număr ca are proprietatea sufixului, dacă el apare ca sufix al pătratului său, adică cifrele sale apar la sfârşitul şirului de cifre ce reprezintă acel număr ridicat la pătrat. De exemplu 9376 are proprietatea sufixului în baza 10 (93762=87909376). Este clar că baza în care lucrăm este importantă, un număr poate avea proprietatea sufixului într-o bază şi poate să nu o aibă în altă bază. Scrieţi un program care citeşte de la tastatură un număr natural n (scris în baza 10) şi verifică dacă numărul are proprietatea sufixului în baza b<=10. Se va folosi o funcţie de transformare a unui număr din baza 10 în baza b, o funcţie care verifică dacă un număr are proprietatea sufixului într-o bază b.

1.5. Se consideră un fişier în care sunt înscrise următoarele informaţii referitoare la o grupă de studenţi: nume student, prenume student, media obţinută de student în sesiunea de examene curentă. Să se scrie funcţii care să realizeze următoarele operaţii:

- să afişeze media şi numele studentului cu media cea mai mare
- să determine media studentului de la mijlocul fişierului şi să semnalizeze situaţia în care fişierul conţine un număr impar de articole
- să ordoneze crescător articolele fişierului după valoarea mediilor şi să afişeze media şi numele studentului aflat pe ultima poziţie din fişier în urma acestei operaţii

1.6. Se citesc de la tastatură două şiruri de litere ale alfabetului englez. Să se tipărească caracterele care nu sunt comune celor două şiruri (adică apar doar într-un singur şir). Un caracter va fi afişat doar o dată. Literele mici nu se vor considera distincte de majuscule. Nu se vor folosi alte tipuri structurate de date.
Exemplu : Dacă cele două şiruri sunt ‘Elicopter’ si ‘TELEFON’ atunci se vor afişa literele ‘c’, ‘F’, ‘I’, ‘N’, ‘P’, ‘R’.

2. Baze de date

Să se creeze tabelele ***Angajati_GRN***, ***Departamente_GRN*** (în şirul de caractere "**GRN**", ***GR*** reprezintă denumirea *grup*, iar ***N*** reprezintă numarul grupului de lucru la practica informatica, din care faceti parte). In cele doua tabele se vor introduce minim 20 linii in fiecare tabel.

DEPARTAMENTE_GRN
Id_dept number(3) cheie primara (PK)
Den_dept varchar2(20)
Id_manager varchar2(3)
Locatie varchar2(100)

ANGAJATI_GRN
Id_angajat number(3) cheie primara (PK)
Id_dept number(3) referinta (FK) la tabela **DEPARTAMENTE_GRN**
Nume varchar2(40)
Prenume varchar2(40)
Functie varchar2(25)
Salariu number(7)
Id_manager varchar2(3)
Data_ang date
Comision number(5)

2.1. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Să se afişeze phone_number şi email pentru angajaţii care au comision şi al căror prenume începe cu o vocală literă mare. (se va folosi tabela ANGAJATI_GRN)
b) Să se afişeze numele şi prenumele celui mai recent angajat din departamentul 50. (se va folosi tabela ANGAJATI_GRN)
c) Să se afişeze numărul şi salariul mediu final (la care se adaugă comision), cu 3 zecimale, al angajaţilor din departamentul *Vanzari*. (se vor folosi tabelele ANGAJATI_GRN şi DEPARTAMENTE_GRN, iar coloanele se vor numi *numar angajati*, respectiv *salariu mediu angajati*)
d) Să se afişeze id-ul, numele şi prenumele angajaţilor, împreună cu numărul total de zile lucrate (coloana se va numi *numar zile*). (se va folosi tabela ANGAJATI_GRN)

2.2. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Să se afişeze numele şi salariul angajaţilor care sunt colegi de departament cu Popescu Vasile.
b) Creaţi un join care afişează toate departamentele din tabela DEPARTAMENTE_GRN chiar dacă au sau nu angajaţi înregistraţi în tabela ANGAJATI_ GRN.
c) Salariul din tabela ANGAJATI_ GRN este salariul lunar. Afişaţi numele, prenumele, şi salariul anual pentru fiecare angajat. Denumiţi coloana respectivă "Salariu Anual". Afişaţi în ordine alfabetică.
d) Să se afişeze toate slujbele(functiile) angajatilor şi data angajării lor, pentru acei angajati care au salariul cuprins între 3000 şi 5000.

2.3. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Să se afişeze media aritmetică a salariilor angajaţilor din departamentul cu numărul 20.
b) Să se afişeze numele salariaţilor al căror salariu este mai mare decât salariul lui Vasile Popa.
c) Să se afişeze numele salariaţilor din departamentul Vanzari
d) Să se afişeze numele salariaţilor împreună cu numele managerului lor.

2.5. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Să se afişeze numele angajaţilor care nu au manager.
b) Sa se afişeze numele şi salariul angajaţilor care-l au ca manager pe Vasile Popa.
c) Să se afişeze numele angajaţilor împreună cu denumirea departamentului din care fac parte, pentru salariaţii al căror comision este nenul.
d) Să se afişeze denumirile departamentelor la care sunt asignaţi angajaţii

3. Programare orientată pe obiecte

3.1. Sa se implementeze o ierarhie de clase pentru a realiza operatii cu multimi ale caror elemente vor fi de tipuri diferite.

```
class Element:
{
        public:
                virtual ~Element();
                virtual void display() = 0;
                virtual void read() = 0;
                int equals(Element&) = 0;
                virtual char *getClassName()=0;
};
class Set
{
        private:
                int size;
                Element *elements;
        public:
                Set(int size);
                Set(int size, Element *pe);
                Set(Set&);
                ~Set();
                void insert(Element*);
                void remove(Element*);
                int lookup(Element*); // Cautarea unui element
                Set operator +(Set &); // reuniunea
                Set operator –(Set &); // diferenta
                Set operator *(Set &); // intersectia
                int operator ==(Set &); // Testarea egalitatii
                int operator <(Set &); // Tesstarea incluziunii
                void read(); //Citirea elementelor multimii
                void display(); //Afisarea elementelor multimii
};
```

Sa se realizeze utilizand aceste clase intersectia a 2 multimi ale caror elemente sunt caractere.

3.2. Sa se implementeze clasa **Produs** (denumire, pret) si clasa **Magazin** care retine produsele dintr-un magazin. Clasa **Magazin** va contine metode pentru listarea ofertei de produse in care se vor afisa preturile in LEI si in USD.
Indicatie: Se va folosi o data membru statica care va retine paritatea leu/usd iar preturile se vor retine doar in dolari.

3.3. Sa se implementeze o clasa **Cursa** (numar_cursa, sursa, destinatie, ora_plecarii). Si o clasa care sa tina evidenta curselor dintr-un aeroport care va cuprinde metode pentru:

- Listarea tuturor curselor cuprinse intre anumite ore
- Ordonare a sborurilor dupa ora plecari si dupa destinatie

3.4. Sa se implementeze clasa **Pacient** (nume, prenume, varsta, diagnostic). Clasa va contine:

- Constructor de initializare
- Constructor de copiere
- Destructuctor
- Metoda pentru citire
- Metoda pentru afisare

Sa se construiasca un tablou cu obiecte de tip **Pacient** si sa se ordoneze crescator dupa varsta.

3.5. Sa se implementeze clasa echipa de fotbal care retine obiecte de tip **Jucator**. Sa se afiseze componenta echipei si sa se implemnteze o functie pentru a realiza substitutii.

4. Proiectarea algoritmilor

4.1. Se dă un graf neorientat cu n noduri. Numărul de noduri şi muchiile grafului se citesc din fişierul „graf.txt”. Să se verifice dacă graful este complet şi să se afişeze un mesaj adecvat. Dacă graful nu este complet să se afişeze muchiile care ar trebui adăugate astfel încât graful să devină complet.
Exemplu: Dacă fişierul „graf.txt” are următorul conţinut:
5
1 3
1 4
1 5
2 4
3 5
Se va afişa mesajul “Graful nu este complet” după care se vor afişa muchiile lipsă:
1 2
2 3
2 5
3 4
4 5

4.2. Fişierul text „arce.txt” conţine pe prima linie un număr natural n reprezentând numărul de noduri ale unui graf orientat, pe al doilea rând un număr natural m reprezentând numărul de arce ale unui graf orientat, iar pe fiecare din următoarele m rânduri câte două numere naturale separate prin spaţii, reprezentând arcele corespunzătoare unui graf orientat.
a) Să se scrie o funcție grad_intern ce primește ca parametru un număr natural x și returnează gradul intern al nodului x. Funcția grad_extern primește ca parametru un număr natural x și returnează gradul extern al nodului x.
b) Să se scrie programul C++ care citeşte datele din fişier, construieşte matricea de adiacenţă asociată grafului, şi care afişează în fişirul text „izolate.txt”, separate prin câte un spaţiu nodurile izolate ale grafului, sau mesajul „Nu există”, dacă în graf nu sunt noduri izolate, folosind apeluri utile ale subprogramelor grad_intern şi grad_extern.
Exemplu: Dacă fişierul ”arce.txt” are forma:
5
3
1 2
2 1
2 4
atunci fişierul „izolate.txt” va conţine numerele 3 şi 5.

4.3. Intr-o zona turistica exista **n** localitati. Intre unele localitati exista legaturi directe, fie prin sosele nationale, fie prin sosele judetene. Legaturile directe sunt caracterizate de lungimea drumului, masurata in kilometri. Se vor folosi doua grafuri: $G_n=(X,U_n)$ pentru legaturile prin sosele nationale si $G_j=(X, U_j)$ pentru legaturile prin sosele judetene. Cele doua grafuri se vor citi din doua fisiere text, care contin pentru fiecare legatura directa, pe cate un rand, separate prin spatiu, cele doua etichete ale nodurilor asociate localitatilor si distanta dintre localitati. Scrieti un program care sa citeasca aceste informatii din fisier si sa le transpuna intr-o implementare a grafului, adecvata problemei si care sa afiseze:

a) Localitatile la care nu ajung drumuri nationale (nodurile isolate, in primul graf)

b) Cea mai scurta legatura directa dintre doua localitati - se va preciza eticheta nodurilor asociate localitatilor, distanta dintre localitati si tipul soselei prin care se asigura legatura;

c) Pentru doua localitati precizate, **a** si **b** (etichetele nodurilor **a** si **b** se citesc de la tastatura), sa se precizeze daca exista legatura directa: daca exista, sa se mai precizeze tipul soselei si distanta dintre localitati;

d) Pentru o localitate **p** (eticheta **p** a nodului se citeste de la tastatura), sa se afiseze toate localitatile cu care are legaturi directe si distanta pana la aceste localitati (**Indicatie** – Se determina graful reuniune a celor doua grafuri).

e) Localitatea care are cele mai multe legaturi directe cu alte localitati – si sa se afiseze localitatile cu care are legaturi (nodul cu grad maxim in graful reuniune).