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.3

1. Programare în limbajul C++

1.1. Din fişierul text Vector.in se citesc: un număr natural n (n<2000) şi n perechi de numere a, b sunt două numere întregi ce reprezintă capetele unor intervale închise la ambele capete, de forma [a,b] (a<b).
Pe primul rând al fişierului text Vector.out, afişaţi suma obţinută prin adunarea tuturor numerelor întregi aflate în intervalul obţinut prin intersectarea celor n intervale date. Dacă intersecţia este mulţimea vidă, atunci se va afişa mesajul „Niciun element!”.
Exemplu: Dacă fişierul Vector.in are forma:
3
-3 2
-2 10
-2 7
atunci fişierul Vector.out va conţine:
0
Explicaţie: Intervalul obţinut prin intersectarea celor n intervale este [-2,2].

1.2. Din fișierul text date.in se citesc numere întregi. Să se determine cel mai mare divizor propriu al fiecărui număr citit. Afișarea va fi făcută în fișierul date.out.
Exemplu:
Date.in
6 8 15 21

Date.out
Divizorul maxim al elementului 6 este 3
Divizorul maxim al elementului 8 este 4
Divizorul maxim al elementului 15 este 5
Divizorul maxim al elementului 21 este 7

1.3. Să se citească din fişiereul text „Date.in” un şir de maximum 255 de caractere format din cuvinte separate prin unul sau mai multe spaţii. Cuvintele sunt formate numai din litere mici ale alfabetului englez. Scrieţi un program C/C++ care citeşte un astfel de şir şi afişează pe ecran frecvenţa de apariţie a fiecăreu litere din şir.
Exemplu:
Pentru şirul: competente profesionale
Se va afişa:
a apare de 1 ori
c apare de 1 ori
e apare de 5 ori
f apare de 1 ori
i apare de 1 ori
l apare de 1 ori
m apare de 1 ori
n apare de 2 ori
o apare de 3 ori
p apare de 2 ori
r apare de 1 ori
s apare de 1 ori
t apare de 2 ori

1.4. Se citesc numere naturale din fişierul text „Numere.txt”. Să se afişeze numerele care au proprietatea de palindrom (numărul citit de la dreapta la stânga este egal cu numărul citit de la sânga la dreapta) şi suma cifrelor să fie pară. Se vor folosi: un subprogram palin care va implementa un algoritm de determinare a proprietăţii de număr palindrom şi un subprogram care să calculeze suma cifrelor unui număr natural.
Exemplu: Pentru numerele 12, 12321, 565, 45, 18, 121 se va afişa 565, 121.

1.5. Se citesc numere naturale din fişierul ”Numere.txt”. Câte dintre aceste numere au toate cifrele cifre impare. Dacă nu există astfel de numere se va afişa un mesaj corespunzător. Se va folosi un subprogram care să verifice cerinţa din enunţ referitoare la paritatea cifrelor.
Exemplu: Pentru numerele 135, 57, 279, 791, 56 se va afişa 3, iar pentru numerele

1.6. Să se afişeze în ordine alfabetică cuvintele formate din două litere dintr-un text dat.
Exemplu:
Date de intrare: tu ai fost la mare
Date de ieşire ai la tu.

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 id-ul şi numele departamentelor al căror nume începe şi se termină cu aceeaşi literă; nu se face deosebire între litere mari şi mici. (se va folosi tabela DEPARTAMENTE_GRN)
b) Să se afişeze numele şi prenumele angajaţilor cu cel mai mare comision, acolo unde există. (se va folosi tabela ANGAJATI_GRN)
c) Să se afişeze numele şi id-ul departamentelor care nu conţin nici un angajat. (se vor folosi tabelele ANGAJATI_GRN şi DEPARTAMENTE_GRN)
d) Să se afişeze numele, prenumele şi salariul angajaţilor cu salariul cuprins între 10000 şi 15000, împreună cu numele departamentului din care fac parte. (se vor folosi tabelele ANGAJATI_GRN şi DEPARTAMENTE_GRN)

2.2. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Afişaţi numele şi prenumele tuturor angajaţilor din departamentul 80, mărind fiecăruia salariul cu 4,20% afişând rezultatul cu două zecimale.
b) Afisati numele, prenumele şi salariul tuturor angajaţilor unde locaţia departamentului este 1700.
c) Să se afişeze id-ul departamentelor care au media salariilor mai mare decât media salariilor din departamentul 60.
d) Să se afişeze numele şi prenumele tuturor angajaţilor, id-ul şi numele departamentului acestora, chiar dacă nu le-a fost desemnat un departament.

2.3. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Salariul din tabela ANGAJATI_GRN este salariul lunar. Afişaţi first name, last name, şi salariul anual pentru fiecare angajat. Denumiţi coloana respectivă "Salariu anual". Afişaţi în ordine alfabetică.
b) Afişaţi numele tuturor şefilor de departamente folosind tabelele ANGAJATI_GRN şi DEPARTAMENTE_GRN.

c) Creaţi un join care afişează toate departamentele din tabela DEPARTAMENTE_GRN chiar dacă au sau nu angajaţi înregistraţi în tabela ANGAJATI_GRN.
d) Din tabela ANGAJATI_GRN, afişaţi pentru toţi angajaţii care au ultima literă „s” în numele de familie: nume, prenume şi salariul indexat cu 10%, în ordine descrescătoare după salariu.

2.4. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Din tabela ANGAJATI_GRN afişaţi angajaţii care au salariul mai mare decât salariul mediu pe unitate.
b) Să se afişeze numărul de luni lucrate de angajaţii din tabela ANGAJATI_GRN.
c) Să se afişeze numele angajaţilor şi numele departamentul unde lucrează fiecare realizând un join pe tabelele ANGAJATI_GRN şi DEPARTAMENTE_GRN.
d) Afişaţi toţi angajaţii din tabela ANGAJATI_GRN care lucrează în acelaşi departament cu angajatul cu numele Mateescu.

3. Programare orientată pe obiecte

3.1. Sa se implementeze clasa urmatoare ce reprezinta polinoame si operatii cu acestea:

```
#define MAX 20

class Polinom
{
        float coef[MAX];
        int grad;
    public:
        Polinom(int m_grad, int* m_coef);
        Polinom(Polinom&);
        friend ostream& << (ostream& o, Polinom&
    p);
        Polinom operator + (Polinom& p);
        Polinom operator * (Polinom& p);
};
```

3.2. Sa se implementeze clasa Imprimanta care retine sub forma unei cozi obiecte de tip **Documen**t (nume, nr de pagini) care sunt trimise spre imprimare. Clasa va trebui sa aiba metode pentru:

- Adaugarea unui document in coada(la sfarsit)
- Listarea unui document (primul)
- Anularea imprimari unui document cu un anumit nume.
- Afisarea continutului cozii.

3.3. Sa se implementeze clasa **Matrice** care retine matrici rare (matrici cu foarte multe elmente 0). Se va utiliza o lista in care se retine: linia, coloana, elementul de pe acea pozitie. Clasa trebuie sa contina:

- Constructor de copiere
- Destructor
- Metoda pentru citire
- Metoda pentru afisare
- Modificarea unui element
- O functie friend care calculeaza suma a doua astfel de matrici.

3.4. Sa se realizeze o clasa **Multime** ale carei elemente sunt numere intregi. Clasa va contine:

- Metoda pentru citirea unei multimi
- Metoda pentru afisarea unei multimi
- Metoda statica care valideaza o multime (testeaza daca toate elemntele sunt distincte).
- Metoda friend care sa caluleze intersectia a doua multimi.

3.5. Sa se implementeze clasa **Planeta** (nume, raza, distanta fata de soare, durata perioade de rotatie in jurul soarelui). Sa se implementeze apoi clasa **SistemSolar** care are un nume si lista planetelor.

4. Proiectarea algoritmilor

4.1. 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
1 4
1 5
2 4
3 5
Şirul de numere 2 4 1 3, citit de la tastatură, formează lanţ elementar.
Şirul de numere 4 2 3 1 5 nu formează lanţ.

4.2. 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.

4.3. Reteaua de strazi dintr-un oras este formata din strazi cu doua sensuri si strazi cu sens unic de circulatie. Ea poate fi reprezentata printr-un graf orientat, in care intersectiile sunt nodurile, iar arcele – sensul de circulatie pe strazile care leaga doua intersectii (traficul auto). Nodurile sunt intersectii de cel putin 3 strazi. Pentru a stabili prioritatea in intersectii si pentru a fluidiza traficul, intersectiile vor fi modernizate. Pentru modernizare se vor folosi panouri cu semne de circulatie, semafoare – si se vor amenaja sensuri giratorii. Pentru fiecare strada din care se poate intra in intersectie, se monteaza in intersectie un panou cu semn pentru prioritate. Pentru fiecare strada pe care nu se poate intra din intersectie se monteaza in intersectie un panou cu semnul de interzicere a circulatiei. In toate intersectiile vor fi montate panouri cu semne de circulatie, corespunzator strazilor incidente cu intersectia. Fiecare dintre aceste mijloace de modernizare are un cost: panoul cu semn de circulatie – costul c1, semaforul – costul c2 si sensul giratoriu - costul c3. Pentru a stabili modul in care este

modernizata fiecare intersectie, intersectiile au fost clasificate in intersectii mici (intersectii cu 3 strazi, in care vor fi montate numai panouri cu semne de circulatie), intersectii mari (intersectii cu 4 strazi, care vor fi semaforizate) si intersectii foarte mari (intersectii cu peste 4 strazi, in care se vor amenaja sensuri giratorii). Desenati o retea ipotetica de strazi si construiti matricea de adiacenta a grafului orientat asociat. Scrieti matricea de adiacenta in fisierul text 'Strazi.txt'. Scrieti un program care sa citeasca matricea de adiacenta din fisier si care sa afiseze:
a) intersectiile la care nu se poate ajunge din nici o alta intersectie (nodurile care nu au predecesori);
b) intersectiile de la care nu se poate ajunge la nici o alta intersectie (nodurile care nu au succesori);
c) daca exista intersectii fara nici o intersectie successor sau fara nici o intersectie predecesor, sa se corecteze desenul retelei (prin adaugarea unui numar minim de arce), astfel incat sa nu existe asemenea intersectii – si sa se actualizeze fisierul strazi.txt.
d) intersectiile la care se poate ajunge direct din cele mai multe intersectii (nodurile care au cel mai mare grad intern);
e) intersectiile de la care se poate ajunge direct la cele mai multe intersectii (nodurile care au cel mai mare grad extern);
f) numarul de strazi pe care se circula in ambele sensuri (numarul de perechi de noduri i si j pentru care, in matricea de adiacenta, elementele a[i][j] si a[j][i] sunt egale cu 1);
g) numarul de intersectii mici, mari si foarte mari (clasificarea nodurilor in functie de numarul de noduri adiacente: noduri cu 3 noduri adiacente, cu 4 noduri adiacente si cu cel putin 5 noduri adiacente);
h) numarul de panouri cu semne pentru prioritate care se vor folosi (suma gradelor interne ale nodurilor);
i) numarul de panouri cu semne pentru interzicerea circulatiei care se vor folosi (diferenta dintre suma gradelor externe ale nodurilor si numarul de strazi cu sens dublu de circulatie);
j) numarul de semafoare care se vor monta (suma gradelor interne ale nodurilor care au 4 noduri adiacente);
k) costul de modernizare necesar pentru fiecare intersectie si costul total al modernizarii.

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.4

1. Programare în limbajul C++

1.1. Scrieţi un program C/C++ care citeşte din fişierul ”Date.in” un număr natural n (1≤n≤20), elementele unei matrice cu n linii şi n coloane, numere întregi din intervalul [-100,100] şi afişează pe ecran media aritmetică a elementelor prime ale matricei, care sunt situate deasupra diagonalei principale. Dacă nu există elemente prime situate deasupra diagonalei principale, programul va afişa mesajul „Nu exista!”.
Exemplu: pentru n=4 şi matricea de mai jos se afişează valoarea 3.33333
-1 2 -4 5
0 6 3 1
2 4 2 0
3 -5 1 -3

1.2. Scrieţi un program C/C++ care citeşte din fişierul ”Date.in” un număr natural n (1≤n≤50), elementele unei matrice cu n linii şi n coloane, numere naturale din intervalul [0,99999] şi afişează pe ecran numărul elementelor palindrom ale matricei, care sunt situate deasupra diagonalei secundare. Dacă nu există elemente palindrom situate deasupra diagonalei secundare, programul va afişa mesajul „Nu exista!”.
Un număr natural este palindrom dacă numărul citit de la stânga la dreapta este același cu numărul citit de la dreapta la stânga.
Exemplu: pentru n=4 şi matricea de mai jos se afişează valoarea 4.
112 121 444 56
102 696 123 17
262 413 221 10
393 585 111 33

1.3. Scrieţi un program C/C++ care citeşte din fişierul ”Date.in” un număr natural n (1≤n≤100), un număr natural m (1≤m≤10) şi elementele unei matrice cu n linii şi m coloane, cifre binare, fiecare linie a tabloului reprezintă cifrele unui număr în baza 2. Să se afişeze pe linii separate ale ecranuluinumerele care se obțin prin transformarea fiecărei linii din matrice în numărul corespunzător în baza 10.
Exemplu: Dacă fișierul conține
3 7
1 1 1 1 0 0 1
1 0 1 0 1 0 1

1 0 0 0 1 1 0
Atunci se va afișa:
121
85
70

1.4. Să se citească de la tastatură două şiruri formate din maximum 25 de caractere fiecare Fiecare şir conţine numai litere mici ale alfabetului englez. Scrieţi un program C/C++ care verifică şi afişează printr-un mesaj sugestiv, dacă cele două şiruri sunt sau nu anagrame (să conţină aceleaşi litere, în orice ordine de apariţie).
Exemplu:
Pentru şirurile „curat” si „urcat” se afişează “sirurile sunt anagrame”
Pentru şirurile „problema” si „emblema” se afişează “sirurile nu sunt anagrame”

1.5. Se dă un vector cu n componente numere naturale (datele se citesc de la tastatură, n<=50). Să se înlocuiască componentele neprime din vector cu numărul divizorilor acestora. Se vor utiliza subprogramele recursive: nrd care va implementa un algoritm de numărare a divizorilor unui număr natural şi inlocuire prin care se vor înlocui componentele neprime din vector.
Exemplu: Dacă vectorul citit este (5,6,1,0,15), după înlocuire devine (5,4,1,0,4).

1.6. Se citesc de la tastatură două numere întregi n și m. Construiți și afișați pe ecran, linie cu linie o matrice n*m construită după regula: a[i][j] = min(i,j). Numerotarea liniilor, respectiv coloanelor matricei începe de la 1.
Exemplu: Pentru n=2 și m=3 se va afișa matricea:
1 1 1
1 2 2

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 toţi angajaţii care lucrează în departamentul Marketing, folosind tabelele **ANGAJATI_GRN** şi **DEPARTAMENTE_GRN**.
b) creaţi un join care arată numele, id-ul departamentului şi numele departamentului pentru toţi angajaţii, chiar dacă nu au asociat un departament şi departamente care nu au angajaţi.
c) Raportaţi care angajaţi din tabela **ANGAJATI_GRN** au salariul mai mare decât media salariilor din întreaga unitate.
d) Afişaţi numele şi prenumele tuturor angajaţilor care au salariul multiplu de 3.

2.2. 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 50, 90 si 110 care au fost angajaţi după data de 07-06-2004.
b) Sa se selecteze pentru fiecare luna (specificata ca numar), cati oameni s-au angajat in acea luna de-a lungul timpului (nu conteaza anul angajarii). Raportul trebuie sa aiba doua coloane (Luna si Numar angajati, si sa prezinte cate o inregistrare pentru fiecare luna - functii de grup)
c) Sa se selecteze toti angajarii care fac parte din departamentul 'Productie', care s-au angajat inaintea angajatului din departamentul lor, care are cel mai mare salariu din departament
d) Sa se afiseze numele fiecarui angajat, insotit de numele departamentului si directiei din care face parte, data angajarii lor, ultima zi a lunii in care s-au angajat, prima zi de marti dupa data angajarii lor

2.3. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Afisati numele, meseria si data de inceput pentru cei care s-au angajat intre 20.02.1981 si 1.05.1981. Afisarea sa fie facuta in ordinea crescatoare a datei de angajare.
b) Afisati numele angajatilor si numerele de departament ale celor care muncesc in departamentele 10 si 30, ordonati alfabetic dupa nume.

c) Afisati numele angajatilor care au 2 de ‘L’ in numele lor si sunt din departamentul 30 sau au manager cu marca 7782.
d) Afisati numele, meseria si salariul pentru toti cei care sunt functionari sau analisti si salariul lor nu este egal cu 1000$ sau 3000$ sau 5000$.

2.4. Folosind baza de date cu tabelele create anterior sa se realizeze urmatoarele:
a) Scrieti o interogare care afiseaza numele, meseria, numarul departamentului, si numele departamentului pentru toti angajatii care lucreaza în DALLAS.
b) Scrieti o interogare pentru afisarea numarului de angajati cu aceeasi meserie.
c) Sa se creeze o cerere pentru a afisa numarul angajatului si numele sau pemtru toti angajatii care castiga mai mult decît salariul mediu. Sa se sorteze rezultatele în ordinea descrescatoare a salariului.
d) Scrieti o interogare care sa afiseze numele , numarul departamentului si salariul oricarui angajat al carui numar de departament si salariu sa se potriveasca cu numarul departamentului si salariul oricarui angajat care percepe comision.

3. Programare orientată pe obiecte

3.1. Sa se implementeze clasa urmatoare ce reprezinta tipul multime si operatii cu acesta:

```
class Multime
{
        int* elem;
        int number;
public:
        Multime(int m_number, int* m_elem);
        Multime(Multime&);
        ~Multime();
        Multime& operator= (Multime& m);
        Multime operator+ (Multime& m);
        Multime operator* (Multime& m);
        Multime operator- (Multime& m);
        Friend ostream& operator << (ostream& o, Multime& m);
};.
```

3.2. Sa se realizeze o clasa **Email** care retine o scrisoare electronica. Clasa va contine urmatoarele campuri (from = adresa de email a persoanei care expediaza scrisoarea, to = adresa de email careia ii este adresata scrisoarea, subject = scurta descriere a continutului, body = continutul scrisorii). Clasa va trebui sa valideze daca un sir de caractere este o adresa de email valida.
(O adresa de email este valida daca este de forma: nume@domeniu; unde nume poate fi orice combinatie de cifre si litere, caracterul underline (_) si punct si nu poate fi vid; domeniu este de forma subsubdomeniu.subdomeniu.domeniu (prin (sub(sub))domeniu se intelege orice combinatie de cifre si litere , _, -). Iar ultimul sir de caractere este de lungime 2 sau 3.

3.3. Sa se implementeze clasa **Numar** care retine un numar natural intreg in orice baza ca un vector al cifrelor. Sa se realizeze o functie friend care realizeaza suma a doua astfel de numere.

3.4. Sa se implementeze clasa **Meci** (nume echipa gazda, nume echipa oaspete, scorul). Sa se construiasca o clasa Etapa care retine meciurile sub forma unui tablou alocat in mod dinamic. Clasa trebuie sa cuprinda metode pentru:

- Listarea tuturor meciurilor
- Afisarea meciurior egale.
- Afisarea victoriilor in deplasare.

3.5. Sa se realizeze clasa **PartidPolitic** (nume partid, nume lider, orientare, numar membri, numar voturi). Sa se utilizeze a obiecte de acest tip pentru a afisa o statistica a unui sondaj de opinie cu privire la popularitatea partidelor la un moment dat.

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 regulat. În situaţia în care graful nu este regulat să se afişeze nodurile terminale şi numărul de noduri izolate. (Un graf neorientat se numește regulat dacă toate nodurile sale au acelaşi grad).
Exemplu: Dacă fişierul „Graf.txt” are următorul conţinut:
5
1 3
1 5
2 4
3 5
Se va afişa mesajul “Graful nu este regulat” după care se vor afişa nodurile terminale: 2, 4 şi numărul de noduri izolate: 0

4.2. Fişierul text „Graf.in” conţine pe prima linie un număr natural n reprezentând numărul de vârfuri ale unui graf neorientat, 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 ce primeşte ca parametru un număr natural x şi returnează gradul vârfului x.
b) Să se scrie programul C++ care citeşte datele din fişier şi care afişează în fişirul text „Graf.out”, pe primul rând, separate prin câte un spaţiu vârfurile terminale ale grafului, sau mesajul „Nu există”, dacă în graf nu sunt vârfuri terminale, folosind apeluri utile ale subprogramului grad.

Exemplu: Dacă fişierul ”Graf.in” are forma:
5
0 0 1 0 1
0 0 0 1 1
1 0 0 0 0
0 1 0 0 0
1 1 0 0 0
atunci fişierul „Graf.out” va conţine numerele 3 şi 4.

4.3. Intr-un munte exista n grote. Fiecare grota i se gaseste la inaltimea h_i fata de baza muntelui. Intre unele grote exista comunicare directa, prin intermediul unor tuneluri. Exista si grote care comunica cu exteriorul. Desenati o retea ipotetica de grote si construiti matricea de adiacenta a grafului neorientat asociat (grotele sunt nodurile, iar tunelurile sunt muchiile). Inaltimile grotelor se vor memora intr-un vector. Gasiti o modalitate de a evidentia grotele care comunica cu exteriorul. (Indicatie – adaugati la graf nodul 0, care reprezinta exteriorul muntelui.) Scrieti matricea de adiacenta si vectorul cu inaltimile grotelor in fisierul text ‘Grote.txt’. Scrieti un program care sa citeasca matricea de adiacenta si vectorul din fisier si care sa afiseze:
a) numarul de tuneluri de comunicare (numarul de muchii ale grafului);
b) grotele care au legatura cu exteriorul (nodurile i pentru care exista muchie cu nodul 0);
c) grotele care comunica direct cu cele mai multe grote (nodurile cu cel mai mare grad);
d) grotele care nu comunica prin tuneluri cu alte grote (nodurile izolate);
e) cea mai inalta grota si cea mai joasa grota care comunica cu exteriorul;
f) pentru o grota a carei eticheta se citeste de la tastatura, sa se afiseze grotele cu care comunica direct, precizand pentru fiecare tunel daca suie, coboara sau este la acelasi nivel cu grota.