Sesiunea 13

Agenda

- Rezolvare varianta Bac August 2025
- Rezolvare exercitii recapitulative functii prelucrare siruri
- Exercitii propus

Rezolvare varianta Bac August 2025

- Link: https://www.pbinfo.ro/resurse/9dc152/examene/2025/E_D_Informatica_2025_sp_MI_C_var_04_LRO.pdf

Subiectul I

1.
   - Dupa aplicarea operatorului `!` expresia devine : `x > 2020 && x <= 2025` deci (2020,2025]
   - a => Adevarat
   - b => Fals! Intervalul e inchis la 2025
   - c => Fals! Intervalul e deschis la 2020
   - d => Fals! Intervalul e inchis la 2025
   - Raspuns corect: `A`

2. Rezolvare:

```
    strcpy(s,"calculator"); // s = "calculator"
    k=strchr(s,s[1])-strchr(s,s[3]); // acum sa spargem pe bucati aceasta
expresie din interior catre exterior:
    s[3] = 'c'
    strchr(s,s[3]) = "calculator"
    s[1] = 'a'
    strchr(s,s[1]) = "alculator"
    k = 1 => Practic avem calculator - alculator si se scad pozitiile
pointerilor.
```

   - Raspuns corect: `D`

3. Rezolvare:

```
    Avem urmatoarele perechi:
    {A, B, C, D, E, F}

    Stiind ca perechea A este prima si perechea F este ultima practic avem:

    A _ _ _ _ F

    Primele 3 solutii sunt:
    (A, B, C, D, E, F),
```

```
    (A, B, C, E, D, F),
    (A, B, D, C, E, F)

    Ultima solutie este:

    A E D C B F
    Deoarece vedem ca solutiile generate sunt in ordine alfabetica deci
intr-un final ajungem la solutia de mai sus.
```

- Raspuns corect: C

4. Rezolvare:

- Din graful de mai jos observam ca pentru urmatoarele noduri avem acelasi grad interior/exterior:
   Nod 6 radacina - 3 - grad interior 1 / grad exterior 1 - 6 - grad interior 1 / grad exterior 1
  - Raspuns corect: B

5. Rezolvare

- Deoarece avem 50 de noduri, excludem varianta de a desena acest graf. Este prea mult si ne-ar incurca.
- Trebuie sa gasim rezolvarea in alt mod.
- Lant elementar: Lanțul care conține numai vârfuri distincte, două câte două, este lanț elementar.
- Si conform enuntului unde parintele unui nod i este nodul [i/2] deci pentru lantul elementar care incepe in 32 si se termina in 23 o aflam parcurgand urmatorii pasi:
- Parinti pentru 32 => 16 - 8 - 4 - 2 - 1
- Parinti pentru 23 => 11 - 5 - 2 - 1
- Deci lantul elementare ar merge => 32 - 16 - 8 - 4 - 2 - 5 - 11 - 23
- Cel mai de jos stramos comun este 2 apare in ambele liste.
- Distanta (numarul de muchii intre doua noduri) => 7
- Raspuns corect: B

Subiectul II

1. Rezolvare
   - a

```
x = 9767, y = 8204
cx = 0, cy = 0;
cat timp x >= 10 sau y >= 10 executa
    cat timp x+y != 0 executa
        cx = 0 + 7 = 7; x = 976
        cy = 0 + 4 = 4; y = 820
```

```
        cat timp x+y != 0 executa
            cx = 7 + 6 = 13; x = 97
            cy = 4 + 0 = 4; y = 82
        cat timp x+y != 0 executa
            cx = 13 + 7 = 20; x = 9
            cy = 4 + 2 = 6; y = 8
        cat timp x+y != 0 executa
            cx = 20 + 9 = 29; x = 0
            cy = 6 + 8 = 14; y = 0
        cat timp x+y != 0 fals

    cat timp x >= 10 sau y >= 10 executa
        cat timp x+y != 0 executa
            cx = 0 + 0 = 0; x = 2;
            cy = 0 + 4 = 4; y = 1
        cat timp x+y != 0 executa
            cx = 0 + 2 = 2; x = 0;
            cy = 4 + 1 = 5; y = 0
        cat timp x+y != 0 fals

    x = 2; cx = 0; y = 5; cy = 0;

    daca x = y -> false -> scrie "NU 2 5"
```

- Răspuns corect: NU 2 5

- b
  - Daca avem x = 2025 si dorim sa fie afisat "DA" trebuie pentru y sa alegem numere care au suma cifrelor egala cu suma cifrelor numarului 2025 adica 9 (sau 18 deoarece facem suma sumelor cifrelor). si trebuie sa alegem din intervalul [10,99]
    - 72 si 18
- c

```
#include <iostream>
using namespace std;

int main() {

int x, y;
cin >> x >> y;
int cx = 0, cy = 0;
while (x >= 10 || y >= 10) {
    while (x+y != 0) {
        cx = cx + x % 10;
        x = x / 10;
        cy = cy + y % 10;
        y = y / 10;
    }

    x = cx;
    cx = 0;
    y = cy;
```

```
            cy = 0;
        }

        if (x == y) {
            cout << "DA " << x;
        } else {
            cout << "NU " << x << " " << y;
        }
        return 0;
        }
```

- d

```
        citește x,y
        (numere naturale distincte)
        cx<-0; cy<-0

        ┌daca x≥10 sau y≥10  atunci
        │┌execută
        ││┌cât timp x+y≠0 execută
        ||| cx<-cx+x%10; x<-[x/10]
        ||| cy<-cy+y%10; y<-[y/10]
        ||└■
        || x<-cx; cx<-0; y<-cy; cy<-0
        │└■ cât timp x≥10 sau y≥10
        └■
       ┌dacă x=y atunci scrie "DA ",x
       │altfel scrie "NU ",x," ",y
       └■
```

2. Rezolvare:

```
    f(3,9) => il putem calcula direct deoarece 3*5 > 9/5 deci intram in
primul if si afisam direct 3
    f(1,1000) =
        X = 1;Y=1000 => 1*5 > 1000/5 => FALS => calculam f(5, 200)
        f(5,200) =
            x = 5;y = 200 => 5 * 5 > 200/5 FALS => calculam f(25, 40)
            f(25,40) =
                x = 25; y = 40 => 25 * 5 >40 /5 => returnam 25;
```

3. Rezolvare

```
    #include <iostream>
    #include <cstring>
    using namespace std;
```

```
    // Nota: doar ce este intre linii trebuie sa scrii pe foaia de la bac
    //-----------------------------------
    struct bijuterie {
        int greutate;
        struct {char denumire[31]; int pret; } metal;
    }b;
    //-----------------------------------
    int main() {

        // Nota: partea de mai jos nu trebuie sa o scrii la bac. Eu o fac
doar ca sa testez ca totul merge bine.
        strcpy(b.metal.denumire, "aur");
        b.metal.pret = 100;
        b.greutate = 121;

        cout << b.greutate * b.metal.pret<<endl;

        return 0;
    }
```

Subiectul III

1. Rezolvare:

   - Practic problema ne cere sa afisam divizorii pari care sunt stricti mai mari decat numar / divizor.
   - Mesajul formulat va fi de genul cout << divizorParCareRespectaCerinta << " parcele de arie " << numar / divizorParCareRespectaCerinta

```
    #include <iostream>

    using namespace std;

    void teren(int x, int y);

    int main() {

        teren(6,8);

        return 0;
    }

    void teren(int x, int y) {
        int arie = x * y;
        int exista = 0;

        for (int i = 2; i <= arie; i++) {
            if (arie % i == 0) {
                int arieLot = arie / i;
                if (i % 2 == 0 &&  i< arieLot) {
                    exista = 1;
```

```
                    cout << i << " parcele de arie " << arieLot << endl;
                }
            }

        }

        if (exista == 0) {
            cout << " nu exista";
        }
    }
```

2. Rezolvare:

   - Practic problema presupune urmatorii pasi:
     - Identificarea valorii minime
     - Salvarea valorii pe care vrem sa o folosim cand vrem sa inlocuim
       - Nota: trebuie sa faci asta inainte de a incepe inlocuirea deoarece este posibil ca si in ultima coloana sa inlocuim.
     - Iteram prin matrice si cand gasim un element egal cu valoarea minima, inlocuim toate elementele de pe coloana sa cu elementul dorit.

```
#include <iostream>

using namespace std;

int main() {

    int m, n;
    cin >> m >> n;
    int matrice[m][n];

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cin >> matrice[i][j];
        }
    }

    int minim = 101;
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (matrice[i][j] < minim) {
                minim = matrice[i][j];
            }
        }
    }

    int elementPentruInlocuire = matrice[m-1][n-1];
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (matrice[i][j] == minim) {
```

```
                    for (int k = 0; k < m; k++) {
                        matrice[k][j] = elementPentruInlocuire;
                    }
                }
            }
        }

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                cout << matrice[i][j] << " ";
            }
            cout << endl;
        }


        return 0;
    }
```

3. Rezolvare:

   - a

```
    O sa implementez un algoritm care mai intai va citi cele doua
variabile minZ si minP dupa care va citi numar cu numar din fisier. Cu
ajutorul unor variabile pe care le voi seta egale cu -1, voi salva
pozitia de inceput a unei perioade si pozitia de sfarsit. Dupa aceea
aplicam logica urmatoare:
    - Atunci cand intalnesc un stoc zilnic care este mai mare sau egal
cu minZ ma voi uita daca trebuie sa marchez inceputul unei perioade
(ziInceputPerioada == -1) caz in care initializez variabila
ziInceputPerioada cu valoarea pozitiei stocului (pentru care avem un
contor declarat), iar daca nu suntem la inceput de perioada, actualizam
variabila ce tine pozitia de sfarsit a perioadei.
    - De asemenea pentru fiecare stoc zilnic valid, actualizam si suma
pentru stocurile zilnice.
    - De fiecare data cand intalnim un stoc ce nu respecta conditia (>=
minZ) ne uitam daca am intalnit deja o perioada (ziInceputPerioada si
ziSfarsitPerioada trebuie sa fie diferite de -1 dar noi verificam doar
variabila pentru sfarsit deoarece ea se actualizeaza dupa variabila de
inceput) si ne mai uitam daca avem cel putin 2 zile in perioada (adica
diferente dintre cele 2 pozitii sa fie mai mare decat 1) dar si daca
suma stocului pentru perioada indeplineste conditia >= minP, si daca
expreesia este adevarata afisez conform cerintei.
    - Indiferent daca expresia este evaluata sau nu cu true, resetez
variabilele ce tin pozitia de inceput si sfarsit a perioadei, respectiv
suma stocului zilnic.

La final avem de facut 2 chestii:
    - 1 mai verificam inca o data daca avem sau nu o perioada valida
pentru a acoperi cazul in care fisierul se sfarsteste cu o perioada
valida
```

```
    - 2 de fiecare data cand afisam, setam un steag pe adevarata ca sa
stim daca am intalnit sau nu cel putin o perioada valida. Daca nu am
intalnit nici o perioada valida, afisam mesajul "nu exista"
```

- b

```
    #include <iostream>
    #include <fstream>

    using namespace std;

    int main(){

        ifstream fin("bac.in");
        int minZ, minP;
        fin >> minZ >> minP;

        int contorZi = 0;
        int ziInceputPerioada = -1;
        int ziSfarsitPerioada = -1;
        int sumaStocPerioada = 0;
        int stocZilnic;
        int existaPerioada = 0;
        while (fin >> stocZilnic) {
            contorZi++;
            if (stocZilnic>=minZ) {
                if (ziInceputPerioada == -1) {
                    ziInceputPerioada = contorZi;
                } else {
                    ziSfarsitPerioada = contorZi;
                }
                sumaStocPerioada += stocZilnic;
            } else {
                if (sumaStocPerioada >= minP && ziSfarsitPerioada != -1
&& (ziSfarsitPerioada - ziInceputPerioada > 1)) {
                    existaPerioada  = 1;
                    cout << ziInceputPerioada << " " <<
ziSfarsitPerioada <<  " " << sumaStocPerioada << endl;
                }
                ziInceputPerioada = -1;
                ziSfarsitPerioada = -1;
                sumaStocPerioada = 0;
            }
        }

        // Mai facem o verificare pentru a fi siguri ca prindem si
cazul cand la final, ultimele numere
        // reprezinta o perioada valabila
        if (sumaStocPerioada >= minP && ziSfarsitPerioada != -1 &&
(ziSfarsitPerioada - ziInceputPerioada > 1)) {
            existaPerioada  = 1;
            cout << ziInceputPerioada << " " << ziSfarsitPerioada <<  "
```

```
  " << sumaStocPerioada << endl;
        }

        if (existaPerioada == 0) {
            cout << "nu exista";
        }
        fin.close();
        return 0;
    }
```

Rezolvare exercitii recapitulative functii prelucrare siruri

Nota: enuntul il gasesti in fisierul pe care l-ai primit sesiunea trecuta 1. ```c++ #include #include

```
    using namespace std;


    int main(){
        char sir1[21], sir2[21];
        cin >> sir1;
        strcpy(sir2, sir1);
        cout << sir2 << endl;
    }
```

2. Rezolvare

```
    #include <iostream>
    #include <cstring>

    using namespace std;


    int main(){
        char sir1[21], sir2[21];
        cin >> sir1;
        cin >> sir2;
        strcat(sir1, sir2);
        cout << sir1;
    }
```

3. Rezolvare

```
    #include <iostream>
    #include <cstring>
```

```
    using namespace std;


    int main(){
        char sir1[21], sir2[21];
        cin >> sir1;
        cin >> sir2;
        int rezultatComparare = strcmp(sir1, sir2);
        if (rezultatComparare == 0) {
            cout << "sirurile sunt egale";
        } else if (rezultatComparare > 0) {
            cout  << "Al doilea sir este mai mic lexicografic decat al
doilea.";
        }  else {
            cout << "Primul sir este mai mic lexicografic decat al doilea.";
        }
    }
```

4. Rezolvare

```
    #include <iostream>
    #include <cstring>

    using namespace std;


    int main(){
        char sir1[21], rezultat[21];
        int n;
        cin >> sir1;
        cin >> n;
        strncpy(rezultat, sir1, n);

        cout << "Primele " << n << " caractere copiate: " << rezultat;
    }
```

5. Rezolvare

```
    #include <iostream>
    #include <cstring>

    using namespace std;


    int main(){
        char sir1[21], sir2[21];
        int n;
        cin >> sir1 >> sir2 >> n;
        strncat(sir1, sir2, n);
```

```
        cout << "Rezultatul concatenarii partiale: " << sir1;
    }
```

6. Rezolvare:

```
    #include <iostream>
    #include <cstring>

    using namespace std;


    int main(){
        char sir1[21], character;
        cin >> sir1 >> character;
        if (strchr(sir1, character) != NULL) {
            cout << character << " se gaseste in sir";
        } else {
            cout << character << " NU se gaseste in sir";
        }

    }
```

7. Rezolvare

```
    #include <iostream>
    #include <cstring>

    using namespace std;


    int main(){
        char sir1[21], character;
        cin >> sir1 >> character;
        int contor = 0;
        char *rezultat = strchr(sir1, character);
        while (rezultat != NULL) {
            contor++;
            rezultat = strchr(rezultat+1, character);
        }

        cout << "Caracterul " << character << " apare de " << contor << "
ori.";

    }
```

8. Rezolvare

```
#include <iostream>
#include <cstring>
#include <bits/fs_fwd.h>

using namespace std;


int main(){
    char sir1[21], sir2[21];
    cin >> sir1 >> sir2;
    if (strstr(sir1, sir2) != NULL) {
        cout << "Al doilea sir este subsir al primului";
    } else {
        cout << "Al doilea sir NU este subsir al primului";
    }
}
```

9. Rezolvare

```
#include <iostream>
#include <cstring>

using namespace std;


int main(){
    char sir1[21], sir2[21], rezultat[21];
    cin >> sir1 >> sir2;
    char *subsir = strstr(sir1, sir2);
    if (subsir != NULL) {

        *subsir = '\0'; // asta o explicam in clasa => ce face, termina
sirul inainte de subsire deoarece el pointeaza catre primul caracter din
subsirul gasit
        strcpy(rezultat, sir1); // copiaza prima parte
        strcat(rezultat, subsir + strlen(sir2)); // adauga partea de
dupa subsir
        cout << "Sirul dupa stergerea subsirului: " << rezultat << endl;
    } else {
        cout << "Sirul " << sir2 << " nu apare ca si subsir in sirul
dat";
    }
}
```

10. Rezolvare

```
#include <iostream>
#include <cstring>
```

```
    using namespace std;


    int main(){
        char sir[51], subsir[21], inlocuitor[21], rezultat[100]="";
        cin >> sir >> subsir >> inlocuitor;
        char *p = sir;             // pointer pentru parcurgerea sirului

        while (*p != '\0') {
            char *aparitieSubsir = strstr(p, subsir); // cauta subșirul in
s, incepand de la p

            if (aparitieSubsir != NULL) {
                // copiem partea inainte de subșir
                char temp[201];
                strncpy(temp, p, aparitieSubsir - p);
                temp[aparitieSubsir - p] = '\0';
                strcat(rezultat, temp);

                // adaugam subsirul de inlocuit
                strcat(rezultat, inlocuitor);

                // mutam pointerul dupa subșirul gasit
                p = aparitieSubsir + strlen(subsir);
            } else {
                // daca nu mai gasim subșirul, copiem restul
                strcat(rezultat, p);
                break;
            }
        }

        cout << "Sirul dupa inlocuire: " << rezultat << endl;

        return 0;
    }
```

Rezolvare exercitii functii cu explicare

1. Rezolvare:
    - Citim textul cu cin.getline() deoarece avem nevoie sa citim toate cuvintele
    - Dupa care cu strtok spargem textul in mai multe cuvinte
    - Pentru fiecare cuvant rezultat, folosind `strncat` unde `n` = 3 o sa le concatenam
2. Rezolvare
    - Parcurgem oricare dintre siruri cuvant cu cuvant folosind functia `strtok`
    - Pentru fiecare cuvant, apelam functia `strstr(sir, cuvant)` si daca este diferit de null inseamna ca exista
    - La final, daca nu avem niciun cuvant in comun, afisam "nu exista"
    - Nota: se pune un flag de true atunci cand gasim un cuvant in comun

3. Rezolvare
    - Parcurgem sirul cuvant cu cuvant folosind functia `strtok`
    - Pentru fiecare cuvant care are lungimea mai mare decat 4 (aici folosim `strlen`) il concatenam la rezultat, folosind functia `strcat`
4. Rezolvare
    - Parcurgem sirul cuvant cu cuvant folosind functia `strtok`
    - Pentru fiecare cuvant, concatenam ultimele 2 caractere la rezultat folosind functia `strcat`
        - Nota: ultimele doua caractere le putem copia astfel: `strcat(destinatie, cuvant + (lungimeCuvant - 2))`
5. Rezolvare
    - Parcurgem sirul cuvant cu cuvant folosind functia `strtok`
    - Copiem cuvintele intr-o matrice de cuvinte
    - Sortam cuvintele folosind bubble sort
    - Parcurgem matricea si concatenam in rezultat folosind `strcat` + spatiu dupa fiecare cuvant adaugat
6. Rezolvare
    - Parcurgem sirul cuvant cu cuvant folosind functia `strtok`
    - Pentru fiecare cuvant, verificam daca acesta are sau nu caracterul dat, folosind functia `strchr`
    - Daca acesta contine, ignoram cuvantul
    - Daca nu contine, cuvantul este adaugat la rezultat folosind functia `strcat`