Aceasta sesiune este o simulare a unei sesiuni de Bacalaureat

Adresa pentru varianta propusa

- https://www.modinfo.ro/bac/variante-test-2021/info/v1.pdf

Subiectul I

1. b.
2. b. (Executia se opreste la al treilea apel deoarece `n % 10` va fi diferit de `k`)
3. a. Toate celelalte variante sunt invalide din punct de vedere al sintaxei
4. d. Atentie mare la scrierea arborelui
5. c. Citind ca fiecare nod are gradul 1, putem avea un graf de genul: (1)-(2), (3)-(4), (5-6) deci avem 3 componente conexe. Atentie din nou la enunt!

Subiectul II

1.
   - a. Programul afiseaza `-1`. Se va executa pas cu pas functia afisata.
   - b. (oricare dintre numerele 7777, 7778, 7779, 7788, 7789, 7799, 7888, 7889, 7899,7999)
   - c:

```
int n, m = 10;
cin >> n;
if (n == 0) {
    m = 0;
} else {
    do {
        int c = n % 10;
        n = n / 10;
        if (c <= m) {
            m = c;
        } else {
            m = -1;
        }
    } while (n != 0);
}
cout << m;
```

   - d:

```
citeste n
m <- 10
daca n = 0 atunci
    m <- 0
altfel
```

```
        cat timp n != 0 executa (atentie la scrierea operatorului
`diferit de`)
            c <- n % 10;
            daca c <= m atunci m <- c
            altfel m <- -1
            n <- n / 10;
    scrie m
```

2. 149 167 347

3.
```
7
2020-2021
```

Subiectul III

1. Solutie:

```
void divX(int n, int x) {
    for (int i = n; i > 0; i--) {
        cout << x * i << " ";
    }
}
```

2. Solutie:

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

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


    // Afisare prima coloana
    for (int i = 0; i < 5; i++) {
        cout << matrice [i][0] << " ";
    }

    // Afisare prima linie
    for (int i = 1; i < 5; i++) {
        cout << matrice [n-1][i] << " ";
```

```
        }

        // Afisare ultima coloana
        for (int i = 0; i < n-1 ; i++) {
            cout << matrice[n-2-i][n-1] << " ";
        }

        // Afisare prima linie
        for (int i = n-2; i > 0; i--) {
            cout << matrice [0][i] << " ";
        }

        return 0;
    }
```

3.

- Limbaj natural
    - Folosim un vector de aparitie pentru numerele de 2 cifre pe care il actualizam in timp ce citim din fisier. Facem acest lucru deoarece, conform cerintei problemei, suntem interesati de numerele de 2 cifre, care NU apar in fisier, prin urmare, suntem interesati de numerele de 2 cifre ce au 0 aparitii.
    - Ne construim un subprogram pentru a identifica numerele de 2 cifre distincte. Acesta va functiona in felul urmator:
        - Verificam daca numarul este de 2 cifre, adica mai mare sau egal cu 10 si mai mic sau egal cu 99, in caz contrar returnam direct false (0)
        - Extragem mai intai ultima cifra, o taiem din numar si extragem si prima cifra. Daca cele doua sunt egale, se va returna false (0) altfel, se va returna true (1)
    - Citim din fisier numerele pe rand si actualizam vectorul de aparitii.
    - Dupa ce am terminat de citit din fisier, ne declaram doua variabile in care vom tine cele doua numere, conform cerintei problemei. Le vom initializa cu -1, valoare ce ne va ajuta mai tarziu ca sa verificam daca au fost sau nu gasite.
    - Cu o instructiune repetitiva ce are un numar finit de pasi, iteram de la 98, pana la 10 inclusiv, pentru a verifica vectorul de aparitii. Plecam direct de la 98 deoarece stim ca numarul 99 nu indeplineste conditia din cerinta problemei (sa aibe cifrele distincte).
        - Pentru fiecare numar, verificam daca are 2 cifre distincte SI 0 apariti
            - Daca da, mai facem o verificare:
                - Daca variabila in care tinem primul numar este egala cu -1, punem valoarea in aceasta variabila
                - Altfel, daca variabila in care tinem al doilea numar este egala cu -1, punem valoarea in aceasta variabila,
                - Altfel oprim iteratia deoarece stim ca am gasit cele doua numere
    - Dupa ce am iesit din instructiunea repetitiva, verificam daca ambele variabile au valori diferite de -1, caz in care le afisam, altfel afisam "nu exista"
    - Eficienta programului sta in faptul ca citim o singura data fisierul, timp in care populam vectorul de aparitii. De asemenea, algoritmul este eficientizat si pe parcurs deoarece odata ce am gasit numerele, instructiunea repetitiva este oprita, pentru a evita alte comparatii.
- Solutie:

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

using namespace std;
int areDouaCifreDistincte(int n);

int main() {
    ifstream f("bac.in");
    int frecventa[100] = {0};
    while (!f.eof()) {
        int number;
        f >> number;
        if (number >= 10 && number < 100) {
            frecventa[number]++;
        }
    }
    f.close();
    int primulNumar = -1;
    int alDoileaNumar = -1;
    for (int i = 98; i >= 10; i--) {
        if(areDouaCifreDistincte(i) && frecventa[i] == 0){
            if (primulNumar == -1) {
                primulNumar = i;
            } else if (alDoileaNumar == -1) {
                alDoileaNumar = i;
            } else {
                break;
            }
        }
    }

    if (primulNumar == -1 && alDoileaNumar == -1) {
        cout << "nu exista";
    } else {
        cout << primulNumar << " " << alDoileaNumar;
    }
}

int areDouaCifreDistincte(int n) {
    if(n < 10 || n > 99) {
        return 0;
    }
    int ultimaCifra = n % 10;
    n = n/10;
    int primaCifra = n % 10;
    return primaCifra != ultimaCifra;
}
```

Adresa pentru barem

- https://www.modinfo.ro/bac/variante-test-2021/info/b1.pdf