Ministerul Educației și Cercetării al Republicii Moldova
CENTRUL DE EXCELENȚĂ ÎN INFORMATICĂ ȘI TEHNOLOGII INFORMAȚIONALE

Lucrare individuală N-2

Tema: Tehnici de Programare

Disciplina: Programarea calculatoarelor
Varianta: 26

Grupa: P-2423
Elev: Pricop Maxim

Profesor: Cojocaru Liuba

Chișinău, 2025

Sarcina

Ghicire a unui număr ascuns.
Tu și calculatorul ați putea juca împreună următorul joc: unul dintre voi „se gândește” la un număr natural x cuprins între 1 și n, iar celălalt încearcă să-l ghicească, evident punând cât mai puține întrebări.
Singurele întrebări permise sunt:

- Numărul este egal cu ... ?
- Numărul este mai mare decât ... ?
- Numărul este mai mic decât ... ?

Celălalt jucător poate răspunde numai prin “Da” sau “Nu”.
Scrieți un program care să simuleze un astfel de joc.

Rezultat

```
Introdu limita n (superioara) pentru numarul ascuns (n >= 1): 40
Numarul ascuns poate fi intre 1 si 40.
Alege o optiune:
1. Numarul este egal cu ... ?
2. Numarul este mai mare decat ... ?
3. Numarul este mai mic decat ... ?
4. Preda-te (afiseaza numarul ascuns si iesi din program).
Optiunea aleasa: 2
Introdu numarul: 20
Nu, numarul ascuns nu este mai mare decat 20.


Alege o optiune:
1. Numarul este egal cu ... ?
2. Numarul este mai mare decat ... ?
3. Numarul este mai mic decat ... ?
4. Preda-te (afiseaza numarul ascuns si iesi din program).
Optiunea aleasa: 2
Introdu numarul: 10
Nu, numarul ascuns nu este mai mare decat 10.


Alege o optiune:
1. Numarul este egal cu ... ?
2. Numarul este mai mare decat ... ?
3. Numarul este mai mic decat ... ?
4. Preda-te (afiseaza numarul ascuns si iesi din program).
Optiunea aleasa: 2
Introdu numarul: 5
Da, numarul ascuns este mai mare decat 5.


Alege o optiune:
1. Numarul este egal cu ... ?
2. Numarul este mai mare decat ... ?
3. Numarul este mai mic decat ... ?
4. Preda-te (afiseaza numarul ascuns si iesi din program).
Optiunea aleasa: 1
Introdu numarul: 6
Felicitari! Ai ghicit numarul ascuns (6).
```

```
Introdu limita n (superioara) pentru numarul ascuns (n >= 1): 1000000
Numarul ascuns poate fi intre 1 si 1000000.
Alege o optiune:
1. Numarul este egal cu ... ?
2. Numarul este mai mare decat ... ?
3. Numarul este mai mic decat ... ?
4. Preda-te (afiseaza numarul ascuns si iesi din program).
Optiunea aleasa: 4
Numarul ascuns era: 476180
```

Rezolvare

Programul este mic, fiind alcătuit doar dintr-un fișier (main.cpp):

```cpp
#include <iostream>
#include <limits>
#include <random>

int main() {
    unsigned long long maxNumber = 1;
    unsigned long long randomNumber = 1;

    while (true) {
        std::cout << "Introdu limita n (superioara) pentru numarul ascuns (n >= 1): ";

        if (!(std::cin >> maxNumber)) {
            std::cout << "Numar invalid!\n\n\n";
            std::cin.clear();
            std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            continue;
        }

        break;
    }

    std::cout << "Numarul ascuns poate fi intre 1 si " << maxNumber << ".\n";

    std::random_device rd;
    std::mt19937_64 generator(rd());
    std::uniform_int_distribution<unsigned long long> dist(1, maxNumber);
    randomNumber = dist(generator);

    while (true) {
        std::cout << "Alege o optiune:\n";
        std::cout << "1. Numarul este egal cu ... ?\n";
        std::cout << "2. Numarul este mai mare decat ... ?\n";
        std::cout << "3. Numarul este mai mic decat ... ?\n";
        std::cout << "4. Preda-te (afiseaza numarul ascuns si iesi din program).\n";
        std::cout << "Optiunea aleasa: ";

        short int userChoice;
        if (!(std::cin >> userChoice)) {
            std::cout << "Optiune Invalida!\n\n\n";
            std::cin.clear();
            std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            continue;
        }
```

```
        if (userChoice != 1 && userChoice != 2 && userChoice != 3 && userChoice != 4) {
            std::cout << "Optiune Invalida!\n\n\n";
            continue;
        }

        if (userChoice == 4) {
            std::cout << "Numarul ascuns era: " << randomNumber << '\n';
            return 0;
        }

        unsigned long long guessedNumber;
        std::cout << "Introdu numarul: ";
        if (!(std::cin >> guessedNumber)) {
            std::cout << "Numar invalid!\n\n\n";
            std::cin.clear();
            std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            continue;
        }
        switch (userChoice) {
            case 1: {
                if (randomNumber == guessedNumber) {
                    std::cout << "Felicitari! Ai ghicit numarul ascuns (" << randomNumber
<< ").\n";
                    return 0;
                } else std::cout << "Numarul ascuns nu este " << guessedNumber <<
".\n\n\n";
                break;
            }
            case 2: {
                if (randomNumber > guessedNumber) std::cout << "Da, numarul ascuns este
mai mare decat " << guessedNumber << ".\n\n\n";
                else std::cout << "Nu, numarul ascuns nu este mai mare decat " <<
guessedNumber << ".\n\n\n";
                break;
            }
            case 3: {
                if (randomNumber < guessedNumber) std::cout << "Da, numarul ascuns este
mai mic decat " << guessedNumber << ".\n\n\n";
                else std::cout << "Nu, numarul ascuns nu este mai mic decat " <<
guessedNumber << ".\n\n\n";
                break;
            }
        }
    }

    return 0;
}
```

Concluzie

Studiul Individual a permis aplicarea practică a conceptelor de interacțiune om−mașină și gestionare a intrărilor în C++, consolidând abilități de proiectare a unei aplicații text simple, robuste și ușor de urmărit. Implementarea realizează un joc funcțional de ghicire a unui număr ascuns: generare aleatorie a unui număr în intervalul [1, n], meniu clar pentru întrebări binare („egal / mai mare / mai mic”), validare a intrării utilizatorului și mesaje explicite de tip „Da/ Nu”.

Testarea manuală pe scenarii tipice și de margine (valori minime/maxime pentru n, ghiciri corecte/incorecte, intrări nenumerice, opțiuni invalide și predare) a demonstrat stabilitatea comportamentului și recuperarea adecvată la erori de input. Fiecare interacțiune a utilizatorului este tratată în timp constant (O(1)), astfel experiența rămâne predictibilă indiferent de mărimea lui n.

Lucrarea atinge obiectivele Studiului Individual: oferă o implementare clară, funcțională și comentată a cerinței, demonstrând competențe de validare, utilizare a generatorului de numere random și structurare a interfeței utilizator.

În concluzie, proiectul oferă un exemplu simplu dar complet de aplicație interactivă în C++, potrivit pentru ilustrarea conceptelor de input/output, gestionare erori și algoritmi simpli, și constituie un punct de plecare solid pentru dezvoltări ulterioare și discuții mai avansate despre performanță și strategii de căutare.