Universitatea Tehnică de Construcții din București
Facultatea de Hidrotehnică
Specializarea: Automatică și Informatică Aplicată

# *Jocul:*

# **PIATRA HARTIE** sau **FOARFECA**

# *(IN C++)*

*realizat de:*

*Irina Dumitrescu*

*Hidrotehnica AIA anul I*

Universitatea Tehnică de Construcții din București
Facultatea de Hidrotehnică
Specializarea: Automatică și Informatică Aplicată

# *CUPRINS*

# 1 Introducere:

## Motivarea temei alese:

Am ales să implementez jocul „Piatra, Hârtie, Foarfecă” utilizând limbajul de programare C++ în cadrul mediului de dezvoltare Visual Studio Code, deoarece această temă, deși aparent simplă, oferă un cadru didactic adecvat pentru aprofundarea unor concepte fundamentale din programare. În primul rând, proiectul implică utilizarea structurilor de control (condiționale și repetitive), a funcțiilor și a manipulării datelor de tip caracter și întreg, ceea ce contribuie semnificativ la consolidarea cunoștințelor de bază în programarea procedurală. În plus, dezvoltarea unui astfel de joc presupune și o componentă de interacțiune cu utilizatorul, favorizând dezvoltarea unei gândiri logice și algoritmice prin modelarea deciziilor și gestionarea cazurilor posibile. Mediul Visual Studio Code a fost ales datorită flexibilității sale, a suportului extins pentru C++ și a integrării eficiente cu extensii utile pentru depanare și compilare. Astfel, această temă nu doar că oferă o punte între teorie și practică, dar creează și oportunitatea de a transpune un concept ludic într-o aplicație funcțională, contribuind la dezvoltarea creativității și a competențelor digitale.

Istoric joc:

Jocul de piatra, hartie si foarfeca are o istorie lunga si interesanta, care dateaza din antichitate. Desi nu se stie exact cand a fost inventat, se crede ca originile jocului sunt in China, in jurul anului 206 i.Hr. – 220 d.Hr., in perioada dinastiei Han. In acea perioada, jocul era cunoscut sub numele de "shoushiling" si era folosit pentru a rezolva conflicte sau pentru a lua decizii intr-un mod amuzant si rapid. In aceasta versiune timpurie, semnele folosite erau diferite de cele de astazi, dar principiul ramanea acelasi: fiecare semn invinge un alt semn si este invins de altul.

Jocul a fost introdus in Japonia in jurul secolului al XVII-lea, unde a devenit foarte popular si a fost denumit "jan-ken". In aceasta varianta, semnele folosite erau piatra, hartie si foarfeca, iar regulile erau aceleasi ca cele pe care le stim astazi. "Jan-ken" este derivat din cuvintele japoneze "jan" (care inseamna "a lovi") si "ken" (care inseamna "mana"). Astfel, semnele reprezinta actiuni: piatra loveste foarfeca, foarfeca taie hartia, iar hartia acopera piatra.

In Japonia, "jan-ken" a fost folosit nu doar ca joc de distractie, dar si ca metoda de luare a deciziilor, fiind adesea folosit pentru a alege cine va face anumite sarcini sau cine va castiga un conflict minor.

Jocul a ajuns in Europa si in America in secolul al XX-lea, odata cu globalizarea si cresterea interactiunii intre culturi. In special in Statele Unite, jocul a devenit foarte popular in anii 1950-1960 si a fost adus in cultura pop prin filme, carti si chiar emisiuni de televiziune. A fost folosit frecvent in diverse contexte sociale, de la decizii informale la jocuri si concursuri.

Astăzi, piatra, hartie si foarfeca este un joc universal cunoscut, jucat de oameni de toate varstele si din toate colturile lumii. De asemenea, a fost adaptat

in diverse forme, cum ar fi variantele online sau jocurile video, si este adesea folosit in competitii si ca un mod rapid de a rezolva conflicte minore.

Obiective:

Scopul principal al realizării proiectului „Piatră, Hârtie, Foarfecă” în limbajul C++ este de a consolida cunoștințele teoretice prin aplicarea practică a conceptelor fundamentale ale programării structurale. Prin intermediul acestei aplicații interactive, se urmărește dezvoltarea abilităților de gândire algoritmică, perfecționarea logicii de ramificare și selecție, precum și familiarizarea cu principiile de bază ale interacțiunii om-calculator. În plus, proiectul contribuie la înțelegerea modului în care pot fi construite aplicații simple, dar funcționale, care implică procese decizionale și generare aleatorie. Un alt obiectiv important este acomodarea cu utilizarea unui mediu de dezvoltare profesional (Visual Studio Code), care oferă facilități moderne pentru scrierea, testarea și depanarea codului sursă.

Obiective specifice ale proiectului:

- Aplicarea practică a instrucțiunilor condiționale (if/else, switch).
- Utilizarea funcțiilor pentru structurarea logicii programului.
- Exersarea manipulării datelor de tip char și int.

- Implementarea unei logici de joc bazată pe decizii multiple.
- Dezvoltarea unei interfețe simple prin interacțiune în consolă.
- Exersarea etapelor de testare și depanare a codului în Visual Studio Code.
- Îmbunătățirea clarității și organizării codului prin utilizarea comentariilor și a unei structuri coerente.

Argumentarea temei alese:

Alegerea temei „Implementarea jocului Piatra-Hârtie-Foarfecă în limbajul C++” se justifică prin relevanța sa didactică și capacitatea de a demonstra, într-un cadru practic, aplicabilitatea noțiunilor fundamentale de programare procedurală. Această temă permite explorarea unor concepte esențiale precum utilizarea structurii condiționale, funcțiilor, manipularea fluxurilor de intrare-ieșire și generarea de numere aleatorii, într-un context accesibil și intuitiv. Prin simplitatea logicii de joc și prin interacțiunea constantă cu utilizatorul, aplicația oferă un cadru optim pentru consolidarea abilităților de programare, fără a presupune o complexitate excesivă. Totodată, tema aleasă permite evidențierea importanței validării datelor introduse de utilizator și a structurării clare a codului, aspecte fundamentale în formarea unei gândiri algoritmice riguroase. Astfel, lucrarea reflectă nu doar capacitatea de a transpune o idee simplă într-un program funcțional, ci și înțelegerea principiilor de bază ale dezvoltării software.

2 Tehnologii utilizate și studiu de caz:

Programul prezentat este dezvoltat utilizând limbajul de programare C++, un limbaj de nivel înalt, compilat, care oferă suport atât pentru programarea procedurală, cât și pentru cea orientată pe obiecte. C++ este ales în mod frecvent în dezvoltarea aplicațiilor care necesită performanță, control detaliat asupra resurselor și portabilitate pe diverse platforme. În contextul acestui joc, C++ este folosit pentru gestionarea logicii aplicației, a interacțiunii cu utilizatorul și a controlului fluxului programului.

Pentru interacțiunea cu utilizatorul, programul utilizează biblioteca standard <iostream>, care oferă funcționalități de bază pentru operații de intrare și ieșire. Această bibliotecă permite afișarea mesajelor în consolă (std::cout) și citirea datelor introduse de utilizator (std::cin). Comunicarea în timp real cu

jucătorul este esențială în aplicații interactive, iar utilizarea acestei biblioteci simplifică procesul de captare a inputului și afișare a rezultatului.

De asemenea, programul face uz de biblioteca <cstdlib> pentru generarea aleatorie a alegerilor calculatorului, prin funcția rand(). Pentru a asigura variabilitate între execuții, funcția srand() din biblioteca <ctime> este folosită pentru a seta o sămânță pseudoaleatorie în funcție de timpul sistemului (time(0)). Această combinație de funcții permite ca rezultatul fiecărei runde să fie imprevizibil, emulând astfel comportamentul „aleatoriu” al adversarului virtual.

În plus, biblioteca <limits> este utilizată pentru a curăța fluxul de intrare atunci când utilizatorul introduce date invalide. Aceasta este o tehnică de validare a inputului, esențială în aplicații interactive pentru a evita comportamentele neașteptate cauzate de introducerea accidentală a valorilor incorecte. Se folosește cin.clear() și cin.ignore() împreună cu numeric_limits<streamsize>::max() pentru a reseta starea fluxului și a elimina caracterele rămase în buffer.

În ceea ce privește stilul de implementare, se adoptă o abordare procedurală, structurând codul în funcții precum numeOptiune() și alegereCalculator() pentru a încuraja reutilizarea codului și pentru a îmbunătăți lizibilitatea acestuia. Deși nu este utilizată programarea orientată pe obiecte în această versiune, separarea logicii în funcții facilitează extinderea ulterioară a aplicației.

Au fost utilizate mai multe biblioteci standard:

- <iostream> – pentru operații de intrare/ieșire: afișarea de mesaje către utilizator (cout) și citirea opțiunilor introduse (cin).

- <cstdlib> – pentru utilizarea funcției rand(), necesară generării aleatorii a alegerii calculatorului.

- <ctime> – pentru inițializarea generatorului aleator (srand(time(0))), astfel încât rezultatele să fie diferite la fiecare rulare.

- <limits> – pentru controlul erorilor de input (cin.fail()) și curățarea bufferului cu cin.ignore() și numeric_limits<streamsize>::max().

- main() – punctul de intrare al programului, în care se desfășoară logica principală a jocului.

- Funcție numeOptiune(int) – mapare între valorile numerice 1, 2, 3 și denumirile „Piatra”, „Hartie”, „Foarfeca” (folosește switch).

- Funcție alegereCalculator() – returnează un număr aleator între 1 și 3, simulând alegerea făcută de calculator.

- Funcție curataEcran() – folosește cout << string(50, '\n') pentru a simula „curățarea” ecranului în consolă.

- Funcția srand() și time(0) – pentru inițializarea generatorului pseudoaleator în mod variabil.

- while (true) – pentru bucla de validare a inputului utilizatorului (asigură că se introduc doar valori valide: 1, 2, 3).

- do-while – permite rularea repetată a jocului până când utilizatorul optează să iasă.

- if / else if / else – determinarea rezultatului rundei (egalitate, victorie, înfrângere).

- switch-case – folosit în funcția numeOptiune pentru returnarea numelui opțiunii în funcție de valoarea numerică.

- Controlul erorilor de input – prin cin.fail(), cin.clear() și cin.ignore() pentru a evita blocarea aplicației în caz de introducere invalidă.

- Programare modulară – prin utilizarea funcțiilor separate pentru diferite componente ale jocului (modularitate și claritate în cod).

- Scorul global – implementat prin variabile globale în cadrul funcției main() (ex: scorJucator, scorCalculator).

- Pseudoaleatoriu – generarea unui comportament imprevizibil pentru calculator prin utilizarea generatorului de numere aleatorii (rand()).

Descrierea funcționalității:

La rularea programului, utilizatorul este întâmpinat cu un meniu simplu, în care poate selecta o opțiune (piatră, hârtie sau foarfecă). Alegerea este introdusă prin tastatură și validată. Calculatorul își generează alegerea în mod aleatoriu. Programul compară cele două alegeri și afișează rezultatul: egalitate, victorie sau înfrângere, în funcție de regulile bine cunoscute ale jocului.

Scorul este actualizat după fiecare rundă și afișat pentru a oferi un feedback continuu utilizatorului. La finalul fiecărei runde, jucătorul este întrebat dacă dorește să continue, putând relua jocul sau încheia sesiunea. În final, se afișează scorul total și un mesaj corespunzător rezultatului general.

De ce CodeBlocks?:

Pentru realizarea proiectului am optat pentru utilizarea mediului de dezvoltare integrat (IDE) Code::Blocks, datorită multiplelor sale avantaje care îl recomandă în mod special pentru aplicații scrise în limbajul C++. Code::Blocks oferă o interfață intuitivă, ușor de utilizat, chiar și pentru programatorii aflați la început de drum, facilitând procesul de scriere, compilare și depanare a codului sursă. Unul dintre principalele atuuri ale acestui mediu este compatibilitatea sa ridicată cu diverse compilatoare, precum GCC, precum și suportul pentru evidențierea sintaxei și autocompletare, ceea ce contribuie la reducerea erorilor și la îmbunătățirea productivității. De asemenea, Code::Blocks permite rularea programelor în mod rapid, oferind un feedback imediat asupra comportamentului aplicației, aspect esențial în procesul iterativ de testare și perfecționare a codului. În contextul acestui proiect, alegerea acestui IDE s-a dovedit a fi una eficientă și practică, facilitând implementarea unei aplicații robuste și ușor de întreținut.

Rezultate de funcționare:

Aplicația funcționează conform așteptărilor și răspunde corect la toate scenariile testate: introducere validă și invalidă, egalitate, victorie, înfrângere și întreruperea jocului. Modul de implementare a validării inputului și utilizarea generării pseudoaleatorii conferă o experiență interactivă și imprevizibilă, esențială pentru un joc de acest tip.

3. Implementarea functionalitatii:

La începutul codului sunt incluse mai multe biblioteci standard ale limbajului C++, esențiale pentru funcționalitățile jocului. Biblioteca iostream permite interacțiunea cu utilizatorul prin funcțiile cin și cout. cstdlib și ctime sunt necesare pentru generarea aleatorie a alegerii calculatorului, iar limits este utilizată pentru a curăța inputul în cazul în care utilizatorul introduce o valoare greșită.

```
1    #include <iostream>
2    #include <cstdlib>
3    #include <ctime>
4    #include <limits>
```

Pentru a menține un cod clar și modular, au fost definite funcții auxiliare. Spre exemplu, funcția numeOptiune() transformă opțiunile numerice ale utilizatorului și ale calculatorului într-un text descriptiv. Această funcție oferă lizibilitate în afișările din consolă, facilitând înțelegerea rezultatului rundei.

```
string numeOptiune(int alegere) {
    switch (alegere) {
        case 1: return "Piatra";
        case 2: return "Hartie";
        case 3: return "Foarfeca";
        default: return "Necunoscut";
    }
}
```

Folosind funcția rand() % 3 + 1, este generată o valoare aleatoare între 1 și 3, corespunzând alegerilor posibile: piatră, hârtie, foarfecă. Aceasta este apelată în interiorul buclei principale a jocului pentru a simula decizia calculatorului.

```
int alegereCalculator() {
    return rand() % 3 + 1;
}
```

Pentru a păstra consola curată între runde, se folosește o metodă simplificată, compatibilă cu majoritatea IDE-urilor: Deși nu este un clear screen real, această metodă oferă impresia de „refresh” în consolă prin spațiere verticală.

```
void curataEcran() {
    cout << string(50, '\n');
}
```

În funcția main(), este folosită o buclă while (true) pentru a forța utilizatorul să introducă o valoare validă. Astfel, introducerea este verificată pentru a evita erori sau blocaje. Această metodă îmbunătățește experiența utilizatorului și face programul mai robust.

```
while (true) {
    cout << "\n Alege: \n";
    cout << " 1. Piatra\n 2. Hartie\n 3. Foarfeca\n";
    cout << "Introdu optiunea ta: ";
    cin >> alegereJucator;

    if (cin.fail() || alegereJucator < 1 || alegereJucator > 3) {
        cout << "EROARE!!!\n";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    } else {
        break;
    }
}
```

După ce ambele părți (jucătorul și calculatorul) au făcut o alegere, rezultatul este stabilit printr-o serie de condiții: Codul tratează atât egalitatea, cât și condițiile de câștig pentru jucător sau calculator, actualizând scorurile corespunzător.

```
if (alegereJucator == alegerePC) {
    cout << "EGALITATE\n ";
} else if ((alegereJucator == 1 && alegerePC == 3) ||
           (alegereJucator == 2 && alegerePC == 1) ||
           (alegereJucator == 3 && alegerePC == 2)) {
    cout << "AI CASTIGAT \n";
    scorJucator++;
} else {
    cout << "DA BANII! SI MAI INCEARCA O DATA \n";
    scorCalculator++;
}
```

După fiecare rundă, utilizatorul este întrebat dacă dorește să continue. Astfel, jocul este reluat doar dacă utilizatorul introduce 'd' sau 'D', permițând controlul interactiv asupra buclei de joc.

```
        cout << "\n Mai bagi o fisa? (da/nu): \n";
cout << "============================================================\n";
        cin >> joacaDinNou;

        while (joacaDinNou != 'd' && joacaDinNou != 'D' &&
               joacaDinNou != 'n' && joacaDinNou != 'N') {
            cout << "Raspunde cu 'd' pentru da sau 'n' pentru nu: ";
            cin >> joacaDinNou;
```

La încheierea sesiunii de joc, este afișat scorul final și un mesaj corespunzător rezultatului general. Această secțiune finalizează experiența utilizatorului și oferă feedback personalizat, ceea ce sporește atractivitatea aplicației.

```
cout << "Joc terminat! \n";
cout << "Scor final: Tu " << scorJucator << " - " << scorCalculator << " Cal \n";
ostream std::cout
if (scorJucator > scorCalculator) {
    cout << "Ai castigat razboiul! \n";
} else if (scorJucator < scorCalculator) {
    cout << "Calculatorul a castigat \n";
} else {
    cout << "La fel de buni amandoi :/ \n";
}
```

Proiectul propus implementează un joc clasic într-o manieră interactivă și robustă, folosind concepte fundamentale din C++ precum: funcții, variabile, structuri de control și validare a inputului. Codul este structurat clar, modular și este potrivit atât pentru exerciții educaționale, cât și pentru proiecte introductive în programare.

## 4 Concluzie:

Programul prezentat reprezintă o aplicație interactivă de tip joc clasic "Piatra-Hartie-Foarfeca", implementată în limbajul de programare C++. Printr-o structură logică clară, utilizatorul este ghidat într-un ciclu repetitiv de luare a deciziilor, în care interacționează cu sistemul prin introducerea unei opțiuni

numerice. Programul utilizează funcții fundamentale ale limbajului C++, precum instrucțiuni de control al fluxului, funcții personalizate, manipularea fluxului de intrare-ieșire și generarea de valori aleatorii.

Pe lângă funcționalitatea de bază a jocului, aplicația oferă un mecanism de validare a introducerii utilizatorului, o prezentare interactivă a scorului și posibilitatea de a relua jocul în mod repetat. Prin utilizarea funcției srand() în combinație cu time(0), este asigurată o variabilitate a deciziilor calculatorului, crescând astfel gradul de imprevizibilitate și realism al interacțiunii.

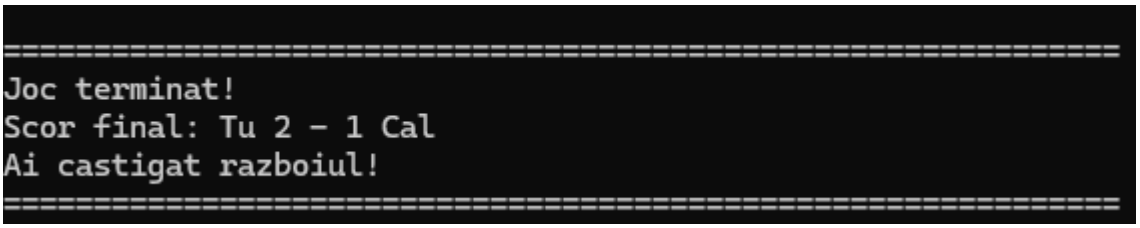
Din punct de vedere didactic, acest program oferă un exemplu relevant de aplicare a noțiunilor fundamentale de programare procedurală, fiind util atât pentru învățarea structurii unui program C++, cât și pentru înțelegerea modului în care logica decizională poate fi transpusă într-o interfață utilizator simplă. Totodată, prin introducerea unor elemente de umor și personalizare a mesajelor, aplicația capătă un caracter ludic, ceea ce contribuie la creșterea atractivității pentru utilizator.

În concluzie, acest proiect reușește să îmbine concepte de bază ale programării cu o experiență interactivă plăcută, servind drept model eficient pentru inițierea în dezvoltarea de aplicații C++ orientate către utilizator.

5 Rezultate finale:

```
===== Bine ai venit la jocul PIATRA - HARTIE - FOARFECA! =====
============================================================

 Alege:
 1. Piatra
 2. Hartie
 3. Foarfeca
Introdu optiunea ta: 1

Alegerea ta: Piatra
Alegere comp Piatra
EGALITATE

 Scor curent:
Tu: 0 | Cal: 0

 Mai bagi o fisa? (da/nu):
============================================================
```

```
============================================================

 Alege:
 1. Piatra
 2. Hartie
 3. Foarfeca
Introdu optiunea ta: 23
EROARE!!!

 Alege:
 1. Piatra
 2. Hartie
 3. Foarfeca
Introdu optiunea ta:
```




6 Bibliografie:

https://www.w3schools.com/cpp/

https://www.youtube.com/watch?v=-TkoO8Z07hI

https://wiki.codeblocks.org/index.php/Basic_Tutorial

https://www.pbinfo.ro/ <3

https://www.onlinegdb.com/