# *LUCRARE PRACTICĂ PENTRU OBŢINEREA ATESTATULUI PROFESIONAL*

PROF. COORDONATOR: PRUȘ PAUL

ELEV: PETREA ANDREI

# POKER ÎN 3 CĂRȚI

# Cuprins

# Introducere

Poker-ul este un joc popular de cărți, unde proporția între noroc si strategia jocului (cacealma) este de 80/20 în favoarea strategiei, în care jucătorii mizează pe valoarea superioară a combinației de cărți ("mâna") aflată în posesia lor, pariind sume de bani pentru obținerea potului, adică a sumei puse în joc.

Câștigător este jucătorul care deține mâna cu cea mai mare valoare în conformitate cu ierarhia cărților de joc, sau cel care rămâne cu cărțile în mână după ce toți ceilalți jucători au renunțat să joace turul respectiv.

Poker-ul are mai multe variante de joc, toate fiind însă similare, una dintre cele mai populare fiind poker-ul în 3 cărţi.

C++ este un limbaj de programare general, compilat. Este un limbaj multi-paradigmă, cu verificarea statică a tipului variabilelor ce suportă programare procedurală, abstractizare a datelor, programare orientată pe obiecte. În anii 1990, C++ a devenit unul dintre cele mai populare limbaje de programare comerciale, rămânând astfel până azi.

Internetul a permis ca noi tipuri de jocuri de noroc să fie disponibile online. Îmbunătățirile tehnologice au schimbat obiceiurile de pariere la fel cum terminalele de loterie video, keno și cartele cu zgârieturi au schimbat industria jocurilor de noroc în secolul al XX-lea. Jocurile de noroc au devenit una dintre cele mai populare și profitabile afaceri de pe Internet.

Aşadar era inevitabil ca şi poker-ul în 3 cărţi să fie transpus in lumea digitala prin intermediul limbajului C++, arhiprezent astazi în lumea IT.

Pe parcursul acestei lucrări, sunt prezentate în detalui regulamentul jocului şi elemente de sintaxa C++ , ca în final să coroborăm cunoştiinţele pentru realizarea programului.

# Capitolul I: Poker-ul în 3 cărţi

## 1.1 Origine si istorie

Varianta de cazinou a Poker-ului în 3 cărti a fost creată pentru prima dată de Derek Webb în 1994 și patentată în 1997. Scopul lui Webb a fost să creeze o versiune de poker care să se joace la fel de rapid ca şi celelalte jocuri de masă. Pentru Webb a fost important să obțină amestecul corect dintre cei trei factori importanți pentru orice joc de cazino: regulile jocului să fie ușor de înțeles, plățile să fie suficient de mari pentru a atrage jucători, iar câştigul casei să fie suficient pentru ca proprietarii de cazinouri să fie interesați de adoptarea jocului.

Webb a înființat o afacere numită Prime Table Games pentru a comercializa jocul atât în Statele Unite, cât și în Regatul Unit. British Casino Association, cunoscută acum sub numele de National Casino Industry Forum (NCiF), a sugerat că Webb să câștige mai întâi o experiență în SUA, deoarece Regatul Unit avea reglementări împotriva unui astfel de joc de masă și aplicația sa nu era suficient de puternică pentru a convinge autoritățile de reglementare pentru a face modificări semnificative la regulile și regulamentele lor pentru un joc nou.

Primul care a adoptat jocul a fost Barry Morris, vicepreședinte al Grand Casino Gulfport din Mississippi , după ce Webb a avut vânzări nereușite cu proprietarii de cazinouri din Reno, Las Vegas și Atlantic City. Un aspect cheie al ofertei lui Webb către Morris a fost să stea la masă pentru a-i antrena pe dealeri înșuși, precum și să urmărească pentru a se asigura că jocul a fost jucat corect. Reglementările Regatului Unit privind jocurile de noroc au fost modificate pentru a permite introducerea Poker-ului în 3 cărti în 2002.

Prime Table Games a continuat să comercializeze Poker-ului în 3 cărti până în 1999, când Shuffle Master a dobândit drepturile asupra jocului în afara insulelor britanice. Vânzarea a fost determinată de un proces inițiat în acel an în instanța federală americană de către Progressive Gaming International Corporation (PGIC), proprietarii de atunci ai stud poker-ului din Caraibe, prin care se pretinde încălcarea brevetului; Shuffle Master a fost de acord să apere acest litigiu ca parte a achiziției. Ulterior, în 2007, Prime Table Games a arătat într-o contra-acțiune că litigiul PGIC din 1999 se baza pe revendicări de brevet invalide; PGIC s-a stabilit pentru 20 de milioane de dolari. Mai mult, Prime Table Games a intentat o acțiune împotriva Shuffle Master în 2008, acuzând

parțial că Shuffle Master știa că cererile PGIC nu erau valabile înainte de cumpărarea din 1999; ulterior a fost stabilit pentru peste 2 milioane de dolari.

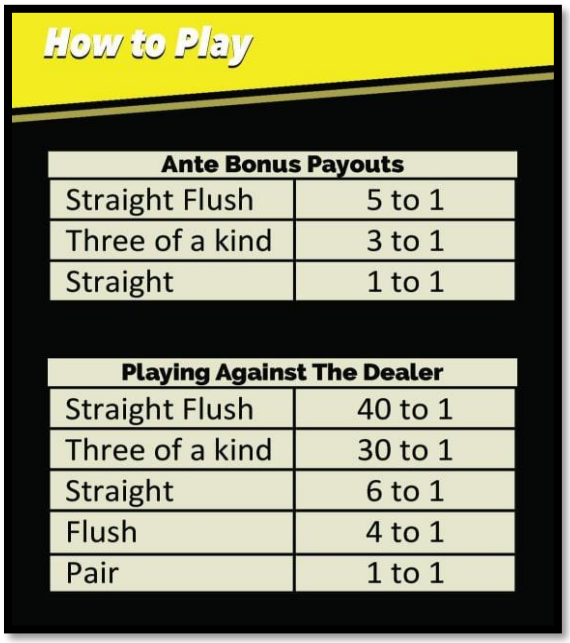
## 1.2 Regulament

Poker-ului în 3 cărti este jucat ca un duel dintre mâna jucătorului și mâna dealerului. După plasarea tuturor pariurilor initiale(**ante**),trei cărți sunt împărțite fiecărui jucător și dealer. Jucătorii au de ales fie să se retrag, fie să continue în joc, plasând un pariu „de joc" egal cu ante-ul. Mâinile sunt apoi expuse și pariurile rezolvate.

Mâna dealerului trebuie să aibe macar o Regină sau o combinaţie egala pentru ca mâna dealerului să poată juca. Dacă dealerul nu joacă, atunci nu există nicio acțiune la pariurile de joc, iar jucatorii işi primesc banii inapoi. Dacă dealerul joacă, mâinile dealerului și ale jucătorului sunt comparate. Dacă mâna jucătorului pierde, atât pariul ante cât și pariul de joc se pierd. Dacă mâna jucătorului câștigă atât pariul ante cât și pariul de joc sunt plătite de 1 la 1. Dacă mâinile sunt egale, atunci nu există nicio acțiune în niciunul dintre pariuri.

Pariuri suplimentare opționale sunt oferite. Pariul **Pair Plus** este pariul jucătorului că mâna sa va conţine o pereche. Pariul Plus Pair se câștigă dacă mâna are cel puțin un flush. Rambursul se aplică indiferent de mâna dealerului, deoarece pariul Pair Plus nu se află în concurență cu mâna dealerului. Unele cazinouri oferă, de asemenea, un bonus **Ante**, care este plătit pe pariul ante pentru un straight sau mai bun. Tabelul tipic Ante Bonus plătește 5 la 1 pentru o tripla royala, 4 la 1 pentru trei de același tip(**flush**) și 1 la 1 pentru un straight. Ca și pariul Pair Plus, bonusul Ante se plătește indiferent dacă acea mână bate mâna dealerului.


Ordinea perechilor




Câștiguri în funcție de mână

## 1.3 Variaţii

Unele cazinouri au adăugat un pariu numit **Prime**,iar jocul este cunoscut sub numele de **Three Three Card Poker**. Pariul Prime este plasat opțional înainte ca cărțile să fie distribuite și se plăteşte pentru culoarea cărților jucătorului. Dacă toate cele trei cărți au aceeași culoare, recompensa este de 3 la 1. Cu toate acestea, atunci când este inclusă cu mâna dealerului, dacă toate cele șase cărți au aceeași culoare, recompensa se mărește la 4 la 1.

O altă variantă este „bonusul cu șase cărți”, în care jucătorii primesc o plată bazată pe cea mai bună mână de poker cu cinci cărți care poate fi făcută folosind orice combinație dintre cele trei cărți ale jucătorului și cele trei cărți ale dealerului. Rata de plată variază de la 5 la 1 pentru trei de un fel la 1000 la 1 pentru chinta regală. Plățile sunt plătite indiferent dacă se plătesc și alte pariuri.

# Capitolul II: C++

## 2.1 Istorie

Bjarne Stroustrup a început să lucreze la "C cu clase" în 1979. Ideea creării unui nou limbaj a venit din experiența de programare pentru pregătirea tezei sale de doctorat. Stroustrup a descoperit că Simula avea facilități foarte utile pentru proiecte mari, însă era prea lent, în timp ce BCPL era rapid, însă nu era de nivel înalt și era nepotrivit pentru proiecte mari. Când a început să lucreze pentru Bell Labs, avea sarcina de a analiza nucleul UNIX referitor la calcul distribuit. Amintindu-și de experiența sa din perioada lucrării de doctorat, Stroustrup a început să îmbunătățească C cu facilități asemănătoare cu Simula.C a fost ales deoarece era rapid și portabil. La început facilitățile adaugate C-ului au fost clase, clase derivate, verificare a tipului, inline și argumente cu valori implicite.

În timp ce Stroustrup a proiectat C cu clase (mai apoi C++), a scris de asemenea și Cfront, un compilator care genera cod sursă C din cod C cu clase. Prima lansare comercială a fost în 1985.

În 1982, numele limbajului a fost schimbat de la C cu clase la C++. Au fost adaugate noi facilități, inclusiv funcții virtuale, supraîncărcarea operatorilor și a funcțiilor, referințe, constante, alocare dinamică, un control al tipului mai puternic și noua variantă de comentariu pe un singur rând (liniile care încep cu caracterele '//').

În 1985 a fost lansată prima ediție a cărții "The C++ Programming Language" (Limbajul de programare C++), oferind informații importante despre limbaj, care încă nu era un standard oficial. În 1989 a fost lansată versiunea 2.0 a C++. Au apărut acum moștenirea multiplă, clase abstracte, funcții statice, funcții constante și membri protected. În 1990 o altă carte a fost lansată, oferind suport pentru standarde viitoare. Ultimele adăugări includeau template-uri, excepții, spații de nume (namespace-uri) și tipul boolean.

O dată cu evoluția limbajului C++, a evoluat și o bibliotecă standard. Prima adăugire a fost biblioteca de intrări/ieșiri (I/O stream), care oferea facilități pentru a înlocui funcțiile tradiționale C cum ar fi printf și scanf. Mai târziu, printre cele mai semnificative adăugari la biblioteca standard a fost STL (Standard Template Library) (Biblioteca de formate standard).

După ani de lucru, un comitet ANSI-ISO a standardizat C++ în 1998 (ISO/IEC 14882:1998).

## 2.2 Variabile şi tipuri de date

În C++ exista mai multe tipuri de date definite printr-un cuvânt cheie. Variabilele sunt containere ce stocheaza datele pe baza tipului de date corespunzător. Sintaxa atribuirii unei variabile o valoare este:

tipul_datei numele_variabilei = valoarea;

Tipuri de date:

- **int :** stocheaza numere intregi pe 4 octeţi
- **short** : stocheaza numere intregi pe 2 octeţi
- **long long** stocheaza numere intregi pe 8 octeţi
- **float :** stocheaza numere reale pe 4 octeţi cu 7 zecimale
- **double :** stocheaza numere reale pe 8 octeţi cu 15 zecimale
- **char :** stocheaza un caracter ASCII pe 1 octet
- **bool :** stocheaza valoarea booleana(adevarat/fals) pe 1 octet

Cuvantul "**unsigned**" pus in fata tipului de date int,short,long long,float şi double restrange intervalul la doar numere pozitive.

| Type | Size (in bytes) | Range |
|---|---|---|
| char | 1 | -127 to 127 or 0 to 255 |
| unsigned char | 1 | 0 to 255 |
| int | 4 | -2147483648 to 2147483647 |
| unsigned int | 4 | 0 to 4294967295 |
| short int | 2 | -32768 to 32767 |
| unsigned short int | 2 | 0 to 65,535 |
| long int | 4 | -2147483648 to 2147483647 |
| unsigned long int | 4 | 0 to 4294967295 |
| float | 4 | +/- 3.4e +/- 38 (~7 digits) |
| double | 8 | +/- 1.7e +/- 308 (~15 digits) |

## 2.3 Structuri de control

În C++ exista 3 tipuri de structuri de control

- **structura liniară**
- **structura repetitivă**
- **structura alternativă**

**Structura liniară** este reprezentată de instrucțiuni care se execută la fel la fiecare executare a programului (sau a secvenței de program), indiferent care sunt valorile variabilelor cu care se lucrează. Instrucțiunea **expresie** este cel mai frecvent folosit tip de instrucțiune dintr-un program C++. O expresie devine instrucțiune dacă este urmată de ;

Expresie;

În anumite situații, este necesară executarea unor instrucțiuni în cadrul unui program numai în anumite condiții. Numite și structuri de decizie, **structurile alternative** permit rezolvarea unor asemenea situații. Instrucțiunea **if** este cea mai utilizată structură alternativă şi are 2 forme:

```
if ( Expresie )
  Instrucțiune1
else
  Instrucțiune2
```

```
if ( Expresie )
  Instrucțiune1
```

**Structurile repetitive** execută o instrucțiune de un anumit număr de ori, sau cât timp o condiție este adevărată. Se mai numesc și bucle sau cicluri.

Structurile repetitive pot fi:

- cu **număr cunoscut de pași** (iterații) – se cunoaște de la început de câte ori se va execută instrucțiunea
- cu **număr necunoscut de pași** (iterații). Instrucțiunea se execută cât timp o condiție este adevărată. La fiecare pas se va evalua condiția, iar dacă aceasta este adevărată se va executa instrucțiunea.

Structurile repetitive cu număr necunoscut de pași pot fi:

- cu **test inițial**: mai întâi se evaluează condiția; dacă este adevărată se execută instrucțiunea și procesul se reia.
- cu **test final**: mai întâi se execută instrucțiunea, apoi se evaluează condiția; Dacă este adevărată, procesul se reia.

Instrucțiunea care se execută în mod repetat poartă numele de corp al structurii repetitive, corp al ciclului, corp al buclei și de foarte multe ori este o instrucțiune compusă.

```
while (Expresie)
    Instructiune
```

```
do
    Instructiune
while ( Expresie );
```

```
for ( Expresie_de_Initializare ;  Expresie_de_Testare ; Expresie_de_Continuare )
    Instructiune
```

## 2.4 Subprogram

Un subprogram este o colecție de tipuri de date, variabile, instrucțiuni care îndeplinesc o anumită sarcină (calcule, citiri, afișări), atunci când este apelat de un program sau de un alt subprogram.

Utilizarea subprogramelor are câteva avantaje:

- reutilizarea codului – după ce am scris un subprogram îl pute apela de oricâte ori este nevoie;
- modularizarea programelor – subprogramele ne permit să împărțim problema dată în mai multe subprobleme, mai simple;
- reducerea numărului de erori care pot să apară în scrierea unui program
- depistarea cu ușurință a erorilor – fiecare subprogram va fi verificat la crearea sa, apoi verificăm modul în care apelăm subprogramele

Subprogramele pot fi de două tipuri:

- **funcții** – subprograme care determină un anumit rezultat, o anumită valoare, pornind de la anumite date de intrare. Spunem că valoarea este returnată de către funcție, iar aceasta va fi apelată ca operand într-o expresie, valoarea operandului în expresie fiind de fapt valoarea rezultatului funcției.
- **proceduri** – subprograme care se folosesc într-o instrucțiune de sine stătătoare, nu într-o expresie. Ele îndeplinesc o sarcină, au un efect și nu returnează un rezultat.

În limbajul C/C++, există doar subprograme de tip funcție. Pentru proceduri se folosește o formă particulară a funcțiilor. Sintaxa pentru declararea unei functii este:

```
tip_date nume(tip_date variabila1, tip_date variabila2....){

        Instructiuni
        return rezultat;

}

void nume(tip_date variabila1, tip_date variabila2....){

        Instructiuni

}
```

## 2.5 Listă simplu înlănţuită

O listă simplu înlănţuită este o **structură de date liniară**, în care elementele nu sunt stocate în locații de memorie contigue. Elementele dintr-o listă sunt legate folosind pointeri.

O listă liniară simplu înlănțuită conține elemente (**noduri**) a căror valori constau din două părți: informația utilă și informația de legătură. Informația utilă reprezintă informația propriu-zisă memorată în elementul liste (numere, șiruri de caractere, etc.), iar informația de legătură precizează adresa următorului element al listei. În C/C++ putem folosi următorul tip de date pentru a memora elementele unei liste liniare simplu înlănțuite alocate dinamic:

```
struct nod{
  int info;
  nod *urm;
};
```

Avantajele listei simplu înlănţuite fata de un vector sunt eficienţa spatiala a memoriei si inserarea/ştergerea rapida a elementelor însa accesarea elementelor este lentă.

## 2.6 Şirul de caractere

În C++ există mai multe modalități de a reprezenta șirurile de caractere, cel mai comun fiind reprezentarea şirului ca un vector de caractere.

Aceste șiruri se mai numesc **null-terminated byte string** (NTBS). În reprezentarea internă, după ultimul caracter (byte, octet) valid din șir se află caracterul '\0' – caracterul cu codul ASCII 0, numit și **caracter nul**.

```
char s[20];
```

Pentru prelucrearea eficienta a şirurilor de caractere se utilizează funcţii prestabilite in header-ul **cstring.**

# Capitolul III: Programul

## 3.1 Abordarea problemei

Deoarece poker-ul în 3 cărţi este un joc dinamic ce necesita stocarea a mai multor variabile(mâinile jucătorului si dealerului, ante-ul etc.) trebuie sa folosim o structura de dată care să utilizeze eficient memoria dar care totuşi să ne permita sa executam instrucţiuni rapid. Aşadar, vom folosi liste simplu înlănţuite, descrise anterior.

Rezolvarea problemei va consta în crearea măinilor jucătorului si dealerului, compararea lor si calculul câştigurilor. Pe lânga elementele de bază a limbajului C++, vom utiliza funcţii specializate ce ne permite uşurarea rezolvării.

## 3.2 Implementarea soluţiei

În total, pentru implementarea poker-ului în 3 cărţi in C++ am avut nevoie de funcţii specializate din cadrul a 5 biblioteci C++.

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <cstring>
#include <iomanip>

using namespace std;
```

- **iostream** cuprinde clasele şablon pentru citirea(**cin**) şi afişarea(**cout)** a datelor citite de la tastatura
- **cstdlib** reprezinta varianta C++ a bibliotecii stdlib.h din C ce cuprinde o multitudine de funcţii specializate dintre care vom folosi **rand** si **srand** pentru generarea mâinilor
- din **ctime** utilizăm funcţia **time** care furnizează timpul actual al calculatorului in format UNIX
- biblioteca **cstring** cuprinde funcţii pentru prelucrarea şirurilor de caractere, îndeosebi **strcmp**, ce compara doua şiruri
- **iomanip** o utilizăm pentru o afişare mai estetica a datelor
- **using namespace std** arată că folosim biblioteca mare **STL**(Standard Template Library).

Înainte de a crea mâinile din poker, trebui sa definim calculatorului termenul de "carte de joc". Pentru aceasta ne vom baza pe cele 3 atribute ale acesteia:

1. Tipul cărţii : Romb, Treflă, Inimă Rosie şi Neagra
2. Numărul cărţii : As , Unu, Doi,...., Rege
3. Valoarea cărţii : egala cu numărul cărţii doar asul având o valoare diferita deoarece asul bate regele in poker

Ca să reţinem datele compact vom utiliza un struct intitulat sugestiv **carti**. Astfel, mâinile vor fi reprezentate ca o lista înlănţuită în care informatia este de forma carti.

```
struct carti
{
   int tip, nr, val;
};
struct nod
{
    carti carte;
    nod *leg;
};
```

Pentru a uşura calculul, vom utiliza o reprezentare numerică pentru numărul cărţii ce o transformam în reprezentarea cu litere la afişare folosind doi vectori de şiruri de caractere.

```
char tip_name[4][15]={"Inima Rosie ", "Inima Neagra", "Romb        ", "Trefla      "};
char nr_name[15][7]={" ", "As","Doi","Trei","Patru","Cinci","Sase","Sapte","Opt",
                     "Noua","Zece"," ","Juvete","Regina","Rege"};
```

Spaţiile sunt necesare pentru afişarea în linie a mâinii.

Pentru rezolvarea problemei o vom sub-diviza in probleme mai mici ce le vom rezolva cu ajutorul subprogramelor.

Prima sarcină este de a creea mâna jucătorului şi a dealerului. Pentru aceasta trebuie să alegem aleatoriu 3 cărţi pentru jucator , respectiv 3 cărţi pentru dealer.

Presupunem că toate cărţile de acelasi tip sunt aranjate consecutiv în pachet în această ordine: Inimă Roşie, Inimă Neagră , Romb şi Trefla, şi că fiecare dintre acestea sunt ordonate crescător de la As la Rege.

Alegând un număr din intervalul 0-51(pachetul începe numărătoarea de la poziţia 0) cartea care se află pe acea poziţie în pachetul nostru o putem încadra într-unul dintre cele 4 tipuri prin împărţirea la 13; cătul reprezentând tipul cărţii. Restul împărţirii va reprezenta numărul cărţii care reprezintă de asemenea şi valoarea cărţii cu excepţia asului.

Pentru ca să nu extragem din pachet aceeaşi carte vom declara un vector de 52 de elemente cu fiecare element 0 care va deveni 1 dupa ce vom realiza procesul de extragere.

Acest proces de repete de 3 ori.

```
nod *creare_pachet()
{
    int v[52], nr_carti_ramase=3, x;
    nod *q, *p=NULL;
    for(int i=0;i<52;i++)
    {
        v[i]=0;
    }
    while(nr_carti_ramase)
    {
        x=rand()%52;
        if(v[x]==0)
        {
            v[x]=1;
            nr_carti_ramase--;
            q=new nod;
            q->carte.tip=x/13;
            q->carte.nr=x%13<10?x%13+1:x%13+2;
            q->carte.val=q->carte.nr==1?15:q->carte.nr;
            q->leg=p;
            p=q;
        }
    }
    return p;
}
```

Functia **rand** returneză un numar intreg aleatoriu, iar expresia **rand()%52** returneaza restul împărţirii unui numar aleatoriu la 52, cu alte cuvinte un număr aletoriu între 0 şi 52.

Următoarele două subprograme realizează sortarea mâinii prin **metoda bulelor** asfel in mod crescător dupa valoare, pentru o afişare mai intuitiva pentru utilizator.

Prima realizează interschimbarea a două noduri iar cea de a doua realizează sortarea propriu-zisă.

```
void schimb(nod *a,nod *b)
{
    carti aux=a->carte;
    a->carte=b->carte;
    b->carte=aux;
}
void sortare(nod *p)
{
    int ok, i;
    nod *q,*u=NULL;

    if (p==NULL)
        return;
    do
    {
        ok=0;
        q=p;

        while (q->leg!=u)
        {
            if (q->carte.val<q->leg->carte.val)
            {
                schimb(q,q->leg);
                ok=1;
            }
            q=q->leg;
        }
        u=q;
    }
    while (ok);
}
```

Pentru calculul pariului **Pair Plus** si a bonusului **Ante** vom folosi un subprogram pe analizează mâna şi o încadrează într-unul din cele 7 cazuri posibile, apoi afişază utilizatorului pe ecran căştigul.

Similar, funcţia **combdilar** analizează mâna dealerului pentru combinaţii speciale, diferenţa constă in faptul că nu calculează suplimentar căştigul si nu-l afişază, returnând o valoare sugestiva pentru fiecare combinaţie.

```
void afizarecomb(nod *p,int ante,int pairplus,int &val)
{
        nod *u;
        u=p;
        val=-1;
        if(u->carte.val==u->leg->carte.val || u->leg->carte.val==u->leg->leg->carte.val ||u->carte.val==u->leg->leg->carte.val)
        {
            cout<<"Aveti o pereche!"<<endl;
            cout<<"Nu ati primit bonus pentru ca nu aveti cel putin un flush sau mai bun."<<endl;
            val=2;

        }
        else if((u->carte.nr+1==u->leg->carte.nr && u->carte.nr+2==u->leg->carte.nr) && (u->carte.tip==u->leg->carte.tip &&
                    u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip))
        {
            cout<<"Aveti o triple scoala!"<<endl;
            ante=5*ante;
            pairplus=40*pairplus;
            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti o triple scoala."<<endl;
            val=6;
        }
        else if(u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip)
        {
            cout<<"Aveti un flush!"<<endl;
            pairplus*=4;
            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti un flush."<<endl;
            val=3;
        }
        else if(u->carte.nr 1==u->leg->carte.nr && u->carte.nr 2==u->leg->carte.nr)
        {
            cout<<"Aveti o tripla!"<<endl;
            pairplus*=5;
            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti o tripla."<<endl;
            val=4;

        }
          else if(u->carte.val==u->leg->carte.val && u->carte.val==u->leg->leg->carte.val)
          {
              cout<<"Aveti 3 de acelasi tip!"<<endl;
              ante=4*ante;
              pairplus*=30;
              cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti 3 carti de acelasi tip."<<endl;
              val=5;
          }
          else if(u->carte.val>=11 || u->leg->carte.val>=11 || u->leg->leg->carte.val>=11)
          {
              cout<<"Aveti carte mare!"<<endl;
              cout<<"Nu ati primit bonus pentru ca nu aveti cel putin flush sau mai bun."<<endl;
              val=1;
          }
          else if(val==-1)
          {
              cout<<"Nu ati primit bonus pentru ca nu aveti cel putin flush sau mai bun."<<endl;
              cout<<endl;
          }


}
```

```
int combdilar(nod *p)
{

        nod *u;
        u=p;
        if(u->carte.val<11 && u->leg->carte.val<11 && u->leg->leg->carte.val<11)
        {
            return 0;
        }
        else if(u->carte.val==u->leg->carte.val || u->leg->carte.val==u->leg->leg->carte.val ||u->carte.val==u->leg->leg->carte.val)
        {

            return 2;

        }
        else if(((u->carte.nr+1==u->leg->carte.nr && u->carte.nr+2==u->leg->carte.nr)) && (u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip))
        {

            return 6;
        }
        else if(u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip)
        {

            return 3;
        }
        else if(u->carte.nr-1==u->leg->carte.nr && u->carte.nr-2==u->leg->carte.nr)
        {

            return 4;

        }
        else if(u->carte.val==u->leg->carte.val && u->carte.val==u->leg->leg->carte.val)
        {

            return 5;
        }
        else if(u->carte.val>=11 || u->leg->carte.val>=11 || u->leg->leg->carte.val>=11)
        {
            return 1;

        }
        else return -1;
}
```

Funcţia **maxim** returnează valoarea cea mai mare a mâinii.

```
int maxim(nod *p)
{
    nod *q;
    int maxim=-1;
    q=p;
    while(q)
    {
        if(q->carte.val>maxim)
        {
            maxim=q->carte.val;
        }
        q=q->leg;
    }
    return maxim;

}
```

Funcţia **afisare** transpune din codul numeric, numele cărţii şi îl afişază pe ecran.

```
void afisare(nod *p)
{
    nod *q=p;
    cout<<endl;
    while(q)
    {
        cout<<setw(22)<<nr_name[q->carte.nr]<<" de "<<tip_name[q->carte.tip]<<endl;
        q=q->leg;
    }
    cout<<endl;
}
```

Funcţia **stergerelista** şterge lista înlănţuită prin eliberarea din memorie a fiecărui nod prin funcţia **free** din cstdlib.

```
void stergerelista(nod *&p)
{
    nod *q=p,*u;
    while(q!=NULL)
    {
      u=q->leg;
      free(q);
      q=u;
    }
    p=NULL;

}
```

Funcţia **joc** cuprinde jocul de poker în sine ce realizează următoarele:

- atribuie jucătorului si dealerului 3 cărţi fiecare
- apelează funcţiile **afisare** şi **afisarecomb**
- apelează funcţia **sortare** atăt pentru jucător cât şi pentru dealer
- calculează câştigătorul
- afişază căştigul total

Dat fiind faptul că functia **rand** nu poate calcula mai multe numere aleatorii in aceaşi rulare, trebuie să folosim funcţia **srand** prin care se pot genera numere aleatorii,însă este necesar un diferit input de fiecare dată când este apelata funcţia rand aşadar folosim funcţia time pentru acest lucru.

Funcţia **setw** are rolul de a pune caracterul " " de n ori, astfel afişând datele estetic pe ecran.

Funcţia **system("pause>nul)"** opreşte rularea programului pâna când utilizatorul apasă tasta **ENTER**, fără să afişeze pe ecran.

```
void joc()
{
     nod *jucator=NULL,*dilar=NULL;
     int ante,pariu,pairplus,x;
     srand(time(NULL));
     cout<<setw(70)<<"******************** Poker in 3 Carti ********************"<<endl;
     cout<<"\nIntroduceti ante-ul:";
     cin>>ante;
     cout<<"Introduceti bonusul pentru perechi+:";
     cin>>pairplus;
     cout<<endl;
     jucator=creare_pachet();
     sortare(jucator);
     cout<<"Mana dumneavoastra:"<<endl;
     afisare(jucator);
     afisarecomb(jucator,ante,pairplus,x);
     cout<<"\nApasati ENTER pentru a continua..."<<endl;
     system("pause>nul");
     cout<<endl;
     dilar=creare_pachet();
     sortare(dilar);
```

```
    if(!combdilar(dilar))
    {
        cout<<"Mana dilarului:"<<endl;
        afisare(dilar);
        cout<<"Ati castigat "<<ante+pairplus<<" euro deoarece dilaru nu are o carte mai mare de regina."<<end
        stergerelista(jucator);
        stergerelista(dilar);
        return;
    }
    else
    {
       cout<<"\nIntroduceti pariul:";
       cin>>pariu;
       cout<<"\nApasati ENTER pentru a continua..."<<endl;
       system("pause>nul");
       cout<<endl;
       int m=combdilar(dilar);
       if(x<m)
       {
           cout<<"Mana dilarului:"<<endl;
           afisare(dilar);
           cout<<"Ati pierdut "<<ante+pairplus+pariu<<" euro."<<endl;
           stergerelista(jucator);
           stergerelista(dilar);
```

```
           return;
        }
        else if(x>m)
        {
            cout<<"Mana dilarului:"<<endl;
            afisare(dilar);
            cout<<"Ati castigat "<<ante+pairplus+pariu<<" euro."<<endl;
            stergerelista(jucator);
            stergerelista(dilar);
            return;
        }
        else if(x==m)
        {
            int a,b;
            a=maxim(jucator);
            b=maxim(dilar);
            if(a>b)
            {
                cout<<"Mana dilarului:"<<endl;
                afisare(dilar);
                cout<<"Ati castigat "<<ante+pairplus+pariu<<" euro."<<endl;
                stergerelista(jucator);
                stergerelista(dilar);
                return;
```

```
                return;
             }
            else if(a<b)
             {
                cout<<"Mana dilarului:"<<endl;
                afisare(dilar);
                cout<<"Ati pierdut "<<ante+pairplus+pariu<<" euro."<<endl;
                stergerelista(jucator);
                stergerelista(dilar);
                return;
             }
             else if(a==b)
             {
                cout<<"Mana dilarului:"<<endl;
                afisare(dilar);
                cout<<"Egalitate.";
                cout<<"Ati primit inapoi "<<ante+pairplus+pariu<<" euro."<<endl;
                stergerelista(jucator);
                stergerelista(dilar);
                return;
             }
         }

    }

}
```

În functia **main,** apelăm **joc,** şi îl întrebăm pe utilizator dacă doreşte să se joace din nou. Dacă **nu**, acesta se opreşte iar dacă **da** reia din nou joc.

Functia **strcmp,** compara 2 şiruri de caractere , returnând 1 daca acestea sunt egale.

Funcţia **system("cls")** şterge tot ecranul de caractere.

```
int main()
{
    char y[3];
    joc();
    cout<<"\nDoriti sa reincercati?"<<endl;
    cin>>y;
    if(!strcmp(y,"NU") ||!strcmp(y,"Nu") ||!strcmp(y,"nU") || !strcmp(y,"nu"))
    {
       return 0;
    }
    while(!strcmp(y,"Da") ||!strcmp(y,"DA") ||!strcmp(y,"dA") || !strcmp(y,"da"))
    {
            system("cls");
            joc();
            cout<<"\nDoriti sa reincercati?"<<endl;
            cin>>y;
    }

    return 0;

}
```

## 3.3 Codul C++

```
#include <iostream>

#include <cstdlib>

#include <ctime>

#include <cstring>

#include <iomanip>

using namespace std;

char tip_name[4][15]={"Inima Rosie ", "Inima Neagra", "Romb       ", "Trefla     "};

char nr_name[15][7]={" ", "As","Doi","Trei","Patru","Cinci","Sase","Sapte","Opt",
                     "Noua","Zece"," ","Juvete","Regina","Rege"};
```

```
struct carti
{
  int tip, nr, val;
};
struct nod
{
   carti carte;
   nod *leg;
};
nod *creare_pachet()
{
   int v[52], nr_carti_ramase=3, x;
   nod *q, *p=NULL;
   for(int i=0;i<52;i++)
   {
      v[i]=0;
   }
   while(nr_carti_ramase)
   {
      x=rand()%52;
      if(v[x]==0)
      {
         v[x]=1;
         nr_carti_ramase--;
```

```
            q=new nod;
            q->carte.tip=x/13;
            q->carte.nr=x%13<10?x%13+1:x%13+2;
            q->carte.val=q->carte.nr==1?15:q->carte.nr;
            q->leg=p;
            p=q;
        }
    }
    return p;
}
void schimb(nod *a,nod *b)
{
    carti aux=a->carte;
    a->carte=b->carte;
    b->carte=aux;
}
void sortare(nod *p)
{
    int ok, i;
    nod *q,*u=NULL;

    if (p==NULL)
        return;
```

```
 do
   {
      ok=0;
      q=p;
      while (q->leg!=u)
      {
         if (q->carte.val<q->leg->carte.val)
         {
            schimb(q,q->leg);
            ok=1;
         }
         q=q->leg;
      }
      u=q;
   }
   while (ok);
}
void afisarecomb(nod *p,int &ante,int &pairplus,int &val)
{
      nod *u;
      u=p;
      val=-1;
      if(u->carte.val==u->leg->carte.val || u->leg->carte.val==u->leg->leg->carte.val ||u->carte.val==u->leg->leg->carte.val)
```

```
        {
            cout<<"Aveti o pereche!"<<endl;
            cout<<"Nu ati primit bonus pentru ca nu aveti cel putin un flush sau mai bun."<<endl;
            val=2;
        }
        else if((u->carte.nr+1==u->leg->carte.nr && u->carte.nr+2==u->leg->carte.nr) && (u->carte.tip==u->leg->carte.tip &&
                        u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip))
        {
            cout<<"Aveti o tripla royala!"<<endl;
            ante=5*ante;
            pairplus=40*pairplus;
            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti o tripla royala."<<endl;
            val=6;
        }
        else if(u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip && u->carte.tip==u->leg->leg->carte.tip)
        {
            cout<<"Aveti un flush!"<<endl;
            pairplus*=4;
            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti un flush."<<endl;
            val=3;
        }
```

```
        else if(u->carte.nr-1==u->leg->carte.nr && u->carte.nr-2==u->leg->carte.nr)

        {

            cout<<"Aveti o tripla!"<<endl;

            pairplus*=6;

            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti o
tripla."<<endl;

            val=4;

        }

        else if(u->carte.val==u->leg->carte.val && u->carte.val==u->leg->leg->carte.val)

        {

            cout<<"Aveti 3 de acelasi tip!"<<endl;

            ante=4*ante;

            pairplus*=30;

            cout<<"Ati primit un bonus de "<<pairplus<<" de euro pentru ca aveti 3 carti de
acelasi tip."<<endl;

            val=5;

        }

        else if(u->carte.val>=11 || u->leg->carte.val>=11 || u->leg->leg->carte.val>=11)

        {

            cout<<"Aveti carte mare!"<<endl;

            cout<<"Nu ati primit bonus pentru ca nu aveti cel putin flush sau mai bun."<<endl;

            val=1;

        }

        else if(val==-1)

        {
```

```
            cout<<"Nu ati primit bonus pentru ca nu aveti cel putin flush sau mai bun."<<endl;
            cout<<endl;
        }
}
int combdilar(nod *p)
{
        nod *u;
        u=p;
        if(u->carte.val<11 && u->leg->carte.val<11 && u->leg->leg->carte.val<11)
        {
            return 0;
        }
        else if(u->carte.val==u->leg->carte.val || u->leg->carte.val==u->leg->leg->carte.val
||u->carte.val==u->leg->leg->carte.val)
        {
            return 2;
        }
        else if(((u->carte.nr+1==u->leg->carte.nr && u->carte.nr+2==u->leg->carte.nr)) &&
(u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip && u-
>carte.tip==u->leg->leg->carte.tip))
        {
            return 6;
        }
        else if(u->carte.tip==u->leg->carte.tip && u->leg->carte.tip==u->leg->leg->carte.tip
&& u->carte.tip==u->leg->leg->carte.tip)
        {
```

```
            return 3;
        }
        else if(u->carte.nr-1==u->leg->carte.nr && u->carte.nr-2==u->leg->carte.nr)
        {
            return 4;
        }
        else if(u->carte.val==u->leg->carte.val && u->carte.val==u->leg->leg->carte.val)
        {
            return 5;
        }
        else if(u->carte.val>=11 || u->leg->carte.val>=11 || u->leg->leg->carte.val>=11)
        {
            return 1;
        }
        else return -1;
}
int maxim(nod *p)
{
    nod *q;
    int maxim=-1;
    q=p;
    while(q)
    {
       if(q->carte.val>maxim)
```

```
      {
         maxim=q->carte.val;
      }
      q=q->leg;
   }
   return maxim;
}
void afisare(nod *p)
{
   nod *q=p;
   cout<<endl;
   while(q)
   {
      cout<<setw(22)<<nr_name[q->carte.nr]<<" de "<<tip_name[q->carte.tip]<<endl;
      q=q->leg;
   }
   cout<<endl;
}
void stergerelista(nod *&p)
{
   nod *q=p,*u;
   while(q!=NULL)
   {
     u=q->leg;
```

```
      free(q);
      q=u;
    }
   p=NULL;
}
void joc()
{
    nod *jucator=NULL,*dilar=NULL;
    int ante,pariu,pairplus,x;
    srand(time(NULL));
    cout<<setw(70)<<"********************** Poker in 3 Carti
**********************"<<endl;
    cout<<"\nIntroduceti ante-ul:";
    cin>>ante;
    cout<<"Introduceti bonusul pentru perechi+:";
    cin>>pairplus;
    cout<<endl;
    jucator=creare_pachet();
    sortare(jucator);
    cout<<"Mana dumneavoastra:"<<endl;
    afisare(jucator);
    afisarecomb(jucator,ante,pairplus,x);
    cout<<"\nApasati ENTER pentru a continua..."<<endl;
    system("pause>nul");
```

```
    cout<<endl;
    dilar=creare_pachet();
    sortare(dilar);
    if(!combdilar(dilar))
    {
        cout<<"Mana dilarului:"<<endl;
        afisare(dilar);
        cout<<"Ati castigat "<<ante+pairplus<<" euro deoarece dilaru nu are o carte mai
mare de regina."<<endl;
        stergerelista(jucator);
        stergerelista(dilar);
        return;
    }
    else
    {
      cout<<"\nIntroduceti pariul:";
      cin>>pariu;
      cout<<"\nApasati ENTER pentru a continua..."<<endl;
      system("pause>nul");
      cout<<endl;
      int m=combdilar(dilar);
      if(x<m)
      {
          cout<<"Mana dilarului:"<<endl;
```

```
    afisare(dilar);
    cout<<"Ati pierdut "<<ante+pairplus+pariu<<" euro."<<endl;
    stergerelista(jucator);
    stergerelista(dilar);
    return;
}
else if(x>m)
{
    cout<<"Mana dilarului:"<<endl;
    afisare(dilar);
    cout<<"Ati castigat "<<ante+pairplus+pariu<<" euro."<<endl;
    stergerelista(jucator);
    stergerelista(dilar);
    return;
}
else if(x==m)
{
    int a,b;
    a=maxim(jucator);
    b=maxim(dilar);
    if(a>b)
    {
        cout<<"Mana dilarului:"<<endl;
        afisare(dilar);
```

```
    cout<<"Ati castigat "<<ante+pairplus+pariu<<" euro."<<endl;
    stergerelista(jucator);
    stergerelista(dilar);
    return;
 }
else if(a<b)
 {
    cout<<"Mana dilarului:"<<endl;
    afisare(dilar);
    cout<<"Ati pierdut "<<ante+pairplus+pariu<<" euro."<<endl;
    stergerelista(jucator);
    stergerelista(dilar);
    return;
 }
 else if(a==b)
 {
    cout<<"Mana dilarului:"<<endl;
    afisare(dilar);
    cout<<"Egalitate.";
    cout<<"Ati primit inapoi "<<ante+pairplus+pariu<<" euro."<<endl;
    stergerelista(jucator);
    stergerelista(dilar);
    return;
 }
```

```
        }
     }
}
int main()
{
    char y[3];
    joc();
    cout<<"\nDoriti sa reincercati?"<<endl;
    cin>>y;
    if(!strcmp(y,"NU") ||!strcmp(y,"Nu") ||!strcmp(y,"nU") || !strcmp(y,"nu"))
    {
       return 0;
    }
    while(!strcmp(y,"Da") ||!strcmp(y,"DA") ||!strcmp(y,"dA") || !strcmp(y,"da"))
    {
          system("cls");
          joc();
          cout<<"\nDoriti sa reincercati?"<<endl;
          cin>>y;
    }
    return 0;
}
```

# Bibliografie


- https://en.wikipedia.org/wiki/Three_Card_Poker
- https://ro.wikipedia.org/wiki/C%2B%2B
- https://www.pbinfo.ro/articole/5547/informatica-clasa-a-ix-a
- https://www.pbinfo.ro/articole/5548/informatica-clasa-a-x-a


# Programe folosite

- *Code::Blocks 13.12* https://www.codeblocks.org/
- *Microsoft Office Word 2007* https://www.microsoft.com/en-us/microsoft-365/previous-versions/download-office-2007