Proiect la

Metode Moderne de Calcul & Simulare

Blackjack

Echipa:

Patrascu Alexandru-Mihail

Iancu Ana

Preda Catalina

Universitatea Bucuresti

Calculatoare si Tehnologia Informatiei

Grupa 454

1. Regulile jocului blackjack

Jocul de blackjack, cunoscut şi sub numele de 21 (douăzeci şi unu), este unul dintre cele mai populare jocuri în cazinourile din întreaga lume şi, probabil, cel mai popular joc cărţi de pe planetă. Blackjack este un joc de cărţi care implică anumite strategii şi, de obicei, se joacă la mese colective cu 6 jucători. Cu toate acestea, fiecare dintre jucătorii de la masă joacă un joc individual împotriva dealerului, nu joacă împotriva celorlalţi jucători. În cazinouri online, vei juca direct împotriva dealerului, fără alţi jucători la masă. O excepţie este cazul în care optezi să aplici Sistem 21 într-un cazinou live.

Scopul jucătorului este să atingă un total al cărţilor de 21 sau să fie cât mai aproape de 21. Atât jucătorul, cât şi dealerul fac blackjack atunci când primele două cărţi sunt as şi 10 (as + cartea 10 sau as + carte cu figură). Backjack se joacă, de obicei, cu 6 pachete de cărţi simultan, iar pachetele sunt amestecate în mod automat de maşină în timpul jocului.

În prima tură, se distribuie cărţi tuturor jucătorilor. Cele două cărţi pe care le primeşti sunt mereu la vedere, cu faţa în sus, adică, şi tu şi dealerul puteţi vedea ce ai în mână. Dintre cărţile dealerului, una este la vedere, iar cealaltă este cu faţa în jos. Cartea la vedere a dealerului este cea care influenţează deciziile de joc ale jucătorilor. Astfel, în funcţie de cartea dealerului, ştii ce trebuie să faci în tura următoare.

Şi dealerul are acelaşi obiectiv ca tine: să facă 21. Cu toate acestea, valoarea cărţilor dealerului trebuie să fie întotdeauna mai mare sau egală cu 17. Aşadar, dacă primele două cărţi ale dealerului au o valoare mai mică de 17, el trebuie să mai tragă cărţi pentru a atinge cel puţin 17 şi cel mult 21. Când suma cărţilor dealerului depăşeşte 21, el face “bust” şi câştigi partida. Dacă dealerul are între 17 şi 21, câştigi doar dacă ai o mână mai bună decât dealerul. Dacă valoarea mâinii tale este egală cu cea a dealerului este remiză. Dacă mâna ta are valoare mai mică decât a dealerului, pierzi partida.

La blackjack se plăteşte 2 la 1, dar dacă faci blackjack cu primele două cărţi, câştigi 3 la 2. În cazul în care dealerul are blackjack, îi bate pe toţi jucătorii de la masă, chiar şi pe cei care

au 21. Dacă faci blackjack şi dealerul are şi el blackjack, se consideră remiză şi ţi se returnează suma pariată.

Reţine: există diferenţă între a face 21 şi a face blackjack. Deşi ambele mâini au o valoare de douăzeci şi unu, prima se realizează din mai multe cărţi care au valoarea totală de 21. Blackjack se face numai cu primele două cărţi: as şi 10 (as + cartea 10 sau as + carte cu figură).

Valoarea cărţilor la blackjack

La blackjack, toate cărţile cu figură au valoarea 10, fie că este vorba de rege, damă sau valet. Indiferent de culoarea cărţii, acesta are întotdeauna valoarea 10. Cărţile de la 1 la 10 au valoarea indicată de număr. astfel, cartea de 4 valorează 4, iar cea de 7 are valoarea 7.

Asul este singura carte din blackjack care poate valora 1 sau 11. Poţi alege valoarea asului, după cum îţi este mai avantajos în joc. Când joci în cazinouri online, software-ul de joc calculează valoarea cărţilor tale şi alege valoarea asului care este mai avantajoasă pentru tine.

Regulile de bază ale jocului blackjack

Split (separare) – când un jucător primeşte două cărţi identice, poate face split şi îşi poate separa cărţile în două mâini independente. În acest caz, jucătorul trebuie să mai pună un pariu egal cu primul şi joacă cu două mâini. Jocul continuă la fel ca înainte: jucătorul primeşte o carte suplimentară pentru fiecare mână şi poate solicita mai multe cărţi, poate abandona sau poate dubla pariul. La splitul din doi aşi, regulile sunt un pic diferite. Dacă a făcut split din doi aşi, jucătorul primeşte o singură carte suplimentară pentru fiecare mână. Dacă la un split din aşi, jucătorul face 21 cu o carte cu figură şi as, această mână nu este considerată blackjack.

Double Down (Dublare) – când un jucător primeşte 2 cărţi care au valoare totală relativ mică şi consideră că următoarea carte îi va aduce un punctaj mare, poate dubla pariul şi primeşte numai o carte suplimentară. De obicei, dublarea este permisă atunci când un jucător are în mână cărţi cu valoarea cuprinsă între 8 – 11, dar aceste valori depind de regulile şi de software-ul folosit de fiecare cazinou. Există şi cazinouri care permit dublarea pariului la valori mai mici sau mai mari.

Insurance(Asigurare) – dacă dealerul are un as la vedere, jucătorul are posibilitatea de a face o asigurare pentru pariul său. Asigurarea este echivalentă cu jumătate din suma pariată. Dacă dealerul are blackjack, jucătorul pierde pariul, dar primeşte înapoi valoarea asigurării la 2 la 1, adică, i se restituie valoarea pariului. Dacă dealerul nu are blackjack, jucătorul pierde asigurarea, dar continuă cu pariul din joc. Când jucătorul şi dealerul fac blackjack, este remiză.

Cărţile dealerului la blackjack

Dealerul trebuie să tragă întotdeauna cărţi până la o valoare totală mai mare sau egală cu 17, chiar dacă valorile mâinilor jucătorilor sunt inferioare. Dealerul nu mai trage cărţi când valoarea mâinii lui este între 17 şi 21 sau când face “bust”, adică totalul cărţilor din mână depăşeşte valoarea de 21. În acest caz, toţi jucătorii câştigă pariurile lor.

2. Scopul Aplicatiei

Scopul aplicatiei este de a simula meciuri de blackjack intre un dealer si un jucator avand in vedere setul de reguli enumarat mai sus si un obiect reprezentand pachetul de carti. Acestea sunt implementate in codul aplicatiei.

3. Instalarea Aplicatiei

Pentru a rula aplicatia este necesara instalare pachetului Java si a pachetului de dezvoltare JDK 1.8. Aplicatia poate fi rulata intr-un IDE la alegere (InteIJ, NetBeans, Eclipse).

Link-uri pentru descarcareapachetelor necesare:

https://www.java.com/en/download/

https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

https://www.jetbrains.com/idea/download/#section=windows

4. Structura Aplicatiei

Aplicatia a fost realizata utilizand limbajul Java si conceptul de Programare Orientata pe Obiecte. Clasele utilzate in aplicatie sunt:

1. Card

```java
package com.company;

public class Card
{
    int secondCount = -1;
    int count;
    String name;
    int[] counts = {11, 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11};
    String[] names = {"Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King"};

    public Card(int num) {
        this.name = names[num];
        this.count = counts[num];
    }

    public Card(int num1, int num2) {
        this.name = names[num1];
        this.count = counts[num1];
        this.secondCount = counts[num2];
    }

    public String getName() { return name; }


    public int getCount() { return count; }

    public int getSecondCount() { return secondCount; }
}
```

2. Deck

```java
package com.company;

import java.util.ArrayList;
import java.util.Collections;

public class Deck
{
    ArrayList<Card> cards;

    Deck() {
        cards = new ArrayList<>();
        //forma
        for (short a = 0; a <= 3; a++) {
            //numar
            for (short b = 0; b <= 12; b++) {
                if (b == 0) {
                    //constructor as
                    cards.add(new Card(b, 1));
                } else {
                    //constructor carte normala
                    cards.add(new Card(b));
                }
            }
        }
    }

    public void shuffle() { Collections.shuffle(cards); }

    public ArrayList<Card> getCards() { return cards; }

    public Card getCard(int i) { return cards.get(i); }
}
```

3. Game

```java
package com.company;

import java.util.ArrayList;

public class Game
{
    Deck deck = new Deck();
    int size = 52;
    int result;
    Card card;
    ArrayList<Card> hand;
    int acesInHand;


    //strategie dealer
    public int dealer(String strat) {

        acesInHand = 0;
        deck.shuffle();
        int cardNumber = 0;
        result = deck.getCard(cardNumber).getCount();
        hand = new ArrayList<>();

        switch(strat) {

            //high risk 16
            case "high risk":
            while (result < 17) {
                result = 0;
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Dealer Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);

            }
                System.out.println("Rezultat: " + result + "\n");
                break;
```

```
        //medium-high risk 14
    case "med-high risk":
        while (result < 14) {

            result = 0;
            cardNumber++;
            if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                acesInHand++;
            }
            hand.add(deck.getCards().get(cardNumber));
            System.out.println("Dealer Draw: " + deck.getCards().get(cardNumber).getName());
            result = calculateResult(hand);
        }
        System.out.println("Rezultat: " + result + "\n");
        break;

        //medium risk 12
    case "med risk":
        result = 0;
        while (result < 12) {
            result = 0;
            cardNumber++;
            if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                acesInHand++;
            }
            hand.add(deck.getCards().get(cardNumber));
            System.out.println("Dealer Draw: " + deck.getCards().get(cardNumber).getName());
            result = calculateResult(hand);
        }
        System.out.println("Rezultat: " + result + "\n");
        break;
```

```
            //low-medium risk 10
        case "low-med risk":
            while (result < 10) {
                result = 0;
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Dealer Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);
            }
            System.out.println("Rezultat: " + result + "\n");
            break;


            //low risk 8
        case "low risk":
            result = 0;
            while (result < 8) {
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Dealer Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);
            }
            System.out.println("Rezultat: " + result + "\n");
            break;


    }

    if (result == 21) {

        System.out.println("BLACKJACK");
    }
    return result;
}
```

```
        //medium risk 16
    case "med risk":
        while (result < 16) {
            cardNumber++;
            if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                acesInHand++;
            }
            hand.add(deck.getCards().get(cardNumber));
            System.out.println("Player Draw: " + deck.getCards().get(cardNumber).getName());
            result = calculateResult(hand);
        }
        System.out.println("Player Rezult: " + result + "\n");
        break;

        //low-medium risk 12
    case "low-med risk":
        while (result < 14) {
            cardNumber++;
            if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                acesInHand++;
            }
            hand.add(deck.getCards().get(cardNumber));
            System.out.println("Player Draw: " + deck.getCards().get(cardNumber).getName());
            result = calculateResult(hand);
        }
        System.out.println("Rezultat: " + result + "\n");
        break;
```

```java
//strategie player
public int player(String strat) {

    acesInHand = 0;
    deck.shuffle();
    int cardNumber = 0;
    result = deck.getCard(cardNumber).getCount();
    hand = new ArrayList<>();

    switch(strat) {

        //high risk 21
        case "high risk":
            while (result < 21) {
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Player Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);
            }
            System.out.println("Rezultat: " + result + "\n");
            break;

            //medium-high risk 18
        case "med-high risk":
            while (result < 18) {
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Player Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);
            }
            System.out.println("Rezultat: " + result + "\n");
            break;
```

```
            //low risk 10
        case "low risk":
            while (result < 12) {
                cardNumber++;
                if (deck.getCards().get(cardNumber).getSecondCount() > 0) {

                    acesInHand++;
                }
                hand.add(deck.getCards().get(cardNumber));
                System.out.println("Player Draw: " + deck.getCards().get(cardNumber).getName());
                result = calculateResult(hand);
            }
            System.out.println("Rezultat: " + result + "\n");
            break;


    }

    if (result == 21) {

        System.out.println("BLACKJACK");
    }
    return result;
}

public int calculateResult(ArrayList<Card> cards) {
    int sum = 0;
    for (int j = acesInHand; j >= 0; j--) {
        sum = 0;
        for (int i = 0; i < cards.size(); i++) {
            int smallAcesInHand = acesInHand;
            if (cards.get(i).getSecondCount() > 0 && smallAcesInHand > 0) {
                sum = sum + cards.get(i).getSecondCount();
                smallAcesInHand--;
            } else {
                sum = sum + cards.get(i).getCount();
            }
        }
        if (sum == 21) {
            return sum;
        }
    }
    return sum;
}
```

4. Blackjack (Main)

```java
package com.company;

import java.util.Scanner;

public class Blackjack {
    public static void main(String[] args) {
        int dealerwins = 0;
        int playerwins = 0;
        int drawresult = 0;


            playerwins = 0;
            dealerwins = 0;
            Game player = new Game();
            Game dealer = new Game();

            Scanner in = new Scanner(System.in);
            System.out.println("Legend for Player: \n" +
                    "low risk means he wil stop after reaching a number greater than 12\n" +
                    "low-med riskmeans he wil stop after reaching a number greater than 14\n" +
                    "med risk means he wil stop after reaching a number greater than 16\n" +
                    "med-high risk means he wil stop after reaching a number greater than 18\n" +
                    "high risk means he wil aim for 21\n");

            System.out.println("Insert the number of deals");
            String counts = in.nextLine();
            System.out.println("Insert Player risk (low risk, low-med risk, med risk, med-high risk, high risk)");
            String riskpayer = in.nextLine();


            for (int i = 0; i <= Integer.parseInt(counts) - 1; i++) {

                //initializare player si dealer
                player.player(riskpayer);
                dealer.dealer( strat: "high risk");


                if (player.result > 21) {
                    dealerwins++;
                    System.out.println("!!!!!!Dealer Wins!!!!!!\n");
                } else if (player.result == dealer.result) {
                    drawresult++;
                    System.out.println("??????No winner, Next Deal??????\n");
```

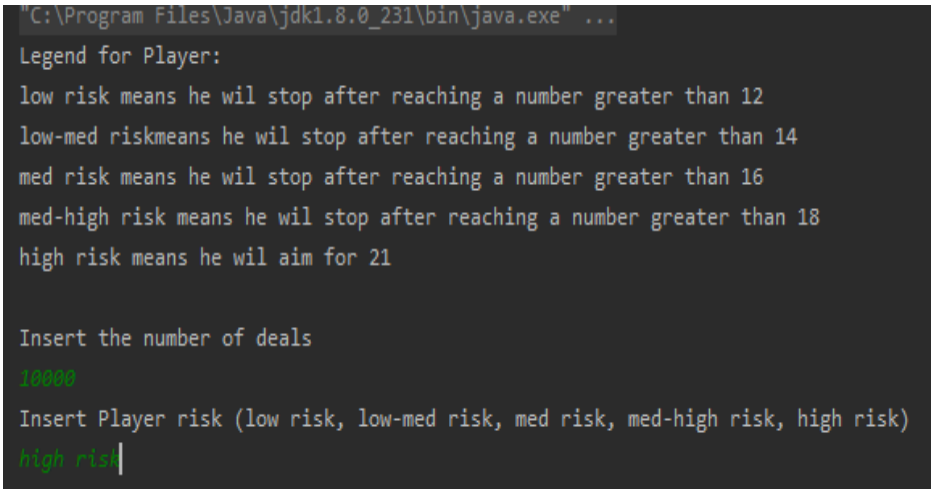
```
        else if ((dealer.result == 21 && player.result < 21)) {
            dealerwins++;
            System.out.println("!!!!!!Dealer Wins!!!!!!\n");
        } else if ((player.result == 21 && dealer.result < 21) || (player.result == 21 && dealer.result > 21)) {
            playerwins++;
            System.out.println("!!!!!!Player Wins!!!!!!\n");
        }
        //verifica daca au sub 21
        else if (dealer.result < 21) {
            if ((dealer.result > player.result)) {

                //sub 21 , dar Dealerul are mai mult
                dealerwins++;
                System.out.println("!!!!!!Dealer Wins!!!!!!\n");

            } else if ((dealer.result < player.result)) {
                //sub 21 , dar Playerul are mai mult
                playerwins++;
                System.out.println("!!!!!!Player Wins!!!!!!\n");

            }

        } else if (dealer.result > 21) {
            //Playerul sub 21, dar Dealerul a trecut peste 21
            playerwins++;
            System.out.println("!!!!!!Player Wins!!!!!!\n");

        } else if ((player.result == 21)) {
            //situatia de blackjack egal sau ambele valori peste 21
            drawresult++;
            System.out.println("??????No winner, Next Deal??????\n");
        }

    }

    float playerpercent = ((float) (playerwins) / (float) (Integer.parseInt(counts))) * 100;
    float dealerpercent = ((float) (dealerwins) / (float) (Integer.parseInt(counts))) * 100;
    float kda = ((float) (playerwins) / (float) (dealerwins));

    System.out.println("Player Total wins: " + playerwins);
    System.out.println("Dealer Total wins: " + dealerwins);
    System.out.println("No winner results: " + drawresult);
```

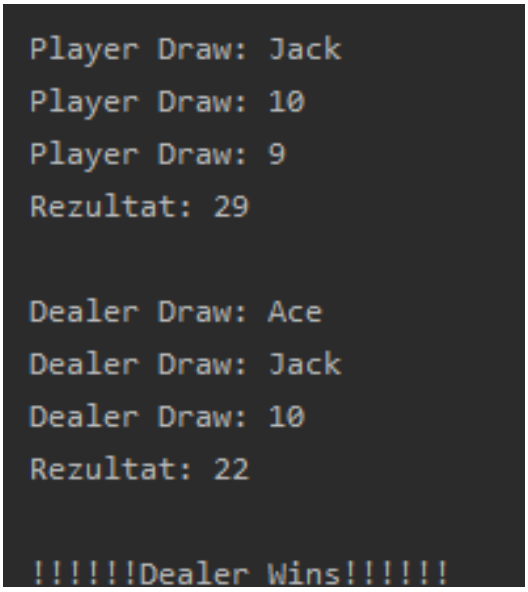
```
        System.out.println("Player's win percent in " + riskpayer + " is : " + playerpercent + "%");
        System.out.println("Dealers's win percent in " + riskpayer + " is : " + dealerpercent + "%");
        System.out.println("Players win/lose ratio in " + riskpayer + " is : " + kda);


    }

}
```

5. Rezultatele simularii

Utilizatorul introduce numarul de meciuri dorite si alege strategia de joc. Dupa simularea jocului de Blackjack, aplicatia furnizeaza un set de rezultate.

Strategiile de joc sunt: high risk (jucatorul incearca sa ajunga la suma de 21 pentru a face Blackjack), med-high risk (jucatorul o sa ceara carti pana va ajunge la suma minima de 18), med risk (jucatorul o sa ceara carti pana va ajunge la suma minima 16), med-low risk (jucatorul o sa ceara carti pana va ajunge la suma minima de 14) si low risk (jucatorul o sa ceara carti pana va ajunge la suma minima de 12).





1. Rezultatul a 10.000 de jocuri in cazul high risk

```
Player Total wins: 1158
Dealer Total wins: 8738
No winner results: 104
Player's win percent in high risk is : 11.58%
Dealers's win percent in high risk is : 87.38%
Players win/lose ratio in high risk is : 0.13252461
```

2. Rezultatul a 10.000 de jocuri in cazul med-high risk

```
Player Total wins: 3805
Dealer Total wins: 5391
No winner results: 804
Player's win percent in med-high risk is : 38.05%
Dealers's win percent in med-high risk is : 53.91%
Players win/lose ratio in med-high risk is : 0.70580596
```

3. Rezultatul a 10.000 de jocuri in cazul med risk

```
Player Total wins: 4094
Dealer Total wins: 5032
No winner results: 874
Player's win percent in med risk is : 40.94%
Dealers's win percent in med risk is : 50.32%
Players win/lose ratio in med risk is : 0.81359303
```

4. Rezultatul a 10.000 de jocuri in cazul low-med risk

```
Player Total wins: 4248
Dealer Total wins: 5010
No winner results: 742
Player's win percent in low-med risk is : 42.48%
Dealers's win percent in low-med risk is : 50.1%
Players win/lose ratio in low-med risk is : 0.8479042
```

5. Rezultatul a 10.000 de jocuri in cazul low risk

```
Player Total wins: 4231
Dealer Total wins: 5151
No winner results: 618
Player's win percent in low risk is : 42.309998%
Dealers's win percent in low risk is : 51.510002%
Players win/lose ratio in low risk is : 0.8213939
```

6. Rezultatele a N simulari

Studiul de caz a fost realizat pentru a analiza diferenta dintre executarea aplicatiei de N ori a unui set de M iteratii, fata de executarea aplicatiei o singura data pentru M*N iteratii.

Observatie: In urma analizei, putem aprecia ca rezultatele celer doua cazuri sunt identice. Executia diferita a iteratiilor nu modifica rezultatul aplicatiei.

1. Rezultatul obtinuit dupa 1000 de iteratii a 10 jocuri

```
Player Total wins: 1
Dealer Total wins: 9
No winner results: 95
Player's win percent in high risk is : 10.0%
Dealers's win percent in high risk is : 90.0%
Players win/lose ratio in high risk is : 0.11111111
Player won 1260 in 1000 iterations of 10 games

Process finished with exit code 0
```

2. Rezultatul obtinuit dupa 1000 de iteratii a 1000 jocuri

```
Player Total wins: 102
Dealer Total wins: 885
No winner results: 10135
Player's win percent in high risk is : 10.2%
Dealers's win percent in high risk is : 88.5%
Players win/lose ratio in high risk is : 0.11525424
Player won 116337 in 1000 iterations of 1000 games

Process finished with exit code 0
```

7. Bibliografie

http://www.system21.ro/regulile-jocului-blackjack/

https://www.royalcaribbean.com/cruise-activities/blackjack

https://www.java.com/en/download/

https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

https://www.jetbrains.com/idea/download/#section=windows