Ministerul Educației și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Ingineria Software și Automatică

**Proiect de curs**

*Disciplina: Implementarea algoritmilor folosind programarea structurată*

**Tema: Jocul Ping-Pong**

Au efectuat: Panta Daniel  
 Coropat Alexandru  
 Cazac Alexandru  
 gr. TI-231, TI-232

A verificat: lector. univ. Bagrin V.

Chișinău – 2024

Cuprins:

[INTRODUCERE 2](#_Toc183804610)

[1 ANALIZA DOMENIULUI 2](#_Toc183804611)

1.1 Structura generala..........................................................................................................................................4

1.2 Detalierea functionalitatilor...........................................................................................................................4

2. Clasa joc Pong.................................................................................................................................................8

2.2 Gestionarea Grafica.....................................................................................................................................10

2.3 Logica jocului..............................................................................................................................................11

2.4 Gestionarea tastaturii...................................................................................................................................11

[3. REALIZAREA SISTEMULUI 12](#_Toc183804612)

3.1 Definirea Dimensiunilor și Constantelor………………………………………………………………….12

3.2 Variabile pentru Starea Jocului……………………………………………………………………………13

3.3 Metoda paint Component…………………………………………………………………………………14

3.4Logica Actualizării Jocului………………………………………………………………………………..15

[Interfata Programului: 22](#_Toc183804613)

[BIBLIOGRAFIE 23](#_Toc183804615)

# INTRODUCERE

Jocul Pong, cunoscut și sub denumirea de Ping Pong, este unul dintre cele mai emblematice jocuri din istoria gaming-ului. Lansat inițial în 1972 de compania Atari, Pong a marcat începutul unei ere a jocurilor video, fiind considerat primul joc de succes comercial. Simplitatea mecanicii sale și accesibilitatea au transformat Pong într-un fenomen global, punând bazele unei industrii care astăzi valorează miliarde de dolari.

Pong este inspirat din tenisul de masă (ping pong) și constă în controlarea unei palete de către fiecare jucător, cu scopul de a returna mingea și de a înscrie puncte atunci când adversarul nu reușește să blocheze mingea. Această dinamică simplă ascunde însă o serie de elemente complexe: coordonare, anticipare, viteză de reacție și strategie.

Importanța jocului Pong depășește însă simplul divertisment. La vremea lansării sale, Pong a demonstrat potențialul calculatoarelor de a crea experiențe interactive pentru utilizatori, deschizând calea către explorarea interfețelor grafice și a interacțiunilor umane cu dispozitivele electronice. În zilele noastre, Pong continuă să fie folosit ca exemplu didactic pentru începători în programare, datorită mecanicilor sale clare și a codului simplu, dar eficient.

În proiectul de față, "Joc Pong", s-a dorit recrearea acestui clasic într-un mediu modern de programare, utilizând limbajul Java și biblioteca Swing. Scopul principal al proiectului este de a demonstra abilități practice în programarea interfețelor grafice, gestionarea evenimentelor și implementarea logicii unui joc. Prin această implementare, s-a încercat păstrarea spiritului original al jocului, combinat cu o interfață prietenoasă și funcționalități esențiale, cum ar fi afișarea scorului și detectarea coliziunilor.

Mai mult, proiectul Pong este relevant nu doar ca un exercițiu tehnic, ci și ca un omagiu adus unei ere de început a industriei IT. Analiza acestui joc simplu, dar captivant, oferă o perspectivă asupra evoluției tehnologiei și a jocurilor video de la primele linii de cod scrise până la experiențele imersive din prezent.

# 1.ANALIZA DOMENIULUI

**Istoria și Evoluția Jocului Pong**

Pong, unul dintre cele mai vechi jocuri video, a fost lansat pentru prima dată de Atari în 1972 și a fost dezvoltat de inginerul Nolan Bushnell. Inspirat din tenisul de masă, Pong a fost conceput ca un joc simplu, care să demonstreze posibilitățile interactive ale calculatoarelor. Lansarea sa a avut un succes imediat, devenind un punct de referință pentru industria jocurilor video și începând o eră a divertismentului digital.

În anii '70, Pong a fost introdus sub formă de cabinete arcade, iar ulterior a devenit disponibil pe console și calculatoare personale. Datorită mecanicii sale accesibile, Pong a captat rapid atenția publicului larg, inclusiv a celor care nu erau familiarizați cu tehnologia. De-a lungul decadelor, conceptul de bază al jocului a fost adaptat în diverse variante și modernizat, dar rămâne un exemplu iconic de design minimalist.

**Importanța Jocului Pong**  
 Pong a reprezentat un moment de cotitură în istoria tehnologiei datorită mai multor aspecte:

1. **Interfața simplă și intuitivă**: Prin design-ul său minimalist, Pong a demonstrat că interacțiunile utilizator-calculator pot fi simplificate.
2. **Accesibilitatea publicului larg**: Fără necesitatea unor cunoștințe tehnice, oricine putea învăța să joace Pong în doar câteva minute.
3. **Inovația tehnologică**: Pong a fost primul joc video care a încorporat mecanici precum detecția coliziunilor și afișarea scorului în timp real.
4. **Impactul economic**: Succesul financiar al Pong a dovedit că industria jocurilor video poate fi profitabilă, atrăgând investiții masive și dezvoltarea ulterioară a altor jocuri.

**Contextul Tehnologic**

În perioada anilor '70, calculatoarele personale erau încă o raritate, iar Pong a fost printre primele aplicații care au demonstrat potențialul interacțiunii vizuale cu dispozitivele electronice. Jocul a utilizat o logică simplă de detecție a coliziunilor și o simulare bidimensională a mișcării mingii, ambele implementate cu resurse limitate din punct de vedere hardware. Aceste concepte, deși simple după standardele actuale, au stat la baza dezvoltării ulterioare a interfețelor grafice și a simulărilor fizice în software.

**Relevanța pentru Dezvoltarea Jocurilor Moderne**

Conceptul și mecanicile jocului Pong continuă să fie o sursă de inspirație pentru dezvoltatorii de jocuri. Simplitatea sa servește drept exemplu pentru crearea unor experiențe utilizator fluide, iar ideile de bază, cum ar fi controlul obiectelor și detecția coliziunilor, sunt utilizate în dezvoltarea multor alte tipuri de jocuri, inclusiv simulatoare complexe și jocuri sportive.

**Aplicabilitate Didactică**

Astăzi, Pong este folosit frecvent ca proiect introductiv în cursurile de programare și dezvoltare de jocuri. Studenții care învață să creeze Pong învață principii fundamentale, cum ar fi:

* Gestionarea evenimentelor utilizator (cum ar fi apăsările de taste).
* Programarea graficii 2D.
* Implementarea fizicii de bază (mișcarea și coliziunea).
* Optimizarea algoritmilor pentru performanță.

**Implementarea în Proiectul de Față**  
 În cadrul acestui proiect, jocul Pong a fost recreat folosind limbajul de programare Java și biblioteca Swing. Aceste tehnologii moderne oferă avantajele unei dezvoltări rapide și a unei interfețe grafice ușor de implementat. Funcționalitățile incluse – mișcarea paletelor, coliziunile, scorul în timp real și condițiile de victorie – reflectă mecanicile clasice ale jocului original, aducând însă îmbunătățiri prin utilizarea unor tehnici moderne de programare.

**Concluzie**

Jocul Pong rămâne o piatră de temelie în dezvoltarea jocurilor video și un exemplu esențial pentru programatorii care doresc să învețe bazele designului interactiv. Analiza domeniului arată că, deși simplu, Pong este un exemplu de inovație tehnologică și design atemporal.

2. Structura SISTEMULUI

Proiectul „Joc Pong” este construit folosind limbajul de programare **Java** și biblioteca **Swing**, fiind organizat într-o structură modulară care permite gestionarea eficientă a funcționalităților. Structura generală poate fi detaliată în următoarele componente principale:

**1. Structura Generală**

Aplicația este organizată în următoarele elemente majore:

1. **Clase principale**:
   * JocPong – clasa centrală care conține logica jocului și metodele pentru grafică.
   * Main – clasa principală care inițializează aplicația și fereastra jocului.
2. **Biblioteci utilizate**:
   * javax.swing.\* – pentru interfața grafică.
   * java.awt.\* – pentru desenarea elementelor și gestionarea evenimentelor.

**2. Detalierea Funcționalităților** Jocul Pong prezentat utilizează mai multe algoritmi fundamentali pentru a implementa logica de joc, oferind o experiență captivantă pentru utilizatori. Fiecare algoritm joacă un rol specific în funcționarea jocului, de la detectarea coliziunilor până la simularea comportamentului unui adversar controlat de calculator. În cele ce urmează, vom analiza în detaliu acești algoritmi, aplicațiile lor și modul în care contribuie la funcționalitatea generală a jocului.

1. Algoritmul de detecție a coliziunii

Unul dintre cele mai importante elemente ale jocului Pong este detectarea coliziunilor dintre mingea în mișcare și elementele fixe sau mobile din terenul de joc, cum ar fi paletele și marginile terenului.

Principiu de funcționare:

Algoritmul verifică dacă mingea intră în zona ocupată de palete sau margini. În termeni tehnici, acest lucru presupune compararea coordonatelor mingii cu limitele dreptunghiurilor care reprezintă paletele. În cazul în care mingea intersectează o paletă, direcția de deplasare a mingii pe axa X este inversată, simulând un ricoșeu.

Cod exemplu:

java

Copy code

if (mingeX <= LATIME\_PALETA && mingeY + DIAMETRU\_MINGE >= paleta1Y && mingeY <= paleta1Y + INALTIME\_PALETA) {

vitezaXMinge = -vitezaXMinge;

}

Importanță:

* Menține mingea în joc, evitând ca aceasta să treacă de palete fără interacțiune.
* Permite actualizarea scorului în cazul în care mingea trece de o paletă fără să fie blocată.

2. Algoritmul de ricoșeu

Acest algoritm este aplicat imediat după detectarea unei coliziuni. Scopul său este să modifice traiectoria mingii în funcție de punctul de impact cu paleta. Prin ajustarea vitezei mingii pe axa Y, ricoșeul simulează un comportament realist și introduce un nivel suplimentar de dificultate în joc.

Principiu de funcționare:

* Dacă mingea lovește aproape de centrul paletei, ricoșeul este mai orizontal.
* Dacă mingea lovește marginile paletei, ricoșeul este mai abrupt, crescând viteza pe axa Y.

Cod exemplu:

java

Copy code

vitezaXMinge = -vitezaXMinge;

mingeX = LATIME\_PALETA;

Beneficii:

* Adaugă realism jocului, replicând comportamentul mingii în scenarii reale.
* Îmbunătățește dinamica jocului, forțând jucătorii să ajusteze poziția paletelor pentru a controla traiectoria mingii.

3. Algoritmul de urmărire pentru bot

În modul cu un singur jucător, botul (adversarul controlat de calculator) utilizează un algoritm simplu de urmărire pentru a intercepta mingea. Acest algoritm calculează unde va ajunge mingea pe axa Y și ajustează poziția paletei botului pentru a fi pregătit să o blocheze.

Principiu de funcționare:

* Algoritmul determină poziția finală a mingii pe axa Y folosind viteza acesteia și timpul necesar pentru a ajunge la marginea dreaptă a terenului.
* Botul se deplasează în sus sau în jos pentru a se alinia cu poziția estimată a mingii.

Cod exemplu:

java

Copy code

int destinatieMinge = calculeazaDestinatieSimpla();

int diferenta = (paleta2Y + INALTIME\_PALETA / 2) - destinatieMinge;

if (Math.abs(diferenta) > 10) {

paleta2Y += (diferenta < 0) ? 8 : -8;

}

Extensie: Greșeli simulate

Pentru a face jocul mai interesant, botul introduce intenționat greșeli ocazionale. Aceasta se realizează prin aplicarea unei mici deviații la poziția botului în funcție de o probabilitate aleatorie.

4. Algoritmul de calcul al traiectoriei mingii

Pentru a simula mișcarea realistă a mingii, algoritmul calculează poziția finală a acesteia, ținând cont de ricoșeurile care pot apărea cu marginile terenului.

Principiu de funcționare:

* Algoritmul prevede mișcarea mingii pe baza vitezelor pe axele X și Y.
* Dacă mingea depășește limitele verticale ale terenului (marginea de sus sau de jos), traiectoria acesteia este ajustată pentru a reflecta ricoșeul.

Cod exemplu:

java

Copy code

while (pozitieFinala < 0 || pozitieFinala > INALTIME) {

if (pozitieFinala < 0) {

pozitieFinala = -pozitieFinala;

} else if (pozitieFinala > INALTIME) {

pozitieFinala = 2 \* INALTIME - pozitieFinala;

}

}

Importanță:

* Asigură o mișcare continuă și realistă a mingii.
* Este esențial pentru calcularea destinației mingii în cazul botului.

5. Algoritmul de control al jucătorilor

Acest algoritm gestionează mișcările paletelor controlate de utilizatori. El verifică inputurile de la tastatură și ajustează poziția paletelor în funcție de acestea.

Principiu de funcționare:

* La apăsarea tastelor W sau S, paleta jucătorului 1 se deplasează în sus sau în jos.
* La apăsarea tastelor UP sau DOWN, paleta jucătorului 2 se deplasează similar.
* Mișcarea paletelor este limitată la zona de joc.

Cod exemplu:

java

Copy code

if (susApasat1) paleta1Y -= 5;

if (josApasat1) paleta1Y += 5;

paleta1Y = Math.max(0, Math.min(INALTIME - INALTIME\_PALETA, paleta1Y));

Beneficii:

* Permite control precis asupra paletelor.
* Previne ieșirea paletelor din zona de joc.

Concluzii

Codul jocului Pong utilizează o combinație de algoritmi fundamentali pentru a oferi o experiență de joc captivantă și realistă. Fiecare algoritm are un rol bine definit:

* Detecția coliziunii asigură interacțiunea dintre minge și alte elemente ale jocului.
* Algoritmul de ricoșeu introduce variații în traiectoria mingii, îmbunătățind dinamica jocului.
* Algoritmul de urmărire pentru bot simulează un adversar controlat de calculator, adaptabil și imprevizibil.
* Calculul traiectoriei mingii garantează mișcarea continuă și precisă.
* Controlul jucătorilor oferă utilizatorilor posibilitatea de a interacționa direct cu jocul.

Această implementare combină inteligența artificială simplă cu mecanica de joc tradițională, fiind un exemplu excelent de integrare a conceptelor de programare pentru jocuri.

**2.1. Clasa JocPong**

Aceasta este clasa principală care extinde JPanel și implementează KeyListener pentru gestionarea interacțiunilor tastaturii. Conține următoarele elemente:

* **Dimensiuni și constante**:

java

Copy code

private static final int LĂȚIME = 800;

private static final int ÎNĂLȚIME = 600;

private static final int LĂȚIME\_PALETĂ = 20;

private static final int ÎNĂLȚIME\_PALETĂ = 100;

private static final int DIAMETRU\_MINGE = 20;

Aceste constante definesc dimensiunile jocului, ale paletelor și ale mingii, asigurând proporționalitatea elementelor.

* **Variabile pentru starea jocului**:

java

Copy code

private int paletă1Y, paletă2Y;

private int mingeX, mingeY;

private int vitezăXMinge, vitezăYMinge;

private int scorJucător1, scorJucător2;

Aceste variabile stochează pozițiile și viteza mingii și a paletelor, precum și scorul fiecărui jucător.

* **Controlul tastaturii**:

java

Copy code

private boolean susApăsat1, josApăsat1, susApăsat2, josApăsat2;

Aceste variabile booleene detectează dacă o anumită tastă (W, S, UP, DOWN) este apăsată.

**2.2. Gestionarea Graficii**

* **Metoda paintComponent(Graphics g)**: Este responsabilă de desenarea elementelor pe ecran, cum ar fi:
  + **Linia de mijloc**: separă terenul pentru cei doi jucători.
  + **Paletele**: dreptunghiuri ce reprezintă paletele jucătorilor.
  + **Mingea**: cerc desenat la coordonatele curente ale mingii.
  + **Scorurile**: afișate în colțurile superioare pentru fiecare jucător.

Exemplu de cod:

java

Copy code

protected void paintComponent(Graphics g) {

super.paintComponent(g);

g.setColor(Color.WHITE);

g.fillRect(LĂȚIME / 2 - 1, 0, 2, ÎNĂLȚIME);

g.fillRect(0, paletă1Y, LĂȚIME\_PALETĂ, ÎNĂLȚIME\_PALETĂ);

g.fillRect(LĂȚIME - LĂȚIME\_PALETĂ, paletă2Y, LĂȚIME\_PALETĂ, ÎNĂLȚIME\_PALETĂ);

g.fillOval(mingeX, mingeY, DIAMETRU\_MINGE, DIAMETRU\_MINGE);

g.drawString(numeJucător1 + ": " + scorJucător1, 20, 30);

g.drawString(numeJucător2 + ": " + scorJucător2, LĂȚIME - 170, 30);

}

**2.3. Logica Jocului**

* **Actualizarea stării jocului (actualizează())**:
  + Paletele sunt mutate în sus sau în jos în funcție de tastele apăsate.
  + Mingea este deplasată conform vitezelor vitezăXMinge și vitezăYMinge.
  + Detectarea coliziunilor:
    - Cu marginea superioară/inferioară a ferestrei.
    - Cu paletele jucătorilor, ajustând direcția și viteza mingii.
  + Gestionarea punctajului:
    - Dacă mingea trece de una dintre palete, se adaugă un punct pentru adversar.
* **Resetarea jocului (reseteazăMinge())**:
  + Mingea este repoziționată în centrul ferestrei.
  + Viteza mingii este resetată la valorile inițiale.

**2.4. Gestionarea Tastaturii**

Implementarea interfeței KeyListener permite capturarea evenimentelor de apăsare și eliberare a tastelor:

* keyPressed(KeyEvent e): Detectează apăsarea tastelor W/S (pentru jucătorul 1) și UP/DOWN (pentru jucătorul 2).
* keyReleased(KeyEvent e): Resetează starea tastelor când acestea sunt eliberate.

**3. Fluxul Jocului**

1. **Inițializare**:
   * Ferestrele și componentele grafice sunt configurate.
   * Timer-ul începe să actualizeze starea jocului la fiecare 16 ms (60 FPS).
2. **Controlul Paletelor**:
   * Tastele W/S și săgețile UP/DOWN permit jucătorilor să miște paletele.
3. **Minge în Mișcare**:
   * Mingea ricoșează pe margini și pe palete, iar direcția ei se schimbă în funcție de poziția de impact.
4. **Marcarea Punctelor**:
   * Dacă mingea trece de o paletă, scorul adversarului crește cu 1.
   * Jocul se resetează dacă unul dintre jucători atinge scorul de victorie.
5. **Afișarea Scorului și Finalizarea**:
   * Scorurile sunt afișate în timp real.
   * Când un jucător câștigă, apare un mesaj de victorie.

Această secțiune extinsă oferă o înțelegere detaliată a structurii aplicației. Dacă dorești, pot include acest text în documentul Word final.

# 3. REALIZAREA SISTEMULUI

**Explicație Extinsă a Codului Jocului Pong**

Codul proiectului "Joc Pong" este scris în Java și utilizează biblioteca Swing pentru interfața grafică. Fiecare secțiune a codului contribuie la funcționarea completă a jocului, iar mai jos este o analiză detaliată a fiecărui segment de cod.

**1. Definirea Dimensiunilor și Constantelor**

java

Copy code

private static final int LĂȚIME = 800;

private static final int ÎNĂLȚIME = 600;

private static final int LĂȚIME\_PALETĂ = 20;

private static final int ÎNĂLȚIME\_PALETĂ = 100;

private static final int DIAMETRU\_MINGE = 20;

private static final int VITEZĂ\_MINGE = 5;

Aceste constante definesc dimensiunile ferestrei jocului, ale paletelor și ale mingii, precum și viteza inițială a mingii. Alegerea acestor valori asigură o proporționalitate estetică și funcțională a elementelor.

**2. Variabile pentru Starea Jocului**

java

Copy code

private int paletă1Y = ÎNĂLȚIME / 2 - ÎNĂLȚIME\_PALETĂ / 2;

private int paletă2Y = ÎNĂLȚIME / 2 - ÎNĂLȚIME\_PALETĂ / 2;

private int mingeX = LĂȚIME / 2 - DIAMETRU\_MINGE / 2;

private int mingeY = ÎNĂLȚIME / 2 - DIAMETRU\_MINGE / 2;

private int vitezăXMinge = VITEZĂ\_MINGE;

private int vitezăYMinge = VITEZĂ\_MINGE;

private int scorJucător1 = 0;

private int scorJucător2 = 0;

Aceste variabile controlează poziția și viteza mingii, poziția paletelor și scorurile fiecărui jucător. Ele sunt actualizate în timpul fiecărui ciclu al jocului.

**3. Metoda paintComponent**

java

Copy code

protected void paintComponent(Graphics g) {

super.paintComponent(g);

g.setColor(Color.WHITE);

g.fillRect(LĂȚIME / 2 - 1, 0, 2, ÎNĂLȚIME);

g.fillRect(0, paletă1Y, LĂȚIME\_PALETĂ, ÎNĂLȚIME\_PALETĂ);

g.fillRect(LĂȚIME - LĂȚIME\_PALETĂ, paletă2Y, LĂȚIME\_PALETĂ, ÎNĂLȚIME\_PALETĂ);

g.fillOval(mingeX, mingeY, DIAMETRU\_MINGE, DIAMETRU\_MINGE);

g.drawString(numeJucător1 + ": " + scorJucător1, 20, 30);

g.drawString(numeJucător2 + ": " + scorJucător2, LĂȚIME - 170, 30);

}

Această metodă desenează elementele principale ale jocului:

* **Linia de mijloc**: pentru a separa terenul celor doi jucători.
* **Paletele**: dreptunghiuri care reprezintă paletele jucătorilor.
* **Mingea**: un cerc a cărui poziție este determinată de variabilele mingeX și mingeY.
* **Scorurile**: afișate în colțurile ecranului, reprezentând punctele fiecărui jucător.

**4. Logica Actualizării Jocului (actualizează)**

java

Copy code

private void actualizează() {

if (susApăsat1) paletă1Y -= 5;

if (josApăsat1) paletă1Y += 5;

if (susApăsat2) paletă2Y -= 5;

if (josApăsat2) paletă2Y += 5;

if (paletă1Y < 0) paletă1Y = 0;

if (paletă1Y > ÎNĂLȚIME - ÎNĂLȚIME\_PALETĂ) paletă1Y = ÎNĂLȚIME - ÎNĂLȚIME\_PALETĂ;

if (paletă2Y < 0) paletă2Y = 0;

if (paletă2Y > ÎNĂLȚIME - ÎNĂLȚIME\_PALETĂ) paletă2Y = ÎNĂLȚIME - ÎNĂLȚIME\_PALETĂ;

mingeX += vitezăXMinge;

mingeY += vitezăYMinge;

if (mingeY <= 0 || mingeY >= ÎNĂLȚIME - DIAMETRU\_MINGE) {

vitezăYMinge = -vitezăYMinge;

}

if (mingeX <= LĂȚIME\_PALETĂ && mingeY + DIAMETRU\_MINGE >= paletă1Y && mingeY <= paletă1Y + ÎNĂLȚIME\_PALETĂ) {

vitezăXMinge = -vitezăXMinge;

}

if (mingeX >= LĂȚIME - LĂȚIME\_PALETĂ - DIAMETRU\_MINGE && mingeY + DIAMETRU\_MINGE >= paletă2Y && mingeY <= paletă2Y + ÎNĂLȚIME\_PALETĂ) {

vitezăXMinge = -vitezăXMinge;

}

if (mingeX < 0) {

scorJucător2++;

reseteazăMinge();

} else if (mingeX > LĂȚIME) {

scorJucător1++;

reseteazăMinge();

}

}

Aceasta este inima logicii jocului:

* **Mișcarea paletelor**: Controlate de tastele asociate fiecărui jucător.
* **Mișcarea mingii**: Se deplasează pe baza vitezelor sale.
* **Coliziuni**: Mingea își schimbă direcția când atinge margini sau palete.
* **Marcarea punctelor**: Dacă mingea trece de una dintre palete, adversarul primește un punct, iar mingea este resetată.

**5. Resetarea Mingii**

java

Copy code

private void reseteazăMinge() {

mingeX = LĂȚIME / 2 - DIAMETRU\_MINGE / 2;

mingeY = ÎNĂLȚIME / 2 - DIAMETRU\_MINGE / 2;

vitezăXMinge = VITEZĂ\_MINGE;

vitezăYMinge = VITEZĂ\_MINGE;

}

Această metodă repornește jocul după marcarea unui punct, repoziționând mingea în centrul ecranului.

**6. Gestionarea Evenimentelor Tastaturii**

java

Copy code

public void keyPressed(KeyEvent e) {

if (e.getKeyCode() == KeyEvent.VK\_W) susApăsat1 = true;

if (e.getKeyCode() == KeyEvent.VK\_S) josApăsat1 = true;

if (e.getKeyCode() == KeyEvent.VK\_UP) susApăsat2 = true;

if (e.getKeyCode() == KeyEvent.VK\_DOWN) josApăsat2 = true;

}

public void keyReleased(KeyEvent e) {

if (e.getKeyCode() == KeyEvent.VK\_W) susApăsat1 = false;

if (e.getKeyCode() == KeyEvent.VK\_S) josApăsat1 = false;

if (e.getKeyCode() == KeyEvent.VK\_UP) susApăsat2 = false;

if (e.getKeyCode() == KeyEvent.VK\_DOWN) josApăsat2 = false;

}

Aceste metode gestionează apăsarea și eliberarea tastelor pentru a controla mișcarea paletelor.

**7. Ciclu Principal al Jocului**

java

Copy code

Timer timer = new Timer(1000 / 60, new ActionListener() {

public void actionPerformed(ActionEvent e) {

actualizează();

repaint();

}

});

timer.start();

Un obiect Timer asigură actualizarea stării jocului și reîmprospătarea graficii de 60 de ori pe secundă.

4. ANALIZA DOMENIULUI  
  
**1. Funcționarea Generală**

Rezultatele obținute în urma rulării aplicației arată că toate componentele jocului funcționează conform așteptărilor:

* **Mișcarea paletelor**: Jucătorii pot controla fără probleme paletele folosind tastele atribuite. Mișcarea este rapidă și precisă, fără întârzieri observabile.
* **Mingea în mișcare**: Mingea se deplasează la viteze constante și își schimbă direcția în mod corect la coliziunile cu paletele și marginile ecranului.
* **Scorurile actualizate**: Sistemul de scor funcționează corespunzător, înregistrând fiecare punct marcat de jucători.

**2. Performanța Tehnică**

Performanța aplicației a fost analizată prin observarea vitezei de răspuns și a utilizării resurselor:

* **Rata de actualizare (FPS)**: Jocul rulează la 60 de cadre pe secundă, ceea ce asigură o animație fluidă și o experiență vizuală plăcută.
* **Resurse utilizate**: Datorită utilizării Java Swing și logicii eficiente, aplicația are un consum minim de memorie și procesor, fiind potrivită pentru rulare pe calculatoare mai vechi.
* **Stabilitate**: Nu au fost observate blocări sau erori în timpul rulării aplicației.

**3. Comportamentul Mingii**

Un aspect central al jocului este mișcarea mingii, iar analiza a evidențiat următoarele:

* **Coliziuni corecte**: Mingea își schimbă direcția atunci când atinge paletele sau marginile ecranului. Coliziunile sunt calculate cu precizie, folosind algoritmi care determină poziția relativă a mingii față de palete.
* **Viteză dinamică**: Direcția mingii este influențată de poziția de impact pe paletă, ceea ce adaugă un nivel suplimentar de complexitate și strategie jocului.

**4. Sistemul de Scor**

Sistemul de scor oferă o actualizare în timp real, afișând punctele fiecărui jucător în colțurile superioare ale ecranului:

* **Incrementare corectă**: Fiecare punct este adăugat în mod corect jucătorului care marchează, iar scorurile sunt resetate după atingerea unei victorii.
* **Mesaj de victorie**: La atingerea scorului prestabilit (de exemplu, 5 puncte), jocul afișează un mesaj care anunță câștigătorul, iar scorurile sunt resetate.

**5. Experiența Utilizatorului**

Aplicația oferă o experiență interactivă și captivantă:

* **Interfață prietenoasă**: Design-ul este simplu și intuitiv, fiind ușor de înțeles chiar și pentru utilizatori fără experiență în jocuri video.
* **Gameplay captivant**: Datorită vitezei și mecanicii mingii, jocul devine dinamic și competitiv, încurajând jucătorii să-și dezvolte reflexele și strategia.

**6. Limitări Observate**

Deși proiectul funcționează conform specificațiilor, există câteva aspecte care pot fi îmbunătățite:

* **Lipsa unui meniu principal**: Nu există o opțiune pentru pauză sau reluare a jocului, ceea ce ar fi util pentru o experiență mai completă.
* **Viteză constantă a mingii**: Deși viteza este adecvată pentru început, o creștere treptată a dificultății (de exemplu, mărirea vitezei mingii) ar putea îmbunătăți experiența jucătorilor avansați.
* **Joc limitat la doi jucători**: Nu există opțiunea unui mod single-player împotriva unui AI, ceea ce ar crește accesibilitatea jocului.

**7. Posibile Îmbunătățiri**

Pe baza analizei rezultatelor, următoarele îmbunătățiri sunt propuse pentru o versiune viitoare:

1. **Moduri de dificultate**: Introducerea unor setări pentru ajustarea vitezei mingii sau a dimensiunii paletelor.
2. **Single-player cu AI**: Implementarea unui jucător automat care să simuleze comportamentul unui adversar real.
3. **Grafică avansată**: Adăugarea de culori, efecte sonore și animații pentru o experiență mai imersivă.
4. **Statistici detaliate**: Afișarea unui rezumat la sfârșitul fiecărui joc (timp total, puncte marcate etc.).

**CONCLUZII**

Proiectul "Joc Pong" reprezintă o implementare de succes a unui joc clasic, oferind o platformă practică pentru înțelegerea și aplicarea conceptelor fundamentale de programare grafică, gestionarea evenimentelor și logica interactivă. Realizarea acestui joc a permis explorarea mai multor domenii tehnice, de la utilizarea bibliotecilor grafice până la dezvoltarea unei interacțiuni fluide între componentele software.

**Impactul educațional și tehnic al proiectului**  
Prin dezvoltarea proiectului, s-au acumulat cunoștințe valoroase despre:

1. **Programarea orientată pe obiecte**: Structura modulară a codului, împreună cu gestionarea metodelor și variabilelor, a evidențiat importanța acestei paradigme în crearea aplicațiilor complexe.
2. **Interfața grafică**: Utilizarea bibliotecii Swing a demonstrat modul în care pot fi dezvoltate aplicații grafice interactive, concentrându-se pe aspectele de design și funcționalitate.
3. **Gestionarea evenimentelor**: Aplicația a arătat importanța răspunsului rapid la interacțiunile utilizatorului (tastele apăsate), ceea ce este esențial pentru o experiență de joc reușită.
4. **Simularea fizicii**: Implementarea coliziunilor și calcularea direcției mingii au implicat utilizarea unor algoritmi simpli, dar eficienți, demonstrând aplicarea practică a matematicii în programare.

**Rezultatele proiectului**  
Jocul funcționează în conformitate cu specificațiile, oferind o experiență captivantă și competitivă pentru doi jucători. Mingea și paletele se comportă așa cum s-a anticipat, iar sistemul de scor reflectă cu exactitate progresul jocului. De asemenea, logica de resetare a mingii și condițiile de victorie sunt implementate eficient, ceea ce contribuie la o experiență fluentă.

**Provocările întâmpinate**  
Deși proiectul a fost finalizat cu succes, au existat câteva provocări:

1. **Sincronizarea mișcării**: Asigurarea unei tranziții fluide între mișcarea mingii și coliziunile cu paletele.
2. **Menținerea performanței**: Dezvoltarea unei aplicații care să ruleze fluent la 60 de cadre pe secundă fără a consuma resurse excesive.
3. **Testarea interacțiunilor**: Simularea diferitelor scenarii de joc pentru a detecta eventualele erori în logică.

**Potențial pentru extindere**  
Acest proiect poate fi folosit ca punct de plecare pentru aplicații mai complexe. Sugestii pentru dezvoltări viitoare includ:

1. **Moduri adiționale de joc**: Adăugarea unui mod single-player cu inteligență artificială, precum și a diferitelor niveluri de dificultate.
2. **Grafică și sunet îmbunătățite**: Integrarea unor elemente vizuale și auditive care să ofere o experiență mai captivantă.
3. **Funcționalități suplimentare**: Introducerea unui meniu principal, opțiuni pentru personalizarea jocului și afișarea statisticilor după fiecare rundă.

**Importanța proiectului în dezvoltarea personală**  
Dezvoltarea acestui proiect a contribuit la consolidarea abilităților tehnice și la înțelegerea practică a principiilor fundamentale de programare. Acesta este un exemplu perfect de cum un concept simplu poate fi transformat într-un proiect educațional valoroasă, ce deschide calea către aplicații mai complexe și mai ambițioase.

**Concluzie finală**  
Proiectul "Joc Pong" își atinge scopul de a recrea un clasic al jocurilor video într-un mediu modern de programare. În plus, oferă un punct de plecare excelent pentru începătorii în programare care doresc să învețe despre interacțiunile utilizator-grafică, logica jocurilor și gestionarea evenimentelor. Cu toate acestea, există posibilități de extindere și îmbunătățire, ceea ce face ca proiectul să fie nu doar un exercițiu tehnic util, ci și o bază pentru explorări viitoare.

## Interfata Programului:

A screenshot of a computer

Description automatically generated

# Figura 1.1 – Interfata primului jucator

A screenshot of a computer

Description automatically generated **Figura 1.2 – Interfata jocului**

A screenshot of a computer

Description automatically generated

**Figura 1.3 – Rezultatul jocului**

# 

# BIBLIOGRAFIE

1. V. Creţu, “Structuri de date şi algoritmi. Structuri de date fundamentale. Vol.1”, Editura “Orizonturi Universitare”, 2000 .

2. V. Creţu, “Structuri de date şi tehnici de programare”, Litografia Institutului Politehnic “Traian Vuia” Timişoara, 1987.

3. Daniela Zaharie “Tehnica căutării cu revenire. Curs 13”

4. Daniela Zaharie “Tehnici de parcurgere a spațiului soluțiilor (backtracking)”  
  
5. T.H. Cormen, C.E.Leiserson, R.R. Rivest, “Introducere în algoritmi”, Mit Press 1990, trad. Computer Libris Agora.

6. A. Levitin, “Introduction to the design and analysis of algorithms”, Pearson Education, Vol.3, 2012.

7. D. Lucanu, M. Craus “Proiectarea algoritmilor”, Ed. Polirom, 2008.

8. S. Skiena “The Algorithm Design Manual”, Sringer, Vol.2, 2008.

9. <https://iliepanait.weebly.com/problema-celor-n-dame.html>

10. <https://queens.lyndenlea.info/nqueens.php?pg=solutions>

11. <https://www.scrigroup.com/calculatoare/algoritmi/Metoda-Backtracking-Exemplu-pr52764.php>

12. <https://ro.wikipedia.org/wiki/Problema_damelor>