รายงาน

เรื่อง การออกแบบและพัฒนาเกมนกหินน้ำ (เป่ายิงฉุบ) โดยใช้ภาษา Java

จัดทำโดย

ทีมที่ 19

1. รหัสนิสิต 6830300479 นางสาวปรียาภักร แสงเกิด
2. รหัสนิสิต 6830300711 นางสาววรรณรัตน์ เอี่ยมยิ้ม

เสนอ

ผศ.ดร.กุลวดี สมบูรณ์วิวัฒน์

รายงานนี้เป็นส่วนหนึ่งของวิชา 03603112 – 65 Programming Fundamentals II

มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตศรีราชา

สารบัญ

โปรแกรมเกม นกหินน้ำ

1.1 กฎของเกม นกหินน้ำ

เกมนกหินน้ำเป็นเกมที่ผู้เล่นต้องพิมพ์คำสั่งผ่านทางหน้าจอแสดงผลแบบข้อความ (Console) เพื่อทำการแข่งขันกับระบบคอมพิวเตอร์ โดยการตัดสินผลแพ้ – ชนะจะเป็นไปตามกติกามาตรฐานสากล ดังต่อไปนี้

- ผู้เล่นพิมพ์คำว่า bird (นก) จะชนะ rock (หิน) เนื่องจากนกสามารถบินข้ามหรือหลบหลีกหินได้ แต่จะแพ้ water (น้ำ)
- ผู้เล่นพิมพ์คำว่า rock (หิน) จะชนะ water (น้ำ) เนื่องจากหินสามารถกั้นหรือถมทางน้ำได้ แต่จะแพ้ bird (นก)
- ผู้เล่นพิมพ์คำว่า water (น้ำ) จะชนะ bird (นก) เนื่องจากนกอาจจมน้ำได้ แต่จะแพ้ rock (หิน)
- ในกรณีที่ผู้เล่นและระบบคอมพิวเตอร์เลือกตัวเลือกเดียวกัน จะถือว่าผลการแข่งขันเสมอ
- หากผู้เล่นพิมพ์คำสั่งนอกเหนือจากตัวเลือกที่กำหนด ระบบจะแสดงข้อความแจ้งเตือนว่า “Enter input”

1.2 ระบบการให้คะแนน (Scoring System)

ระบบจะทำการคำนวณคะแนนสะสม (Rating) ให้แก่ผู้เล่นในแต่ละรอบของการเล่น โดยมีเกณฑ์การให้คะแนนดังนี้

- กรณีชนะ (Win) ผู้เล่นจะได้รับ 2000 rial
- กรณีเสมอ (Draw) ผู้เล่นจะได้รับ 1000 rial
- กรณีแพ้ (Lose) ผู้เล่นจะไม่ได้รับคะแนน และไม่มีการหักคะแนน

1.3 ระบบจัดอันดับ (Ranking Tiers)

เมื่อผู้เล่นสะสมคะแนนถึงเกณฑ์ที่กำหนด ระบบจะทำการจัดอันดับ (Rank) ให้โดยอัตโนมัติ ตามระดับคะแนนสะสม ดังนี้

- SPD No.1 : คะแนนระหว่าง 10,000 rial
- P’Keng hor!!! : คะแนนระหว่าง 5,000 – 9,999 rial
- Penguin : คะแนนระหว่าง 1,000 – 4,999 rial

1.4 คำสั่งระบบ (System Commands)

นอกเหนือจากคำสั่งที่ใช้ในการเล่นเกมตามปกติแล้ว ผู้เล่นยังสามารถใช้คำสั่งพิเศษของระบบเพื่อเรียกดูข้อมูลและสถิติต่าง ๆ ได้ โดยมีรายละเอียดดังต่อไปนี้

- **!rating**
  ใช้สำหรับตรวจสอบข้อมูลสถานะของผู้เล่นในปัจจุบัน ได้แก่ คะแนนสะสม (Rating) ระดับอันดับ (Rank) สถิติการชนะ–แพ้ และจำนวนการชนะต่อเนื่องในขณะนั้น (Current Win Streak) โดยไม่จำเป็นต้องออกจากเกม
- **!exit**
  ใช้สำหรับออกจากเกม โดยระบบจะแสดงสรุปสถิติการเล่นทั้งหมด (Total Statistics) ของผู้เล่น ได้แก่ อัตราการชนะ (Win Rate %) และจำนวนการชนะต่อเนื่องสูงสุด (Maximum Win Streak)

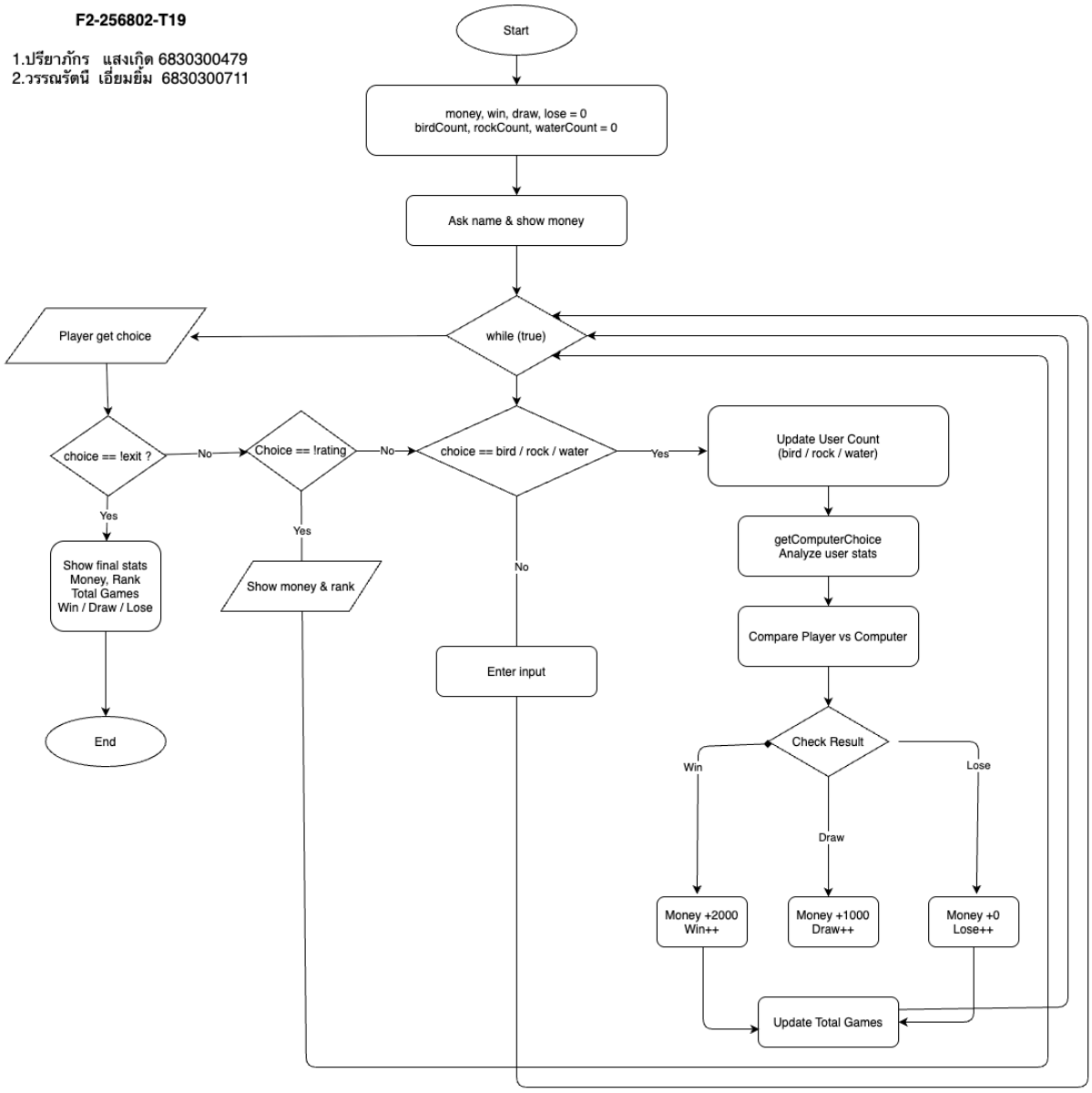
1.5 ระบบเพิ่มความยากของเกม (Difficulty Adjustment)

เพื่อเพิ่มความท้าทายในการเล่น เกมได้ออกแบบระบบปรับระดับความยากโดยอัตโนมัติ ซึ่งคอมพิวเตอร์จะทำการเรียนรู้พฤติกรรมการเลือกของผู้เล่น หากตรวจพบว่าผู้เล่นมีอัตราการชนะสูงหรือชนะต่อเนื่องเป็นจำนวนมาก ระบบจะเพิ่มโอกาสให้คอมพิวเตอร์เลือกตัวเลือกที่สามารถเอาชนะผู้เล่นได้ โดยกำหนดความน่าจะเป็นไว้ที่ประมาณร้อยละ 60 ทั้งนี้เพื่อรักษาสมดุลและความน่าสนใจของเกม

ตัวอย่างการเล่นเกม

```
Enter your name: primp
Hello, primp!
Current Money: 0 rial
Let's play Bird - Rock - Water!
Type: bird / rock / water
Type !exit to quit
> bird
You LOSE! Computer chose water
Money: 0 rial | Rank: No Rank
> rock
DRAW! (rock)
Money: 1000 rial | Rank: Penguin
> rock
You LOSE! Computer chose bird
Money: 1000 rial | Rank: Penguin
> water
You WIN! Computer chose bird
Money: 3000 rial | Rank: Penguin
> rock
You LOSE! Computer chose bird
Money: 3000 rial | Rank: Penguin
> bird
DRAW! (bird)
Money: 4000 rial | Rank: Penguin
> water
You WIN! Computer chose bird
Money: 6000 rial | Rank: P'Keng hor!!!
> rock
```

บทที่ 2
การออกแบบโปรแกรม



คำอธิบายขั้นตอนการทำงานของโปรแกรม Bird-Rock-Water

โปรแกรมนี้เป็นเกมที่ให้ผู้เล่นแข่งขันกับคอมพิวเตอร์โดยใช้รูปแบบ Bird-Rock-Water ซึ่งมีหลักการแพ้ชนะตามกติกาที่กำหนดไว้

1. เริ่มต้นโปรแกรม

- โปรแกรมเริ่มต้นด้วยการกำหนดค่าตัวแปรสำหรับเก็บสถานะต่าง ๆ เช่น จำนวนเงินเริ่มต้นของผู้เล่น รวมถึงสถิติการชนะ เสมอ และแพ้ ถูกตั้งค่าเป็นศูนย์

2. รับข้อมูลผู้เล่นและแสดงสถานะเบื้องต้น

- โปรแกรมทำการรับชื่อผู้เล่นผ่านคีย์บอร์ด จากนั้นแสดงข้อความต้อนรับพร้อมเงินเริ่มต้นและคำแนะนำเบื้องต้นสำหรับการเล่นเกม

3. เข้าสู่ลูปการเล่นเกมหลัก

- โปรแกรมเข้าสู่ลูปที่รอรับคำสั่งจากผู้เล่นอย่างต่อเนื่อง โดยมีเงื่อนไขการหยุดเล่นเมื่อผู้เล่นป้อนคำสั่ง "!exit"

4. รับคำสั่งจากผู้เล่น

- โปรแกรมรับคำสั่งที่ผู้เล่นป้อน ได้แก่ "bird", "rock", "water" หรือคำสั่งพิเศษ เช่น "!exit"

5. ตรวจสอบคำสั่งออกจากเกม

- หากผู้เล่นป้อน "!exit" โปรแกรมจะแสดงสรุปผลการเล่นทั้งหมด เช่น จำนวนเงินที่ได้รับ ยศ ปริมาณเกมที่เล่น และสถิติการชนะ เสมอ แพ้ ก่อนจะสิ้นสุดโปรแกรม

6. ตรวจสอบความถูกต้องของคำสั่ง

- หากผู้เล่นป้อนคำสั่งนอกเหนือจาก "bird", "rock", "water" และ "!exit" โปรแกรมจะแจ้งเตือนให้นำคำสั่งที่ถูกต้องและรอรับคำสั่งใหม่

7. บันทึกสถิติการเลือกของผู้เล่น

- หากคำสั่งถูกต้อง โปรแกรมจะทำการบันทึกจำนวนครั้งที่ผู้เล่นเลือกแต่ละตัวเลือก (Bird, Rock หรือ Water) เพื่อใช้วิเคราะห์แนวโน้ม

8. คอมพิวเตอร์เลือกตัวเลือกของตนเอง

- โปรแกรมจะวิเคราะห์สถิติการเลือกของผู้เล่น และสุ่มเลือกคำตอบที่เพิ่มโอกาสชนะผู้เล่น โดยพิจารณาจากตัวเลือกที่ผู้เล่นใช้บ่อยที่สุด

9. เปรียบเทียบผลการแข่งขัน

- โปรแกรมจะทำการเปรียบเทียบตัวเลือกของผู้เล่นกับของคอมพิวเตอร์ตามกติกาการชนะของเกม

10.ประเมินผลและปรับสถานะ

- หากผู้เล่นชนะ จะได้รับเงินรางวัล 2,000 rial และเพิ่มจำนวนชนะ
- หากเสมอ จะได้รับเงินรางวัล 1,000 rial และเพิ่มจำนวนเสมอ
- หากแพ้ จะไม่ได้รับเงิน และเพิ่มจำนวนแพ้

11.อัปเดตจำนวนเกมที่เล่น

- โปรแกรมบันทึกจำนวนเกมที่ผู้เล่นได้เล่นไปเพิ่มขึ้นหนึ่งครั้ง

12. แสดงผลสถานะปัจจุบัน

- โปรแกรมจะแสดงจำนวนเงินที่ผู้เล่นมี และยศปัจจุบันที่ผู้เล่นได้รับตามเกณฑ์

13.วนกลับเพื่อรับคำสั่งใหม่

- โปรแกรมจะวนลูปกลับไปรับคำสั่งใหม่จากผู้เล่นจนกว่าจะมีคำสั่ง "!exit" เพื่อออกจากเกม

14.สิ้นสุดโปรแกรม

- เมื่อผู้เล่นเลือกคำสั่ง "!exit" โปรแกรมจะแสดงสรุปผลสุดท้ายและจบการทำงาน

โครงสร้างของโปรแกรม

```java
rockwater.java
import java.util.Random;
import java.util.Scanner;

public class BirdRockWater {

    static int birdCount = 0;
    static int rockCount = 0;
    static int waterCount = 0;

    public static void greeting(Scanner scn, int money) {
        System.out.print("Enter your name: ");
        String name = scn.nextLine();
        System.out.println("Hello, " + name + "!");
        System.out.println("Current Money: " + money + " rial");
        System.out.println("Let's play Bird - Rock - Water!");
        System.out.println("Type: bird / rock / water");
        System.out.println("Type !exit to quit");
    }
```

1. **ส่วนการนำเข้าไลบรารี (Import Section)**
   โปรแกรมมีการนำเข้าไลบรารีมาตรฐาน ได้แก่ java.util.Scanner สำหรับรับข้อมูลจากผู้ใช้
   และ java.util.Random สำหรับการสุ่มค่าที่ใช้ในการตัดสินผลของเกม
2. **ส่วนของคลาสหลัก (Main Class)**
   คลาส BirdRockWater เป็นคลาสหลักของโปรแกรม ทำหน้าที่ควบคุมการทำงานทั้งหมด รวมถึงการเก็บตัวแปรและเมธอดที่เกี่ยวข้องกับเกม
3. **ส่วนของตัวแปรระดับคลาส (Class Variables)**
   ตัวแปรแบบ static ถูกใช้เพื่อเก็บข้อมูลสถิติการเลือกของผู้เล่น ได้แก่ นก หิน และน้ำ ซึ่งสามารถเรียกใช้งานได้จากทุกเมธอดภายในคลาส
4. **ส่วนของเมธอด (Methods)**
   โปรแกรมประกอบด้วยเมธอด greeting ซึ่งทำหน้าที่ต้อนรับผู้เล่น รับชื่อ แสดงข้อมูลเริ่มต้น และอธิบายวิธีการเล่นเกมก่อนเข้าสู่ขั้นตอนการทำงานหลัก

```java
    public static int win(String computer) {
        System.out.println("You WIN! Computer chose " + computer);
        return 2000;
    }

    public static int lose(String computer) {
        System.out.println("You LOSE! Computer chose " + computer);
        return 0;
    }

    public static int draw(String computer) {
        System.out.println("DRAW! (" + computer + ")");
        return 1000;
    }
```

5. โปรแกรมมีการกำหนดเมธอดสำหรับจัดการผลลัพธ์ของเกมจำนวน 3 เมธอด ได้แก่ win, lose และ draw ซึ่งทำหน้าที่แสดงผลการเล่นในแต่ละกรณี และคืนค่ารางวัลเป็นจำนวนเงิน
6. เมธอด win ใช้ในกรณีที่ผู้เล่นชนะเกม โดยจะแสดงข้อความแจ้งผลว่าผู้เล่นชนะ พร้อมทั้งแสดงตัวเลือกที่คอมพิวเตอร์เลือก และคืนค่าเงินรางวัลจำนวน 2000 หน่วย

7. เมธอด lose ใช้ในกรณีที่ผู้เล่นแพ้เกม โดยแสดงข้อความแจ้งผลการแพ้ พร้อมตัวเลือกของคอมพิวเตอร์ และคืนค่าเงินรางวัลเป็น 0 หน่วย
8. เมธอด draw ใช้ในกรณีที่ผลการแข่งขันเสมอ โดยแสดงข้อความแจ้งผลเสมอ และคืนค่าเงินรางวัลจำนวน 1000 หน่วย

```
static int playGame(String player, String computer) {
    if (player.equals(computer)) {
        return draw(computer);
    }
    if (player.equals("bird")) {
        return compupublicter.equals("rock") ? win(computer) : lose(computer);
    }
    if (player.equals("rock")) {
        return computer.equals("water") ? win(computer) : lose(computer);
    }
    if (player.equals("water")) {
        return computer.equals("bird") ? win(computer) : lose(computer);
    }
    return 0;
}

public static String getComputerChoice() {
    Random rand = new Random();

    if (birdCount >= rockCount && birdCount >= waterCount) {
        return rand.nextInt(100) < 60 ? "water"
                : rand.nextBoolean() ? "bird" : "rock";
    }

    if (rockCount >= birdCount && rockCount >= waterCount) {
        return rand.nextInt(100) < 60 ? "bird"
                : rand.nextBoolean() ? "rock" : "water";
    }

    if (waterCount >= birdCount && waterCount >= rockCount) {
        return rand.nextInt(100) < 60 ? "rock"
                : rand.nextBoolean() ? "bird" : "water";
    }

    String[] choices = {"bird", "rock", "water"};
    return choices[rand.nextInt(3)];
}
```

โปรแกรมประกอบด้วยเมธอดสำคัญ 2 เมธอด ได้แก่ playGame และ getComputerChoice ซึ่งทำหน้าที่ควบคุมกลไกหลักของเกมนก–หิน–น้ำ

1. เมธอด playGame ทำหน้าที่เปรียบเทียบตัวเลือกของผู้เล่นกับตัวเลือกของคอมพิวเตอร์ เพื่อใช้ตัดสินผลการแข่งขัน โดยเริ่มจากการตรวจสอบกรณีที่ทั้งสองฝ่ายเลือกเหมือนกัน ซึ่งจะถือว่าเสมอ จากนั้นจึงพิจารณาผลแพ้–ชนะตามกติกาของเกมในแต่ละกรณี และเรียกใช้เมธอด win, lose หรือ draw เพื่อแสดงผลลัพธ์และคืนค่าเงินรางวัลที่เหมาะสม
2. เมธอด getComputerChoice ทำหน้าที่สุ่มตัวเลือกของคอมพิวเตอร์ โดยอาศัยข้อมูลสถิติการเลือกของผู้เล่นในรอบก่อนหน้า ได้แก่ จำนวนครั้งที่เลือกนก หิน และน้ำ เพื่อปรับความน่าจะเป็นในการเลือกของคอมพิวเตอร์ให้มีความฉลาดมากขึ้น หากผู้เล่นเลือกตัวเลือกใดบ่อย คอมพิวเตอร์จะมีโอกาสเลือกตัวเลือกที่ได้เปรียบมากขึ้น แต่ยังคงมีการสุ่มเพื่อไม่ให้ผลลัพธ์ตายตัวจนเกินไป

```
public static String getRank(int money) {
    if (money >= 10000) return "SPD No.1";
    if (money >= 5000) return "P'Keng hor!!!";
    if (money >= 1000) return "Penguin";
    return "No Rank";
}
```

เมธอด getRank ทำหน้าที่กำหนดอันดับของผู้เล่นตามจำนวนเงินที่สะสมได้ในเกม โดยรับค่าจำนวนเงิน (money) เป็นพารามิเตอร์ และใช้เงื่อนไขแบบลำดับขั้นในการพิจารณาระดับของผู้เล่น หากผู้เล่นมีเงินตั้งแต่ 10,000 หน่วยขึ้นไป จะได้รับอันดับสูงสุด หากมีเงินตั้งแต่

5,000 หน่วยขึ้นไป จะได้รับอันดับรองลงมา และหากมีเงินตั้งแต่ 1,000 หน่วยขึ้นไป จะได้รับอันดับเริ่มต้น ส่วนผู้เล่นที่มีเงินต่ำกว่าที่กำหนดจะไม่ได้รับอันดับ

```java
public static void main(String[] args) {

    Scanner scn = new Scanner(System.in);
    int money = 0;
    int totalGames = 0;
    int win = 0, lose = 0, draw = 0;

    greeting(scn, money);

    while (true) {
        System.out.print("> ");
        String choice = scn.nextLine();

        if (choice.equals("!exit")) {
            System.out.println("\n=== GAME SUMMARY ===");
            System.out.println("Money: " + money + " rial");
            System.out.println("Rank: " + getRank(money));
            System.out.println("Games: " + totalGames);
            System.out.println("Wins: " + win);
            System.out.println("Draws: " + draw);
            System.out.println("Loses: " + lose);
            System.out.println("Bye!");
            break;
        }

        if (!(choice.equals("bird") || choice.equals("rock") || choice.equals("water"))) {
            System.out.println("Enter input");
            continue;
        }

        if (choice.equals("bird")) birdCount++;
        if (choice.equals("rock")) rockCount++;
        if (choice.equals("water")) waterCount++;

        String computer = getComputerChoice();
        int result = playGame(choice, computer);

        money += result;
        totalGames++;

        if (result == 2000) win++;
        else if (result == 1000) draw++;
        else lose++;
```

เมธอด main เป็นจุดเริ่มต้นของการทำงานของโปรแกรม ทำหน้าที่ควบคุมลำดับขั้นตอนทั้งหมดของเกมนก–หิน–น้ำ โดยเริ่มจากการสร้างออบเจกต์ Scanner เพื่อรับข้อมูลจากผู้ใช้ และกำหนดตัวแปรสำหรับเก็บข้อมูลสำคัญ ได้แก่ จำนวนเงินสะสม จำนวนรอบการเล่น และสถิติผลการแข่งขัน (ชนะ แพ้ และเสมอ)

เมื่อเริ่มต้นโปรแกรม จะมีการเรียกใช้เมธอด greeting เพื่อแสดงข้อความต้อนรับและข้อมูลเบื้องต้นแก่ผู้เล่น จากนั้นโปรแกรมจะเข้าสู่ลูปการเล่นเกมแบบไม่จำกัดรอบ โดยรอรับคำสั่งจากผู้ใช้ในแต่ละรอบ หากผู้ใช้ป้อนคำสั่ง !exit โปรแกรมจะแสดงสรุปผลการเล่นทั้งหมด เช่น เงินสะสม อันดับ จำนวนรอบ และสถิติการแข่งขัน ก่อนจบการทำงานของโปรแกรม

ในแต่ละรอบของการเล่น โปรแกรมจะตรวจสอบความถูกต้องของข้อมูลที่ผู้ใช้ป้อน หากไม่อยู่ในรูปแบบที่กำหนดจะให้ผู้ใช้ป้อนใหม่ เมื่อได้รับค่าที่ถูกต้อง โปรแกรมจะบันทึกสถิติการเลือกของผู้เล่น สุ่มตัวเลือกของคอมพิวเตอร์ เรียกใช้เมธอดตัดสินผลการแข่งขัน และนำผลลัพธ์ไปคำนวณเงินสะสม รวมถึงอัปเดตสถิติการเล่นและอันดับของผู้เล่นแบบเรียลไทม์