```java
// Harini. R 192324108
//JFo section-7 |
package test;

import java.util.Random;

class Card {
    private int cardNumber;
    private int creditBalance;
    private int ticketBalance;

    public Card(int cardNumber) {
        this.cardNumber = cardNumber;
        this.creditBalance = 0;
        this.ticketBalance = 0;
    }

    public int getCardNumber() {
        return cardNumber;
    }

    public int getCreditBalance() {
        return creditBalance;
    }

    public int getTicketBalance() {
        return ticketBalance;
    }

    public void addCredits(int credits) {
        if (credits > 0) {
            this.creditBalance += credits;
        }
    }

    public void subtractCredits(int credits) {
        if (credits > 0 && this.creditBalance >= credits) {
            this.creditBalance -= credits;
```

```java
            }
        }

    public void addTickets(int tickets) {
        if (tickets > 0) {
            this.ticketBalance += tickets;
        }
    }

    public void subtractTickets(int tickets) {
        if (tickets > 0 && this.ticketBalance >= tickets) {
            this.ticketBalance -= tickets;
        }
    }
}

class Game {
    private String gameName;
    private int requiredCredits;
    private Random random;

    public Game(String gameName, int requiredCredits) {
        this.gameName = gameName;
        this.requiredCredits = requiredCredits;
        this.random = new Random();
    }

    public void playGame(Card card) {
        if (card.getCreditBalance() >= requiredCredits) {
            card.subtractCredits(requiredCredits);
            int ticketsWon = random.nextInt(10);
            card.addTickets(ticketsWon);
            System.out.println("Card " + card.getCardNumber() + " won " + ticketsWon + " tickets. New ticket balance: " + card.getTicketBalance());
        } else {
            System.out.println("Card " + card.getCardNumber() + " has insufficient credits to play " + gameName + ".");
        }
    }
}
```

```java
class PrizeCategory {
    private String prizeName;
    private int ticketCost;
    private int prizeCount;

    public PrizeCategory(String prizeName, int ticketCost, int prizeCount) {
        this.prizeName = prizeName;
        this.ticketCost = ticketCost;
        this.prizeCount = prizeCount;
    }

    public String getPrizeName() {
        return prizeName;
    }

    public int getTicketCost() {
        return ticketCost;
    }

    public int getPrizeCount() {
        return prizeCount;
    }

    public void decrementPrizeCount() {
        if (prizeCount > 0) {
            prizeCount--;
        }
    }
}

class Terminal {
    private PrizeCategory[] prizes = new PrizeCategory[3];

    public Terminal(PrizeCategory prize1, PrizeCategory prize2, PrizeCategory prize3) {
        prizes[0] = prize1;
        prizes[1] = prize2;
        prizes[2] = prize3;
```

```java
115         }
116
117⊖    public void addCredits(Card card, int money) {
118            int creditsToAdd = money * 2;
119            card.addCredits(creditsToAdd);
120            System.out.println("Card " + card.getCardNumber() + " has been credited with " + creditsToAdd + " credits. New balance: " + card.getCreditBalance());
121        }
122
123⊖    public void checkBalance(Card card) {
124            System.out.println("Card " + card.getCardNumber() + " has " + card.getCreditBalance() + " credits and " + card.getTicketBalance() + " tickets.");
125        }
126
127⊖    public void transferCredits(Card fromCard, Card toCard, int credits) {
128            if (fromCard.getCreditBalance() >= credits) {
129                fromCard.subtractCredits(credits);
130                toCard.addCredits(credits);
131                System.out.println("Transferred " + credits + " credits from Card " + fromCard.getCardNumber() + " to Card " + toCard.getCardNumber());
132                checkBalance(fromCard);
133                checkBalance(toCard);
134            } else {
135                System.out.println("Insufficient credits to transfer.");
136            }
137        }
138
139⊖    public void transferTickets(Card fromCard, Card toCard, int tickets) {
140            if (fromCard.getTicketBalance() >= tickets) {
141                fromCard.subtractTickets(tickets);
142                toCard.addTickets(tickets);
143                System.out.println("Transferred " + tickets + " tickets from Card " + fromCard.getCardNumber() + " to Card " + toCard.getCardNumber());
144                checkBalance(fromCard);
145                checkBalance(toCard);
146            } else {
147                System.out.println("Insufficient tickets to transfer.");
148            }
149        }
150
151⊖    public void redeemPrize(Card card, String prizeName) {
152            for (PrizeCategory prize : prizes) {
```

```java
            if (prize.getPrizeName().equals(prizeName)) {
                if (card.getTicketBalance() >= prize.getTicketCost() && prize.getPrizeCount() > 0) {
                    card.subtractTickets(prize.getTicketCost());
                    prize.decrementPrizeCount();
                    System.out.println("Card " + card.getCardNumber() + " redeemed " + prizeName + ". Remaining count: " + prize.getPrizeCount());
                    checkBalance(card);
                } else {
                    System.out.println("Insufficient tickets or the prize is out of stock.");
                }
                return;
            }
        }
        System.out.println("Prize not found.");
    }
}

public class ArcadeTest {
    public static void main(String[] args) {
        Card card1 = new Card(1);
        Card card2 = new Card(2);

        PrizeCategory prize1 = new PrizeCategory("Stuffed Toy", 10, 5);
        PrizeCategory prize2 = new PrizeCategory("Candy", 5, 10);
        PrizeCategory prize3 = new PrizeCategory("Sticker", 2, 20);
        Terminal terminal = new Terminal(prize1, prize2, prize3);

        terminal.addCredits(card1, 10);
        terminal.addCredits(card2, 5);

        Game game = new Game("Win Random Tickets", 5);
        game.playGame(card1);
        game.playGame(card2);

        terminal.transferCredits(card1, card2, 5);
        terminal.transferTickets(card1, card2, 3);

        terminal.redeemPrize(card2, "Candy");
        terminal.redeemPrize(card2, "Stuffed Toy");
```

```java
191
192        game.playGame(card1);
193        terminal.redeemPrize(card1, "Sticker");
194        terminal.checkBalance(card1);
195        terminal.checkBalance(card2);
196    }
197 }
198
```

```
Card 1 has been credited with 20 credits. New balance: 20
Card 2 has been credited with 10 credits. New balance: 10
Card 1 won 1 tickets. New ticket balance: 1
Card 2 won 3 tickets. New ticket balance: 3
Transferred 5 credits from Card 1 to Card 2
Card 1 has 10 credits and 1 tickets.
Card 2 has 10 credits and 3 tickets.
Insufficient tickets to transfer.
Insufficient tickets or the prize is out of stock.
Insufficient tickets or the prize is out of stock.
Card 1 won 3 tickets. New ticket balance: 4
Card 1 redeemed Sticker. Remaining count: 19
Card 1 has 5 credits and 2 tickets.
Card 1 has 5 credits and 2 tickets.
Card 2 has 10 credits and 3 tickets.
```