Project 4 – Advanced Game of Roulette

Robert Zou & Mai Pham

CSCI 145 T/Th 8:00-11:10 AM

Coded in Eclipse

Source Files:

Casino.java – handles menu selection and all roulette games
Driver.java – starts Casino
Player.java – handles betting and player traits
Roulette.java – handles the game and calls player betting/payouts
VIP.java – derived class of player for VIP types
Wheel.java – static class used by Roulette to roll for random value

Details:

The Advanced Roulette program should have all base functionalities outlined in the project assignment packet implemented. This includes the assumptions made on the 3rd page of the packet as well as the final implementation outline on the 4th page of the packet.

report.txt files uploaded with source for both test case 1 and 2 below

EC attempted – We feel like we attempted an OOP approach. Some functions are quite lengthy (due in part to mathematical counting operations that could have been written as separate methods but were not used elsewhere, so we kept them in the larger method), but in general, we tried to keep all data within each class private or protected and used getters if necessary. We handled a moderate amount of exceptions, though not all, including: invalid bet amounts (insufficient chips or too high/low), queue errors (no one left in queue to add or table is full), but did not handle input validation for blatantly incorrect user input (e.g. putting in a string when asking for a betting option (1-3). Our test cases are comprehensive enough to test returns, betting amounts, re-entering games, maximum seating, chip exchange, and multiple bets. Our final file output "report.txt" does have some minor spacing/format issues with the individual transactions depending on the data input.

Test case 1 (tests all but invalid bet amount and new player addition):

Console:

Welcome to a simple version of roulette game.
You can place a bet on black, red, or a number.
A color bet is paid 2 the bet amount.
A number bet is paid 35 the bet amount.
You can bet on a number from 1 to 36.
Gamble responsibly. Have fun and good luck!

Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 1
Select a game (1-2): 1

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 2
Anonymous player have $100 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 3 0
You bet $15

Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36

Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) y
How much to bet in chips ($100, $25, $5 & $1)? 0 0 2 0
You bet $10
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36

Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) n

Jane Smith have $500 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 3 3
You bet $18
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36

Please enter a betting option: 3
Please enter a number: 15
Would you like to make a one more bet? (max 3, y/n) n

John Smith have $300 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 2 0
You bet $10
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36

Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) n

The number is: 0
The color is: Green
Anonymous player won $0
Anonymous player won $0
Play again [y/n]? n
Anonymous player get 0 cash back bonus.
Jane Smith won $0
Play again [y/n]? n
Jane Smith get 1 cash back bonus.
John Smith won $0
Play again [y/n]? y

Main Menu
1. Add a player to the game

2. Play one round
3. Quit

Option --> 3
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 2
Enter player type (0 - regular; 1 - VIP; 2 - SuperVIP): 1
Enter player money: 150
Enter player ID: 1234
Enter player name: Robert Zou
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 1
Select a game (1-2): 2

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player could not be added. Returning to the queue

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 2
Anonymous player have $500 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 2 0 0
You bet $50
Betting Options:
  1. Bet on black (even numbers)
  2. Bet on red (odd numbers)
  3. Bet on a number between 1 and 36
Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) n

Hot Shot have $1000 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 4 0
You bet $20
Betting Options:
  1. Bet on black (even numbers)
  2. Bet on red (odd numbers)
  3. Bet on a number between 1 and 36
Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) y
How much to bet in chips ($100, $25, $5 & $1)? 0 2 0 5
You bet $55
Betting Options:
  1. Bet on black (even numbers)
  2. Bet on red (odd numbers)
  3. Bet on a number between 1 and 36
Please enter a betting option: 3
Please enter a number: 24
Would you like to make a one more bet? (max 3, y/n) n

Anonymous player have $200 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 2 5 25
You bet $100
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36
Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) n

Anonymous player have $300 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 3 0
You bet $15
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36
Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) n

Charles B have $2000 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 1 1 0
You bet $30
Betting Options:
1. Bet on black (even numbers)
2. Bet on red (odd numbers)
3. Bet on a number between 1 and 36
Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) n

The number is: 36
The color is: Black
Anonymous player won $100
Play again [y/n]? n
Anonymous player get 0 cash back bonus.
Hot Shot won $0
Hot Shot won $0
Play again [y/n]? n
Hot Shot get 4 cash back bonus.
Anonymous player won $0
Play again [y/n]? n
Anonymous player get 0 cash back bonus.
Anonymous player won $30
Play again [y/n]? n
Anonymous player get 0 cash back bonus.
Charles B won $60
Play again [y/n]? n

Charles B get 2 cash back bonus.

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 3
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 1
Select a game (1-2): 1

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 2
John Smith have $200 cash and $90 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 0 0 4 0
You bet $20
Betting Options:
   1. Bet on black (even numbers)
   2. Bet on red (odd numbers)
   3. Bet on a number between 1 and 36
Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) y
How much to bet in chips ($100, $25, $5 & $1)? 0 0 0 15
You bet $15
Betting Options:
   1. Bet on black (even numbers)
   2. Bet on red (odd numbers)
   3. Bet on a number between 1 and 36
Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) n

The number is: 15
The color is: Red
John Smith won $40
John Smith won $0
Play again [y/n]? n
John Smith get 2 cash back bonus.

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 3
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 3

Report.txt:

```
Game: 100A1
Initial Balance: 9500
       Cash: $0
       $100 chips: 50
       $25 chips: 100
       $5 chips: 200
       $1 chips: 1000

Player 1 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 2 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 3 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip

Round 1 (Green 0)
Trans Player BAmount ($100 $25 $5 $1)  BType  Pay ($100 $25 $5 $1)
  1     1      15    (   0   0  3  0)   B   0  (   0   0  0  0)
  2     1      10    (   0   0  2  0)   R   0  (   0   0  0  0)

  3     2      18    (   0   0  3  3)   N(15)   0  (   0   0  0  0)

  4     3      10    (   0   0  2  0)   R   0  (   0   0  0  0)


Ending Balance: 9553
       Cash: $300
       $100 chips: 50
       $25 chips: 94
       $5 chips: 195
       $1 chips: 928

The difference amount for this session: 53
Game: 100A2
Initial Balance: 18500
       Cash: $0
       $100 chips: 100
       $25 chips: 200
       $5 chips: 500
       $1 chips: 1000

Player 1 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 2 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 3 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 4 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip
Player 5 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip

Round 1 (Black 36)
Trans Player BAmount ($100 $25 $5 $1)  BType  Pay ($100 $25 $5 $1)
  1     1      50    (   0   2  0  0)   B   100  (   1   0  0  0)
  2     2      20    (   0   0  4  0)   R   0  (   0   0  0  0)
```

```
  3     2      55    (   0   2  0  5)   N(24)   0  (   0   0  0  0)
  4     3      100    (   0   2  5  25)   R   0  (   0   0  0  0)
  5     4      15    (   0   0  3  0)   B   30  (   0   1  1  0)
  6     5      30    (   0   1  1  0)   B   60  (   0   2  2  0)


Ending Balance: 18580
       Cash: $500
       $100 chips: 99
       $25 chips: 194
       $5 chips: 485
       $1 chips: 905

The difference amount for this session: 80
Game: 100A1
Initial Balance: 9553
       Cash: $300
       $100 chips: 50
       $25 chips: 94
       $5 chips: 195
       $1 chips: 928

Player 3 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip

Round 2 (Red 15)
Trans Player BAmount ($100 $25 $5 $1)  BType  Pay ($100 $25 $5 $1)
  1     3      20    (   0   0  4  0)   R   40  (   0   1  3  0)
  2     3      15    (   0   0  0  15)   B   0  (   0   0  0  0)


Ending Balance: 9548
       Cash: $400
       $100 chips: 50
       $25 chips: 91
       $5 chips: 191
       $1 chips: 918

The difference amount for this session: -5
```

Test Case 2 (tests new player, full table rule, incorrect bet amounts for insufficient chip/incorrect range. Note I cycle out players in the queue to get to the newly added):

```
Welcome to a simple version of roulette game.
You can place a bet on black, red, or a number.
A color bet is paid 2 the bet amount.
A number bet is paid 35 the bet amount.
You can bet on a number from 1 to 36.
Gamble responsibly.  Have fun and good luck!

Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 2
Enter player type (0 - regular; 1 - VIP; 2 - SuperVIP): 2
Enter player money: 300
Enter player ID: 1234
Enter player name: Robert Zou
```

```
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 1
Select a game (1-2): 1

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player could not be added. Returning to the queue

Main Menu
1. Add a player to the game
2. Play one round
3. Quit
```

```
Option --> 1
Player could not be added. Returning to the queue

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player could not be added. Returning to the queue

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 3
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 1
Select a game (1-2): 2

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 1
Player added!

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 2
Robert Zou have $300 cash and $0 in chips.
Would you like to exchange $100 for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: y
Exchange completed.
How much to bet in chips ($100, $25, $5 & $1)? 1 0 0 0
You bet $100
The amount is invalid.
Please bet again ($100, $25, $5 & $1)? 0 0 0 3
You bet $3
The amount is invalid.
Please bet again ($100, $25, $5 & $1)? 0 2 5 0
You bet $75
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
    3. Bet on a number between 1 and 36
Please enter a betting option: 2
Would you like to make a one more bet? (max 3, y/n) y
How much to bet in chips ($100, $25, $5 & $1)? 0 0 0 20
You bet $20
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
```

```
    3. Bet on a number between 1 and 36
Please enter a betting option: 1
Would you like to make a one more bet? (max 3, y/n) n

The number is: 10
The color is: Black
Robert Zou won $0
Robert Zou won $40
Play again [y/n]? n
Robert Zou get 5 cash back bonus.

Main Menu
1. Add a player to the game
2. Play one round
3. Quit

Option --> 3
Main Menu
1. Select a game
2. Add a new player to the list
3. Quit

Choose an option (1-3): 3
```

Report.txt:

```
Game: 100A1
Initial Balance: 9500
       Cash: $0
       $100 chips: 50
       $25 chips: 100
       $5 chips: 200
       $1 chips: 1000

Ending Balance: 9500
       Cash: $0
       $100 chips: 50
       $25 chips: 100
       $5 chips: 200
       $1 chips: 1000

The difference amount for this session: 0
Game: 100A2
Initial Balance: 18500
       Cash: $0
       $100 chips: 100
       $25 chips: 200
       $5 chips: 500
       $1 chips: 1000

Player 1 exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip

Round 1 (Black 10)
Trans Player BAmount ($100 $25 $5 $1)  BType  Pay ($100 $25 $5 $1)
  1     1      75    (   0   2  5  0)   R   0  (   0   0  0  0)
  2     1      20    (   0   0  0  20)   B   40  (   0   1  3  0)


Ending Balance: 18555
```

```
        Cash: $100
        $100 chips: 100
        $25 chips: 199
        $5 chips: 497
        $1 chips: 995

The difference amount for this session: 55
```

Source Code:

```
// Class Casino for CSCI 145 Project 4 Fall '17
// Modified by: Robert Zou & Mai Pham

import java.util.*;
import java.io.*;

public class Casino {
	private PrintWriter outFile;
	private Queue<Player> playerQueue = new LinkedList<Player>();
	private Roulette games[] = new Roulette[5];
	private int numberOfGames;
	private String model;

	public Casino() {   // Casino constructor setting up player and roulette data
structures
		try {
			outFile = new PrintWriter("report.txt");
		}
		catch (FileNotFoundException e) {
			System.out.println("File " + e + " not found.");
		}
		this.numberOfGames = 0;
		this.inputGames();
		this.inputPlayers();
		// for debugging
		// while (!playerQueue.isEmpty()) {
		//     System.out.println(playerQueue.remove());
		// }
	}

	public void inputPlayers() {           // read players.txt into player queue
		try {
			File playerFile = new File("players.txt");
			Scanner sc = new Scanner(playerFile);
			while (sc.hasNextLine()) {
				int playerID;
				String playerName;
				int playerType = sc.nextInt();
				int playerMoney = sc.nextInt();
				if (playerType != 0) {
					playerID = sc.nextInt();
					playerName = sc.nextLine();
					playerName = playerName.substring(1);
```

```
                                Player tempPlayer = new VIP(playerMoney, playerType,
playerID, playerName);
                                playerQueue.add(tempPlayer);
                        }
                        else {
                                Player tempPlayer = new Player(playerMoney);
                                playerQueue.add(tempPlayer);
                        }
                }
        }
        catch (FileNotFoundException e){
                e.printStackTrace();
        }
}

public void inputGames() {// read games.txt into games array
        try {
                File gameFile = new File("games.txt");
                Scanner sc = new Scanner(gameFile);
                model = sc.next();
                int numGames;
                numGames = sc.nextInt();
                numberOfGames += numGames;
                for (int i = 0; i < numGames; i++) {
                        int minBet = sc.nextInt();
                        int maxBet = sc.nextInt();
                        int hundreds = sc.nextInt();
                        int twentyfives = sc.nextInt();
                        int fives = sc.nextInt();
                        int ones = sc.nextInt();
                        Roulette tempGame = new Roulette(model, minBet, maxBet,
hundreds, twentyfives, fives, ones);
                        games[i] = tempGame;
                }
        }
        catch (FileNotFoundException e) {
                e.printStackTrace();
        }
}

public void addPlayer() {
        Scanner sc = new Scanner(System.in);
        int playerType;
        int playerMoney;
        System.out.print("Enter player type (0 - regular; 1 - VIP; 2 -
SuperVIP): ");
        playerType = sc.nextInt();
        System.out.print("Enter player money: ");
        playerMoney = sc.nextInt();
        if (playerType != 0) {
                int playerID;
                String playerName;
                System.out.print("Enter player ID: ");
                playerID = sc.nextInt();
                sc.nextLine(); // get rid of \n
```

```
                        System.out.print("Enter player name: ");
                        playerName = sc.nextLine();
                        Player tempPlayer = new VIP(playerMoney, playerType, playerID,
playerName);
                        playerQueue.add(tempPlayer);
                }
                else {
                        Player tempPlayer = new Player(playerMoney);
                        playerQueue.add(tempPlayer);
                }
        }

        public void start() {      // start the games/menu selection
                Scanner sc = new Scanner(System.in);
                int mainSelection = 0;             // priming selection
                while (mainSelection != 3) {     // initial menu
                        System.out.println("Main Menu");
                        System.out.println("1. Select a game");
                        System.out.println("2. Add a new player to the list");
                        System.out.println("3. Quit\n");
                        System.out.print("Choose an option (1-3): ");
                        mainSelection = sc.nextInt();
                        switch(mainSelection) {    // main menu selection
                                case 1: {
                                        System.out.print("Select a game (1-" + numberOfGames
+ "): ");
                                        int rouletteChoice = sc.nextInt() - 1; // roulette
choice mapped to index in gameArray
                                        // File file = new File("report.txt");
                                        // PrintWriter outFile = new
PrintWriter("report.txt");
                                        // outFile = new PrintWriter ("report.txt");
                                        outFile.println("Game: " + model +
(rouletteChoice+1));
                                        outFile.print("Initial ");
                                        int initialB;
                                        initialB = games[rouletteChoice].Balance(outFile);
                                        // create Roulette object
                                        int gameSelection = 0;
                                        while (gameSelection != 3) {
                                                System.out.println("\nMain Menu");
                                                System.out.println("1. Add a player to the
game");
                                                System.out.println("2. Play one round");
                                                System.out.println("3. Quit\n");
                                                System.out.print("Option --> ");
                                                gameSelection = sc.nextInt();
                                                switch(gameSelection) {    // select roulette
operations.
                                                        case 1: {
                                                                // add player from queue to game
& remove from queue
                                                                if (playerQueue.isEmpty())
                                                                        System.out.println("No
players in the queue.");
```

```
                                                        else {
                                                                Player addedPlayer =
playerQueue.remove();
                                                                boolean canAdd =
games[rouletteChoice].addPlayer(addedPlayer);
                                                                // if the game is full,
add the player back to the queue
                                                                if (!canAdd) {

        playerQueue.add(addedPlayer);

        System.out.println("Player could not be added. Returning to the queue");
                                                                }
                                                                else {


        System.out.println("Player added!");
                                                                }
                                                        }
                                                        break;
                                                }
                                                case 2:         {

        games[rouletteChoice].playRound(outFile);
                                                                if
(games[rouletteChoice].getNumPlayers() == 0) {
                                                                        gameSelection = 3;
                                                                        System.out.println("No
players remaining in the game.");
                                                                }
                                                                break;
                                                }
                                                case 3:         {
                                                                int endingB;
                                                                outFile.print("Ending ");
                                                                endingB =
games[rouletteChoice].Balance(outFile);
                                                                outFile.println("The difference
amount for this session: " + (endingB - initialB));
                                                                break;
                                                }
                                                default: {
                                                                System.out.println("Invalid
option. Try again (1-3)");
                                                                break;
                                                }
                                        }
                                }
                                break;
                        }
                        case 2: {
                                this.addPlayer();
                                break;
                        }
                        case 3: {
                                // exit - no code needed
```

```
                                // for each game printsummary
                                outFile.close();
                                break;
                        }
                        default: {
                                System.out.println("Invalid option. Try again (1-
3)");
                                break;
                        }
                }
        }
}
```

```
// Class Driver for CSCI 145 Project 4 Fall '17
// Modified by: Robert Zou & Mai Pham

public class Driver {
	public static void main(String[] args) {
		Casino mainCasino = new Casino();
		Wheel.welcomeMessage();
		mainCasino.start();
	}
}
```

```
// Class Player for CSCI 145 Project 2 Fall 17
// Modified by: Robert Zou & Mai Pham

import java.io.PrintWriter;
import java.util.*;

//*************************************************************************
//   Class Player represents one roulette player.
//*************************************************************************
class Player	{
	protected String name;                          // player name
	protected int hundredChips;                     // # of 100 chips <player
possession>
    protected int twentyFiveChips;                  // # of 25 chips <player
possession>
    protected int fiveChips;                        // # of 5 chips <player
possession>
    protected int oneChips;                         // # of 1 chips <player
possession>
	boolean active = false;                         // is customer playing or
not
	protected int amountBet;                    //total amount $$ of bet
	protected int numberBet    ;                    //number of bet
	protected int cash;                             //cash in hand
	protected int []one;                            // # of 1 bet
	protected int []five;                           // # of 5 bet
	protected int []twentyFive;                     // # of 25 bet
	protected int []hundred;                    // # of 100 bet
    //protected int money;                      //total of chips
```

```
    protected int bet[];                                // amount of bet
    protected int numBet;                               // # of bet
    protected int betType[];                            // bet color
    protected int number[];                                 // bet number
    protected int winAmount[];                          // amount win

    public Player (int amount)   {
              name = "Anonymous player";
              cash = amount;
              active = true;
              oneChips = 0;
              fiveChips = 0;
              twentyFiveChips = 0;
              hundredChips = 0;
              amountBet = 0;
              numBet = 0;
              one = new int[4];
              five = new int[4];
              twentyFive = new int [4];
              hundred = new int [4];
              bet = new int[4];
              betType = new int[4];
              number = new int[4];
              winAmount = new int[4];
       }

    public boolean isPlaying()   {
              return active;
    }

    public String getName()      {
              return name;
    }
    public int getCash()  {
              return cash;
    }

    public void exchange$100()   {
              cash -= 100;
              twentyFiveChips += 2;
              fiveChips += 5;
              oneChips += 25;
    }

    public int totalChips()      {
              int totalChips = 100*hundredChips + 25*twentyFiveChips + 5*fiveChips +
oneChips;
              //System.out.println("total chips :" + totalChips);
              return totalChips;
    }

    public int cashBack() {
              return 0;
    }
```

```
    public int getNumBet(){
            return numBet;
    }
    public void makeBet(Scanner scan, int minBet, int maxBet)    {
            String answer = "y";
            numBet = 0;
            while (answer.equalsIgnoreCase("y") && numBet < 3)
            {
                  System.out.print("How much to bet in chips ($100, $25, $5 & $1)?
");
                  hundred[numBet] = scan.nextInt();
                  twentyFive[numBet] = scan.nextInt();
                  five[numBet] = scan.nextInt();
                  one[numBet] = scan.nextInt();
                  bet[numBet] = 100*hundred[numBet] + 25*twentyFive[numBet] +
5*five[numBet] + one[numBet];
                  System.out.println("You bet $" + bet[numBet]);
                  while (bet[numBet] < minBet || bet[numBet] > this.totalChips() ||
bet[numBet] > maxBet || hundred[numBet] > hundredChips || twentyFive[numBet] >
twentyFiveChips || five[numBet] > fiveChips || one[numBet] > oneChips)  {
                        System.out.println("The amount is invalid.");
                        System.out.print("Please bet again ($100, $25, $5 & $1)?
");
                        hundred[numBet] = scan.nextInt();
                        twentyFive[numBet] = scan.nextInt();
                        five[numBet] = scan.nextInt();
                        one[numBet] = scan.nextInt();
                        bet[numBet] = 100*hundred[numBet] + 25*twentyFive[numBet]
+ 5*five[numBet] + one[numBet];
                        System.out.println("You bet $" + bet[numBet]);
                  }
                  hundredChips -= hundred[numBet];
                  twentyFiveChips -= twentyFive[numBet];
                  fiveChips -= five[numBet];
                  oneChips -= one[numBet];
                  amountBet += bet[numBet];
                  numberBet++;

                  Wheel.betOptions();
                  System.out.print("Please enter a betting option: ");
                  betType[numBet] = scan.nextInt();
                  while (betType[numBet] < 1 || betType[numBet] > 3)  {
                        System.out.print("The betting option is invalid. \nPlease
enter betting option again: ");
                        betType[numBet] = scan.nextInt();
                  }
                  if (betType[numBet] == Wheel.NUMBER)          {
                        System.out.print("Please enter a number: ");
                        number[numBet] = scan.nextInt();
                        while (number[numBet] < Wheel.MIN_NUM || number[numBet] >
Wheel.MAX_NUM)      {
                              System.out.print("The number is invalid. \nPlease
enter a number again: ");
                              number[numBet] = scan.nextInt();
                        }
```

```
                    }
                    System.out.print("Would you like to make a one more bet? (max 3,
y/n) ");
                    answer = scan.next();
                    numBet++;
               }
               System.out.println();
     }         // method makeBet

     public int getWinAmount(int numBet){
               return winAmount[numBet];
     }
     public int getHundreds(int numBet) {
        return hundred[numBet];
     }
     public int getTwentyFives(int numBet) {
        return twentyFive[numBet];
     }
     public int getFives(int numBet) {
        return five[numBet];
     }
     public int getOnes(int numBet) {
        return one[numBet];
     }

     public int payment(int numTrans, int playerSeat, PrintWriter outFile)      {
               //int chips;
               //int hChips, tFChips, fChips, oChips;
               int amount;
               String output = "";
               String bType = "";
               for (int i = 0; i < numBet; i++) {
                    numTrans++;
                    output += "  " + numTrans + "     "+ (playerSeat+1);
                    output += "       " + bet[i];
                    output += "    (" + "   " + hundred[i] + "   " + twentyFive[i] +
"  " + five[i] + "  " + one[i] +")   ";
                  winAmount[i] = Wheel.payoff(bet[i], betType[i], number[i]);
                  amount = winAmount[i];
                  System.out.println(name + " won $" + amount);
                  if (betType[i] == 1)
                     bType = "B";
                  if (betType[i] == 2)
                     bType = "R";
                  if (betType[i] == 3)
                     bType = "N(" + number[i] + ")";
                  output += bType + "   " + amount + "  (";

                  // convert money to chips
                  hundred[i] = (amount/100);
                  hundredChips += hundred[i];
                  amount %= 100;
                  twentyFive[i] = (amount/25);
                  twentyFiveChips += twentyFive[i];
                  amount %= 25;
```

```
                five[i]= (amount/5);
                fiveChips += five[i];
                amount %= 5;
                one[i] = amount;
                oneChips += one[i];
                output += "   " + hundred[i] + "   " + twentyFive[i] + "  " +
five[i] + "  " + one[i] +")";
                // reset for next line
                outFile.println(output);
                output = "";
                bType = "";
                //hChips = 0;
                //tFChips =0;
                //fChips = 0;
                //oChips = 0;
            }
            return numTrans;
    }
    /*
    public void displayStatus()
    {
            System.out.println("The total amount player win/lose is: " + winLose);
    }*/

    public boolean playAgain(Scanner scan)    {
       String answer;
       System.out.print ("Play again [y/n]? ");
       answer = scan.next();
       if (!answer.equalsIgnoreCase("y"))
              active = false;
       return (answer.equals("y") || answer.equals("Y"));
    }  // method playAgain

    // for debugging
    public String toString() {
       String output = this.cash + " " + this.name + " ";
       return output;
    }
}


// Class Roulette for CSCI 145 Project 4 Fall '17
// Modified by: Robert Zou & Mai Pham

import java.util.*;
import java.io.*;

class Roulette
{
    private Player playerList[] = new Player[5];           // up to 5 players per
game
    private int numPlayers;                                          // number of
players in the game
    //private Transaction TransList[] = new Transaction[30];      // up to 30
transactions
```

```
    protected int numTrans;                                          // number of
transactions
    private int numRounds;                                       // number of rounds
play
    private int money;                                               // house
money
    private int hundredChips;                                    // # of 100 chips
    private int twentyfiveChips;                                 // # of 25 chips
    private int fiveChips;                                       // # of 5 chips
    private int oneChips;                                        // # of 1 chips
    private int minBet;                                              // min bet
for this roulette
    private int maxBet;                                              // max bet
for this roulette
    private String gameID;                                       // game id (game
type + instance num) - created through string addition
    Scanner scan = new Scanner(System.in);

    public Roulette(String ID, int min, int max, int hundred, int twentyfive, int
five, int one) {
            gameID = ID;
            minBet = min;
            maxBet = max;
            hundredChips = hundred;
            twentyfiveChips = twentyfive;
            fiveChips = five;
            oneChips = one;
            //money = 100*hundred + 25*twentyfive + 5*five + one;
            numPlayers = 0;
            numTrans = 0;
            numRounds = 0;
      }

    public int Balance(PrintWriter outFile)   {
            int balance;
            balance = money + 100*hundredChips + 25*twentyfiveChips + 5*fiveChips +
oneChips;
            outFile.println("Balance: " + balance);
            outFile.println("\tCash: $" + money);
            outFile.println("\t$100 chips: " + hundredChips);
            outFile.println("\t$25 chips: " + twentyfiveChips);
            outFile.println("\t$5 chips: " + fiveChips);
            outFile.println("\t$1 chips: " + oneChips);
            outFile.println();
            return balance;
    }

    public int getNumPlayers() {
            return numPlayers;
    }

    public boolean addPlayer(Player incomingPlayer) {
            for (int i = 0; i < 5; i++) {
                  if (playerList[i] == null || !playerList[i].isPlaying()) {
                        playerList[i] = incomingPlayer;
```

```
                        numPlayers++;
                        return true;
                  }
            }
            return false;
      }

      public void playRound(PrintWriter outFile) {
            numRounds++;
            String answer;
            for (int i = 0; i < 5; i++)      {
                  if (playerList[i] != null && playerList[i].isPlaying())    {
                        System.out.println(playerList[i].getName() + " have $" +
playerList[i].getCash() + " cash and $" + playerList[i].totalChips() + " in chips.");
                        if (playerList[i].getCash() >= 100)    {
                              System.out.print("Would you like to exchange $100
for chips? (2 x $25; 5 x $5; 25 x $1) - y/n: ");
                              answer = scan.next();
                              if (answer.equalsIgnoreCase("y")) {
                                    this.exchange$100();
                                    playerList[i].exchange$100();
                                    System.out.println("Exchange completed.");
                                    outFile.println("Player " + (i + 1) + "
exchanges $100 for 2 $25-chip, 5 $5 chip, 25 $1-chip");
                              }
                        }
                        playerList[i].makeBet(scan, this.minBet, this.maxBet);
                        for (int j = 0; j < playerList[i].getNumBet(); j++) {
                              hundredChips += (playerList[i].getHundreds(j));
                              twentyfiveChips +=
(playerList[i].getTwentyFives(j));
                              fiveChips += (playerList[i].getFives(j));
                              oneChips += (playerList[i].getOnes(j));
                        }
                  }
            }
            Wheel.spin();
            numTrans = 0;
            outFile.println("\nRound " + numRounds + " (" + Wheel.getColor() + " "
+Wheel.getNumber() + ")");
            outFile.println("Trans Player BAmount ($100 $25 $5 $1)  BType  Pay ($100
$25 $5 $1)");
            for (int i = 0; i < 5; i++)      {      // 5 seats at table
                  if (playerList[i] != null && playerList[i].isPlaying()) {
                        //numTrans++;
                        //outFile.print("  " + numTrans + "     "+ (i+1));
                        //outFile.print(playerList[i].payment(numTrans, i,
outFile));
                        numTrans = playerList[i].payment(numTrans, i, outFile);
                        outFile.println();
                        for (int j = 0; j < playerList[i].getNumBet(); j++) {
                              if (playerList[i].getWinAmount(j) > 0) {
                                    hundredChips -=
(playerList[i].getHundreds(j));
```

```
                                        twentyfiveChips -=
(playerList[i].getTwentyFives(j));
                                        fiveChips -= (playerList[i].getFives(j));
                                        oneChips -= (playerList[i].getOnes(j));
                                    }
                                }

                                if(!playerList[i].playAgain(scan))     {
                                    System.out.println(playerList[i].getName() + " get "
+ playerList[i].cashBack() + " cash back bonus.");
                                    playerList[i] = null;
                                }
                            }
                        }
                        outFile.println();
    }

    public void exchange$100()   {
                        money += 100;
                        twentyfiveChips -= 2;
                        fiveChips -= 5;
                        oneChips -= 25;
    }
}

// Class VIP for CSCI 145 Project 2 Fall 17
// Modified by: Robert Zou & Mai Pham

class VIP extends Player
{
      final private double CASH_BACK = 0.05;              // VIP cash back %
      private int cashBack = 0;                                 // amount cash back

      private int playerType;                                          // player
type
      private int id;                                                        //
player ID

      public VIP(int amount, int type, int ID, String newName)   {
            super(amount);
            playerType = type;
            id = ID;
            this.name = newName;
      }

      public int getPlayerType(){
            return playerType;
      }

      public int getID()  {
            return id;
      }

      public String getName()   {
            return name;
```

```
	}

	public int cashBack()	{
		double cash = 0;
		if (playerType == 1 || playerType == 2){
			cash = amountBet * CASH_BACK;
			cashBack = (int)Math.round(cash);
			if (playerType == 2){
				if(numberBet <= 20 && numberBet >= 10)
					cashBack += 20;
				else if (numberBet > 20)
					cashBack += 50;
			}
		}
		return cashBack;
	}

	// for debugging
    public String toString() {
	String output = this.cash + " " + this.name + " " + this.playerType + " " +
this.id;
	return output;
    }
}


// Class Wheel for CSCI 145 Project 4 Fall '17
// Modified by: Robert Zou & Mai Pham

//*************************************************************************
//   Class Wheel represents a roulette wheel and its operations.  Its
//   data and methods are static because there is only one wheel.
//*************************************************************************

class Wheel
{
    // public name constants -- accessible to others
    public final static int BLACK     =  0;		// even numbers
    public final static int RED       =  1;		// odd numbers
    public final static int GREEN     =  2;		// 00 OR 0
    public final static int NUMBER    =  3;		// number bet
    public final static int MIN_NUM   =  1;		// smallest number to bet
    public final static int MAX_NUM   = 36;		// largest number to bet

    // private name constants -- internal use only
    private final static int MAX_POSITIONS = 38;	// number of positions on
wheel
    private final static int NUMBER_PAYOFF = 35;	// payoff for number bet
    private final static int COLOR_PAYOFF  = 2;		// payoff for color bet

    // private variables -- internal use only
    private static int ballPosition;				// 00, 0, 1 .. 10
    private static int color;					// GREEN, RED, OR
BLACK
```

```
    //=========================================================================
    //  Presents welcome message
    //=========================================================================
    public static void welcomeMessage()
    {
       System.out.println("Welcome to a simple version of roulette game.");
       System.out.println("You can place a bet on black, red, or a number.");
       System.out.println("A color bet is paid " + COLOR_PAYOFF + " the bet
amount.");
       System.out.println("A number bet is paid " + NUMBER_PAYOFF + " the bet
amount.");
       System.out.println("You can bet on a number from " + MIN_NUM + " to " +
MAX_NUM + ".");
       System.out.println("Gamble responsibly.  Have fun and good luck!\n");
    }


    //=========================================================================
    //  Presents betting options
    //=========================================================================
    public static void betOptions()    {
       System.out.println("Betting Options:");
       System.out.println("    1. Bet on black (even numbers)");
       System.out.println("    2. Bet on red (odd numbers)");
       System.out.println("    3. Bet on a number between " + MIN_NUM +
                  " and " + MAX_NUM);
    }


    public static void spin()    {
             ballPosition = (int)(Math.random()*(MAX_POSITIONS));

             if (ballPosition == 0 || ballPosition == (MAX_NUM+1))            {
                   if (ballPosition == (MAX_NUM+1))
                         System.out.println("The number is: 00");
                   else
                         System.out.println("The number is: " + ballPosition);
                   color = GREEN;
                   System.out.println("The color is: Green");
             }
             else   {
                   System.out.println("The number is: " + ballPosition);
                   if (ballPosition % 2 == BLACK)   {
                         color = BLACK;
                         System.out.println("The color is: Black");
                   }
                   else if (ballPosition % 2 == RED){
                         color = RED;
                         System.out.println("The color is: Red");
                   }
             }
    }

    public static int getNumber()      {
```

```
            return ballPosition;
    }

    public static String getColor()    {
            String output = "";
            if (color == GREEN)
                  output = "Green";
            if (color == BLACK)
                  output = "Black";
            if (color == RED)
                  output = "Red";
            return output;
    }

    public static int payoff(int bet, int betType, int number)    {
            int payoff = 0;
            if (betType == NUMBER && number == ballPosition)
                  payoff = bet * NUMBER_PAYOFF;
            else if (betType == 1 && color == BLACK || betType == 2 && color == RED)
                  payoff = bet * COLOR_PAYOFF;
            return payoff;
    }
}
```