Exámen métodos de programación

José Mauricio Muñoz Diéguez

Un pequeño resumen del proyecto...

Casino ATS el cual contiene varios juegos:

- Blackjack
- Connecta 4
- Adivina Adivinador
- Tragamonedas

Organización del código

Todo en sus respectivas clases separadas por carpetas

Games: Contiene las clases de los 4 juegos.
Util: Clases de utilidad para realizar diferentes acciones (limpiar consola, imprimir con colores, etc.)

Root: Contiene la clase Main, User y Game.

Menú (`Casino.java`)

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

    ConsoleUtil.clearConsole();

    System.out.println(Color.BLUE + "Ingresa tu nombre de usuario:" + Color.RESET);
    String username = scanner.nextLine();

    ConsoleUtil.clearConsole();

    System.out.println("----------------------------------------");
    System.out.println(Color.RED + "Bienvenido, " + username + " al Casino ATS!" + Color.RESET);
    System.out.println("----------------------------------------");

    float balance;

    do {
      System.out.println(Color.BLUE + "Ingresa tu saldo inicial." + Color.RESET
          + " (Debe ser mayor a $50.00 y menor a $1000.00):");
      balance = scanner.nextFloat();
    } while (balance < 50.0 || balance > 1000.0);

    User user = new User(username, balance);

    handleMenu(user);

}
```

```
...

choice = scanner.nextInt();

ConsoleUtil.clearConsole();

switch (choice) {
  case 1:
    handleGame(user);
    break;
  case 2:
    System.out.println("\nTu saldo es: " + Color.GREEN + user.getBalance() + Color.RESET);
    ConsoleUtil.pressAnyKeyToContinue();
    ConsoleUtil.clearConsole();
    break;
  case 3:
    user.playHistory.printPlayHistory(10);
    ConsoleUtil.pressAnyKeyToContinue();
    ConsoleUtil.clearConsole();
    break;
  case 4:
    System.out.println("\nGracias por jugar!");
    scanner.close();
    System.exit(0);
    break;
  default:
    System.out.println(Color.RED + "La opción que elegiste es inválida" + Color.RESET);
    break;
}
```

Menú de juegos

Ejemplo: **Tragamonedas**

```
switch (choice) {
  case 1:
    Slot slot = new Slot(user, bet);

    Game.handleInstructions(slot.NAME,
        "El programa mostrará 3 palabras aleatorias. Si 3...");

    while (slot.status != Game.GameStatus.COMPLETED) {
      slot.play();
    }
    ConsoleUtil.clearConsole();
    break;
}
```

Otros juegos que son un poco diferente...

Ejemplo: **Blackjack**

```
while (blackjack.status != Game.GameStatus.COMPLETED) {
  blackjack.play();
  // Ask if the user wants to play another round
  System.out.println("\nDeseas jugar otra ro...");
  String ans = scanner.next();
  if (ans.equals("n")) {
    blackjack.finishGame();
    break;
  }
}
```

Un vistazo a las clases de utilidad

Color.java

Clase estática que permite impimir en consola a color
(No funciona en Windows)

```
public static final String GREEN = "\u001B[32m";
```

ConsoleUtil.java

Clase estática que ayuda a limpiar la consola y a pedir al usuario que presione `enter` para continuar

```java
// Limpia la consola
public static void clearConsole() {
  System.out.print("\033[H\033[2J");
  System.out.flush();
}

// Le indica al usuario que espere para dar enter
public static void pressAnyKeyToContinue() {
  System.out.println("\nPresiona [enter] para continuar...");
  try {
    System.in.read();
  } catch (Exception e) {
    e.printStackTrace();
  }
}
```

PlayHistory.java

Una clase que guarda todas las partidas que he hecho, si gané o perdí, y cuánto diero se me sumó o restó

```
List<PlayInfo> playHistory;
```

Donde `PlayInfo` es una clase que contiene...

```
String game; // El nombre del juego
float bet; // La apuesta ganada/perdida
Game.WinLoseStatus status; // Si ganó, perdió o quedó empate
```

`Stopwatch.java`

Clase de utilidad que permite tener un cronometro y tiene funciones como `elapsedTime()` o `reset()`

Utiliza por dentro `System.currentMillis()`

Dejando a un lado las clases de utilidad...

Clase `Game.java`

Una clase con funciones estáticas que contienen lógica que comparten todos los juegos

- Status
- Lógica cuando se gana
- Lógica cuando se pierde
- Etc.

Cuando el usuario gana algún juego...

```
public static void handleWin(User user, float bet, float ratio, String gameName) {
  float oldBalance = user.getBalance();

  float balanceToAdd = bet * ratio - bet;

  user.addBalance(balanceToAdd);

  System.out
      .println(
          Color.GREEN + "\n\nFelicidades, has ganado " + balanceToAdd + "$");
  System.out.println("Saldo Anterior: " +  oldBalance + "$");
  System.out.println("Saldo Actual: "+ user.getBalance() + "$\n");
  user.playHistory.addPlay(gameName, bet, WinLoseStatus.WIN);
  ConsoleUtil.pressAnyKeyToContinue();
}
```

Función que llama `handleLose()` para burlarse del jugador...

```
public static void makeLoserSentence() {
  int randomIndex = new Random().nextInt(loserStrings.length);
  System.out.println(loserStrings[randomIndex]);
}
```

Listado de todas las funciones:

- `handleBet()` : Se encarga de pedirle al usuario la apuesta a realizar y verifica si es válida
- `handleWin()` : Muestra en consola que el usuario ganó cierta cantidad de dinero. Lo modifica de la clase `User`
- `handleLose()` : Lo mismo que `handleWin()` solo que resta dinero y manda a llamar a `makeLoserSentence()`
- `handleDraw()` : Lo mismo que las dos anteriores solo que no te quita dinero
- `makeLoserSentence()` : Se burla del usuario utilizando frases pre-escritas
- `handleInstructions()` : Muestra en consolas las instrucciones de un juego determinado

Clase `User.java`

Contiene información básica del usuario como el nombre, salario y su historial de jugadas.

```
public class User {

  String name;
  float balance;
  PlayHistory playHistory;
}
```

Cuenta también con funciones para accesar y modificar cada una de ellas.

Ahora si lo divertido...

Las clases de los juegos