Rummikub

EL CLÁSICO JUEGO EN PYTHON

Programación — 1º DAW

Sergio Majada

David Ramírez

Sergi Darder

Índice

Investigación y desarrollo conceptual (IDC)

En este apartado indicaremos cómo jugar al Rummikub para dos personas:

1. Preparación

1.1. Se adquiere un conjunto de fichas conformado por cuatro grupos en color negro, rojo, azul y naranja del 1 al 13.

1.2. Se barajan las fichas.

1.3. Cada jugador coge 14 fichas.

1.4. Se elige el primer turno de forma aleatoria.

2. Jugar turnos

2.1. El jugador que tiene el turno tiene dos opciones:

2.1.1. Coger una ficha del montón principal.

2.1.2. Añadir una ficha al tablero.

2.2. Si no quiere sacar más fichas, pasa de turno.

3. Finalizar partida

3.1. La partida finaliza cuando un jugador se queda sin fichas.

3.2. Gana el jugador que primero se quede sin fichas.

4. Reglas del turno

- En el primer turno, la suma de los números de las fichas que el jugador añada en el tablero debe ser superior o igual a 30.
- Los conjuntos de fichas deben ser de dos tipos:
    - Una escalera de fichas del mismo color.
    - Un conjunto de fichas con el mismo número y cada una de diferente color.

Traslado a entorno de programa (TEP)

Preparación

1. Crear el conjunto de fichas
    - Crear una lista vacía de fichas = [ ]
2. Crear los colores
    a. Crear lista con 4 elementos [“negro”, “azul”, “naranja”, “rojo”]
    b. Para cada ficha de color en la lista colores (hasta 13):
        - Crear una carta y añadir al mazo
3. Cambiar el orden de los elementos de la lista aleatoriamente con la librería “random”
4. Definir el número de jugadores
    - Crear un input para saber el número de jugadores (de 2 a 4)
5. Crear una mano de fichas del jugador vacía = [ ]
6. Crear el tablero.
    - Crear una lista vacía, el tablero = [ ]
7. Ejecutar la función darBienvenida()
8. Pedir al usuario el número de jugadores y asignarlo a numJugadores.
    a. Si numJugadores es menor que 2 o mayor que 4, mostrar un mensaje de error y volver a pedir el número de jugadores
        - Mostrar un mensaje "¡Perfecto! En unos segundos estará todo listo."
9. Ejecutar la función progressBar()
10. Repartir 14 fichas a cada jugador.
11. Asignar 0 a jugadorTurno, 1 a partidaTurno, y True a jugando.
12. Asignar True a primerTurno.
13. Asignar False a puedePasar y True a puedeCogerFicha.
14. Ejecutar la función startGameTxt()

Jugar turnos

1. Iniciar un bucle while con la variable jugando:
2. Si no es primerTurno:
    a. Asignar True a puedeCogerFicha y False a puedePasar.
    b. Iniciar un bucle while con la variable acabarAccion:
    c. Ejecutar la función mostrarEstadoJuego() con los argumentos jugadorTurno y jugadores[jugadorTurno].
    d. Pedir al usuario que ingrese un número de acción.
        i. Si la acción es 1 y puedeCogerFicha es True, ejecutar la función opcion1() con el argumento jugadores[jugadorTurno].
        ii. Si la acción es 2, ejecutar la función opcion2(). Si la función devuelve True, asignar False a puedeCogerFicha y True a puedePasar.
        iii. Si la acción es 3, ejecutar la función opcion3() con el argumento jugadores[jugadorTurno]. Si la función devuelve True, asignar False a puedeCogerFicha y True a puedePasar
        iv. Si la acción es 4, asignar True a acabarAccion
        v. Si la acción no es válida, mostrar un mensaje de error "Opción no válida"
3. Si és primerTurno:
    a. Asignar True a puedeCogerFicha y False a puedePasar.
    b. Iniciar un bucle while con la variable acabarAccion:
    c. Ejecutar la función mostrarEstadoJuego() con los argumentos jugadorTurno y jugadores[jugadorTurno].
    d. Pedir al usuario que ingrese un número de acción.

i. Si la acción es 1 y puedeCogerFicha es True, ejecutar la función opcion1() con el argumento jugadores[jugadorTurno].
ii. Si la acción es 2, ejecutar la función opcion2(). Si la función devuelve True, asignar False a puedeCogerFicha y True a puedePasar.
iii. Si la acción es 3, ejecutar la función opcion3() con el argumento jugadores[jugadorTurno]. Si la función devuelve True, asignar False a puedeCogerFicha y True a puedePasar.
iv. Si la acción es 4, asignar True a acabarAccion y False a primerTurno.
v. Si la acción no es válida, mostrar un mensaje de error "Opción no válida"

Interacción del jugador

Se muestra el menú del primer jugador con el tablero y las fichas en su mano.

El usuario elige una de las opciones que se muestran:

1. Opción 1. Coger ficha
    a. Si la elige, aparece una ficha más en su mano
2. Opción 2. Añadir ficha a una posición existente.
    a. Pide la ficha que se quiere añadir
    b. Pide el conjunto al que se quiere unir
    c. Se verifica que el movimiento entra dentro de las reglas.
    d. Añade esa ficha a ese conjunto y, en caso de escalera, lo filtra en orden ascendente.
3. Opción 3. Añadir ficha a una posición nueva.
    a. Pide las fichas que se quieren añadir.
    b. Se verifica que el movimiento entra dentro de las reglas.

    c. Añade esas ficha a un conjunto nuevo y de nuevo, en caso de escalera, lo filtra en orden ascendente.

4. Opción 4. Pasa turno.
    a. Muestra la pantalla del siguiente jugador.

Terminar partida

1. Asignar al jugador siguiente (jugadorTurno + 1) % numJugadores a jugadorTurno
2. Incrementar partidaTurno en 1
3. Revisar la condición para continuar el juego y asignar el valor correspondiente a jugando.
4. Repetir el bucle principal mientras jugando sea True.
5. Finalizar el juego.

Codificación y creación de funciones (CCF)

Lo primero que haremos será importar las librerías que vamos a usar. En este caso: *random, os, time, math y rich*.

A continuación, definiremos las funciones que nos servirán más adelante. En este caso, hemos definido un total de 23 funciones, las cuales hemos dividido en dos categorías:

- Funciones estéticas
- Función de juego

Dentro de las funciones estéticas, encontramos:

- La función "prRed" es para imprimir un texto en color rojo.
- La función "prOrange" es para imprimir un texto en color naranja.
- La función "prBlue" es para imprimir un texto en color azul.
- La función "prPurple" es para imprimir un texto en color morado.
- La función "prGreen" es para imprimir un texto en color verde.
- La función "borrarTerminal" es para limpiar el terminal.
- La función "pintarFichaColor" es para imprimir un texto en un color específico.
- La función "darBienvenida" es para dar la bienvenida al juego y mostrar información sobre los creadores.
- La función "progressBar" es para mostrar una barra de progreso mientras se carga el juego.
- La función "startGameTxt" es para mostrar un mensaje de inicio del juego.
- La función "finalPartida" es para mostrar un mensaje de fin de partida y el ganador.

Dentro de las funciones de juego, encontramos:

- La función "conjuntoFichas()" se utiliza para crear un conjunto de fichas con los colores "Naranja", "Azul", "Negro" y "Rojo", y los valores del 1 al 13. Estas fichas se almacenan en una lista y se devuelven al final de la función.
- La función "shuffleFichas(fichas)" se utiliza para barajar las fichas. Se recorre la lista de fichas y se intercambian fichas en una posición aleatoria. La lista barajada se devuelve al final de la función.
- La función "cogerFichas(numFichas)" se utiliza para tomar un número específico de fichas de la lista de fichas barajadas. El número de fichas se pasa como argumento y se devuelve una lista con las fichas tomadas.
- La función "mostrarMano(jugador, manoJugador)" se utiliza para mostrar la mano del jugador actual. El número de jugador y la mano del jugador se pasan como argumentos.
- La función "mostrarTurno(puedePasar, puedeCogerFicha)" se utiliza para mostrar las opciones que tiene el jugador en su turno. La función recibe dos argumentos booleanos que indican si el jugador puede pasar su turno o tomar una ficha.
- La función "mostrarTablero(tablero)" se utiliza para mostrar el tablero del juego. El tablero se pasa como argumento.
- Las funciones "opcion1(manoJugador)", "opcion2()" y "opcion3 (manoJugador)" se utilizan para manejar las opciones del turno del jugador. La primera función se utiliza para añadir una ficha a la mano del jugador, la segunda función se utiliza para añadir una ficha a un conjunto existente en el tablero y la tercera para añadir una ficha en un nuevo conjunto.
- Las funciones "reglasOpcion2()" y "reglasOpcion3()" se utilizan para determinar las reglas del juego, determinan si un conjunto es válido o no según la relación entre colores y números.

- La función “finalPartida(jugador)” se utiliza al final del juego para mostrar quién es el jugador de la partida.

Una vez definidas las funciones, se declararán las variables que utilizaremos en el juego:

1. "fichasRummy" es una lista que contiene todas las fichas creadas con la función "conjuntoFichas()" y barajadas con la función "shuffleFichas(fichas)". Esta variable se utiliza para repartir fichas a los jugadores y para mantener un registro de las fichas disponibles en el juego.
2. "jugadores" es una lista de listas que contiene las manos de cada jugador. Cada elemento de la lista principal representa un jugador y contiene una lista con las fichas en su mano.
3. "tablero" es una lista de listas que contiene los conjuntos de fichas en el tablero. Cada elemento de la lista principal representa un conjunto de fichas en el tablero y contiene una lista con las fichas en ese conjunto.
4. "jugadorTurno" es una variable entera que indica el número del jugador que tiene el turno actual. Esta variable se utiliza para determinar el jugador actual en el juego y para tomar decisiones en función de su turno.
5. "partidaTurno" es una variable entera que indica el número de turno de la partida. Esta variable se utiliza para llevar el registro de los turnos en la partida.
6. "jugando" es una variable booleana que indica si el juego sigue en curso. Si es verdadero, el juego seguirá ejecutándose, si es falso, el juego terminará.
7. "primerTurno" es una variable booleana que indica si es el primer turno en la partida. Esta variable se utiliza para determinar si el jugador puede pasar el turno.
8. "splitFicha" es una lista que contiene dos elementos: el color y el valor de una ficha en particular. Esta variable se utiliza para dividir la ficha en sus dos partes y hacer comparaciones.

   a. "color" es una variable de cadena que contiene el color de una ficha en particular. Esta variable se utiliza para hacer comparaciones.
   b. "valorFicha" es una variable entera que contiene el valor de una ficha en particular. Esta variable se utiliza para hacer comparaciones.
9. "puedePasar" es una variable booleana que indica si el jugador puede pasar el turno. Esta variable se utiliza para determinar si mostrar o no la opción de pasar en el menú de opciones.
10. "puedeCogerFicha" es una variable booleana que indica si el jugador puede tomar una ficha. Esta variable se utiliza para determinar si mostrar o no la opción de tomar una ficha en el menú de opciones.

Tras la declaración de estas variables globales, procedemos a la creación de un bucle while que nos servirá como menú principal. La idea es que podamos seleccionar, mediante unos controles, la opción deseada:

La variable "jugando" es una variable booleana que indica si el juego está activo o no. Mientras "jugando" sea verdadero, el bucle se ejecutará.

Dentro del bucle, se llaman a las funciones "mostrarMano" y "mostrarTablero" para mostrar la mano del jugador actual y el estado del tablero del juego. A continuación, se llama a la función "mostrarTurno" para mostrar las acciones disponibles para el jugador en ese momento. El jugador elige una acción a realizar mediante una entrada del usuario.

Dependiendo de la acción elegida (1, 2 o 3), se llama a una de las funciones "opcion1", "opcion2" u "opcion3" correspondientes. Si el jugador elige una opción no válida, se le notifica y se le vuelve a preguntar por una acción.

Si el jugador elige la acción 4, se sale del bucle interior y se comprueba si la mano del jugador está vacía. Si es así, se termina el juego y se llama a la función "finalPartida" con el

número del jugador ganador. Si no es así, se suma 1 al turno de la partida y se cambia al siguiente jugador. Si el turno es el del último jugador, se vuelve al primer jugador.

Pruebas y corrección de errores (PCE)

1. ***Fallo:*** El bucle no accede a la opción 4

   Solución: Se elimina la condición "si acción == 4", porque el bucle sale cuando se le introduce 4. Los pasos que debía seguir entonces se sacan del bucle para que se acceda.

2. ***Fallo:*** En el primer turno del jugador 2 no aparece la opción 1: "Coger ficha".

   Solución: Se añade el booleano "puedeCogerFicha = True" dentro del bucle principal del juego cuando no es el primer turno. De esta manera inicializamos el booleano de nuevo ya que después de recorrer el bucle de "Primer turno" se quedaba en False.

3. ***Fallo:*** La opción 2 no se muestra cuando hay un conjunto en el tablero

   Solución: Se cambia la condición en la función "mostrarTurno()" para que se muestre la opción 2 cuando el turno de la partida sea mayor a 1 o cuando el tablero tenga más de un conjunto.

4. ***Fallo:*** Al acabar el turno de *jugador 2* y pulsar en "Opción 4: Pasar turno" da un Index error: "line 424, IndexError: list index out of range"

   Solución: Existía un error al cambiar el turno del jugador, que se ha arreglado sumando 1 al turno y si el turno es igual al número de jugadores, cambiarlo a 0 de nuevo.

5. ***Fallo:*** Las funciones "reglasOpcion2()" y "reglasOpcion3()" solo se aplican al *jugador 1*.

   Solución: Dentro de cada función se determina con condicionales y con la variable "jugadorTurno()" que se apliquen las reglas para comparar el conjunto del *jugador 1, 2* o *3*.

6. ***Fallo:*** No quita las fichas de la mano del jugador

   Solución: Se ha cambiado la función reglasOpción3() y reglasOpción2() para que haga un .pop cuando se añada en el tablero.

7. ***Fallo:*** La opción 2 no se muestra cuando hay un solo conjunto.

   Solución: Se cambia la condición de la función mostrarTurno() para que la opción 2 muestre también cuando la longitud de la variable conjunto sea mayor a 0.

8. ***Fallo:*** Cuando eres player 2, aparece la opción 4: pasar turno y no debería.

   Solución: Se ha cambiado el momento en que debe aparecer la opción 4 para que únicamente sea en los turnos que no son los primeros.

9. ***Fallo:*** Al insertar un carácter diferente a un número en la variable "acción" se imprime un ValueError.

   Solución: Se ha añadido una comprobación de errores para ValueError, de esta manera vuelve a pedir el input de la variable "acción" cada vez que el programa detecta un ValueError.

10. ***Fallo:*** Al crear un nuevo conjunto y añadir una ficha al principio de la array "nuevoConjunto", se duplica la ficha y es añadida dos veces.

    Solución: La operación *append* estaba duplicada en dos funciones distintas, al borrarla de una de ellas se ha solucionado el error.