Instituto de Formación Técnica Superior N°21

Asignatura: Estructura de Datos

Información Anexa Mini-Proyecto 02

Grupo 03

Integrantes: Ariel Mauro Quispe Ramos, Rodrigo Nicolás Valverde, M. Victoria Cunill

Docente: Pablo Akerman

Fecha de entrega: 22 de noviembre

Año 2021

Contenido:

* Planilla
* Desarrollo narrado

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre archivo .py** | **Tipo** | **Atributos** | **Tipo de dato** | **Métodos** | **Retorno** | **Tipo de dato retorno** | **Aclaraciones** |
| Bolillero | Clase | CATEGORIAS base\_arte base\_ciencia base\_geografia base\_historia base\_entretenimiento base\_deportes preguntas\_arte preguntas\_ciencia preguntas\_geografia preguntas\_historia preguntas\_entretenimiento preguntas\_deporte pregunta opciones | tupla diccionario diccionario diccionario diccionario diccionario diccionario lista lista lista lista lista lista string diccionario | sacar\_categoria() sacar\_pregunta(categoria) | categoria pregunta, opciones | string string, dict | - |
| Jugador | Clase | personajes nombre puntos corona personajes\_ganados | lista de tuplas string entero booleano lista | método especial \_\_str\_\_() sumar\_puntos() reiniciar\_puntaje() verificar\_corona() elegir\_personaje() guardar\_personaje\_ganado()  mostrar\_puntos\_personajes() mostrar\_personaje\_ganado() | nombre puntos puntos corona personaje, categoria personajes\_ganados None  None | string entero entero booleano string string void  Void | - |
| Tablero | Clase | - | - | jugar() si\_tiene\_corona() ciclo\_pregunta() mostrar\_pregunta() verificar\_respuesta() | cantidad\_personajes\_ganados perdio, cantidad\_personajes perdio respuesta\_usuario perdió | Entero  boolean, entero  booleano  string  booleano | - |
| arte\_ascii | modulo | PREGUNTADOS TIRANDO\_BOLILLERO FIN | string string string | - | - | - | - |
| main | programa principal - modulo | tablero boli  player1  player2  nadie\_gano  cantidad\_personajes  ganador  gano | Tablero  Bolillero  Jugador  Jugador  Booleano  Entero  Booleano  Jugador | main() verificar\_ganador() | None es\_ganador, jugador | Void boolean, Jugador | - |
| base\_arte | modulo | BASE\_ARTE | diccionario | - | - | - | En caso de ser un juego con miles de preguntas, en lugar de usar estos módulos usaríamos bases de datos |
| base\_ciencia | modulo | BASE\_CIENCIA | diccionario | - | - | - |
| base\_deportes | modulo | BASE\_DEPORTES | diccionario | - | - | - |
| base\_entretenimiento | modulo | BASE\_ENTRETENIMIENTO | diccionario | - | - | - |
| base\_geografía | modulo | BASE\_GEOGRAFÍA | diccionario | - | - | - |
| base\_historia | modulo | BASE\_HISTORIA | diccionario | - | - | - |

***Desarrollo del proyecto en forma narrada***

**Como armamos el main:**

Comenzamos el juego en el main. Con arte ascii importado del módulo arte\_ascii mostramos nombre del juego (sabemos que no sumaba puntaje, pero nos parecía que sumaba al juego)

Instanciamos 1 objeto de tipo Tablero, 1 objeto de tipo Bolillero y 2 objetos de tipo Jugador. Estos van a ser los objetos que interactuarán en nuestro juego.

Al instanciar los objetos de tipo jugador, el constructor va a pedir a cada jugador instanciado, que ingrese su nombre.

Luego vamos a tener la variable nadie\_gano, que va a ser una variable booleana inicializada en True (ya que su estado al iniciar el juego es que no hay un ganador aún) y que va a ser la variable centinela que nos va a permitir salir del while que dice: Mientras nadie gane, se ejecuta el código dentro de este while.

Cuando el valor de nadie\_gano cambia a False, se sale del while, se muestra quien ganó y cuantos personajes, arte ascii con el fin del juego y se sale del juego.

Pero mientras se esté ejecutando el while, ya que nadie\_gano es igual a True, vamos a mandarle al método jugar del objeto tablero, el jugador número1 (que es el primero en jugar según las reglas) y el bolillero. Dentro de este objeto, van a interactuar el jugador y el bolillero (lo explicamos más abajo) y nos va a retornar la cantidad de personajes que el jugador posee acumulados. Según las reglas, el jugador que gane, debe acumular 2 personajes para ganar.

Para continuar, vamos a verificar si la cantidad de personajes\_ganados por el jugador y recibidos del método jugar del Tablero, es la requerida para ganar, llamando al método verificar\_ganador de este main, mandándole la cantidad de personajes y el jugador. Nos devuelve un booleano indicándonos si el jugador es ganador o no. Si es ganador, le mandamos el jugador a la variable “gano” y cambiamos el valor del centinela “nadie\_gano” a False para salir del bucle (ya que ahora alguien si ganó).

Vamos al final del main, felicitamos al ganador, mostramos los personajes ganados que tiene el jugador. imprimimos el arte ascii fin y salimos del juego.

Ahora, si no es ganador iniciamos el proceso anterior, pero con el jugador 2. O sea, como el jugador1 perdió, perdió su turno y le toca jugar al jugador2. Por lo tanto, le mandamos al método jugar, del objeto tablero, el jugador 2 y el objeto de tipo bolillero. Este método nos retorna la cantidad de personajes acumulados del jugador2. Verificamos si cumple el requisito para ser ganador del juego. Si cumple el requisito, cambiamos el valor de la variable centinela nadie\_gano a False, cortando el bucle y finalizando el partido (felicitamos al ganador, mostramos sus personajes ganados y salimos del juego)

**Como armamos el tablero:**

Cuando el main nos envía al jugador y a boli al método jugar, ponemos a esos dos objetos a jugar. Para ello, sabemos que deben cumplirse ciertos requisitos:

* Que el jugador no pierda, ya que si pierde, pierde su turno y vuelve al main para cambiar de jugador (perdio va a ser la variable centinela).
* Que la cantidad de personajes ganados, que tenga el jugador sea menor a 2, ya que si es igual a 2, sale de este método y vuelve al main, y tenemos un ganador del juego, ya que el jugador cumple el objetivo máximo (cantidad\_personajes\_ganados va a ser la otra variable centinela).

Entonces dentro de jugar vamos a usar un ciclo while para decir que mientras no perdió (o sea si perdió es True, sale del ciclo) y la cantidad de personajes ganados sea menor a 2 (o sea, mientras se cumplan ambas condiciones):

* Vamos a pedirle al objeto Boli, que saque la categoría y nos devuelva el string categoria a Tablero (pueden ser 6 categorías y una casilla especial llamada corona).

Si es corona directamente llama al método si\_tiene\_corona, que lo explicamos más abajo, ya que también se invoca al mismo método, en el caso de que sume 3 puntos y obtenga corona.

Entonces, si la categoría NO es corona, invocamos al método ciclo\_pregunta (perteneciente a esta misma clase tablero) y le mandamos el valor de la categoría y boli para que:

Saque pregunta y opciones según la categoría (invocando al método sacar\_pregunta de boli), muestre la pregunta, las opciones y pida que ingrese respuesta al jugador (invocando al método mostrar\_pregunta de esta misma clase), y por último verifique si la respuesta es True o False. Devuelve el valor booleano al método a la variable centinela “perdió”, lo que habilitaría a continuar en caso de True o salir y cambiar de jugador en caso de False

Entonces si acertó, sumamos 1 punto de los 3 requeridos para obtener corona invocando al método sumar\_puntos del objeto de tipo jugador. Y verificamos si completó una corona, invocando al método verificar\_corona del objeto de tipo jugador y mandándole los puntos.

Si no completó corona, vuelve a iniciar el while y contínua por otra pregunta.

Si completó corona (o salió corona al sacar categoría, como explicábamos más arriba), invocamos al método “si\_tiene\_corona” de esta misma clase y le pedimos al jugador que elija un personaje, invocando al método elegir\_personaje del objeto de tipo jugador. Este método nos devuelve el personaje y la categoría correspondiente al personaje elegido. También reiniciamos el puntaje que tiene guardado el jugador, ya que al obtener corona, tanto acierte como erre a la pregunta, los puntos acumulado vuelven a 0.

En este método, invocamos al método ciclo\_pregunta, y pedimos a jugador que nos traiga la cantidad de personajes ganados que tiene. Si no gano, retornamos la variable centinela “gano” (que está en false) y la cantidad de personajes que tiene el jugador acumulados.

Si ganó, le indicamos que ganó y le mostramos el nombre del personaje. A su vez le mandamos el personaje al método guardar\_personaje\_ganado de la clase jugador, para que lo agregue a la lista de personajes que ganó el jugador y volvemos a pedir la cantidad de personajes que tiene ganados. Devolvemos la variable centinela gano (en este caso en True) y la variable centinela cantidad de personajes al método jugar.

Si gano, vuelve a iniciar el while y verifica si la cantidad de personajes ganados es menor a dos. Si es mayor, vuelve al main y finaliza el juego, si es menor ingresa al while y vuelve a iniciar todo el código con el proceso de preguntas, verificaciones etc.

Si perdio, al volver al while, no se va a ejecutar, ya que perdio es True, por lo que vuelve al main y cambia de jugador.

**Como armamos el bolillero:**

El bolillero lo que va a ser básicamente es cada vez que el tablero le pida la categoría que se va a jugar, va a elegir, usando la librería random a través de su método sacar\_categoria, un string que contiene el nombre de una categoría, de una Tupla de strings y lo devuelve al tablero.

La otra cosa que hace a pedido del tablero también, es sacar una pregunta, a partir del parámetro recibido con la categoría, y si la categoría coincide con el nombre de la categoría cargada en la clase, entonces implementando otra vez la librería random, va a invocar a la lista que contiene las claves de la base correspondiente a la categoria (en este caso las preguntas) y los valores, que son otro diccionario con las opciones de respuesta y sus valores en booleano.

Con estos datos, va a devolver a tablero, un string guardado en la variable preguntas, y un diccionario con las opciones y sus valores booleanos.

**Como armamos al jugador:**

El jugador lo que básicamente hace es ingresar su nombre, sumar puntos para la corona (la corona se llena cuando obtiene 3 puntos) y guardar personajes ganados. Los puntos los va sumando a medida que desde tablero se invoca al método sumar\_puntos, cada vez que el jugador acierta una pregunta. Luego de sumarlos, desde tablero le piden a boli que verifique si la cantidad de puntos que tiene el jugador son los requeridos para tener corona.

Entonces en caso de ganar una corona (cuando junta 3 puntos o cuando el bolillero le manda la categoría corona al tablero), el jugador elije el personaje que quiere obtener, a través del método elegir\_personaje. Dentro de este método, se le muestran los personajes y las categorías correspondientes (de una lista que contiene tuplas con los personajes y las categorías que representan dichos personajes). Y se le pide que ingrese el número del personaje que quiere ganar. Se retorna el nombre del personaje y la categoría que representa dicho personaje en dos variables (se desempaqueta)

Si vuelve a acertar la pregunta, el personaje se guarda en guardar\_personaje\_ganado, que recibe de tablero el personaje y la categoría y dentro de este mismo método, se elimina de la lista de personajes, la tupla con el personaje y su categoría.

Por último, esta clase tiene un método para mostrar los personajes ganados en tanto y en cuanto se lo invoque. En nuestro proyecto, se lo invoca al ganar el partido.