**Materia:** Fundamentos de Programación

**Trabajo:** Proyecto Final: Juego Scrabble

**Profesora:** Diana Margot Lopez Herrera

**Fecha:** 26 de mayo de 2021

**Integrantes:** Pablo Arango Castaño

John Alexander Acevedo Serna

Miguel Angel Calvache Giraldo

**CLASE PRINCIPAL**

**METODOS**

***turnoJ1***

Public static void turnoJ1()

Metodo que ejecuta el turno del jugador 1

**Parametros:**

Sin parámetros

**Retorna:**

Sin retorno

***TurnoJ2***

Public static void turnoJ2()

Metodo que ejecuta el turno del jugador 2

**Parametros:**

Sin parámetros

**Retorna:**

Sin retorno

***llenarLetrasEnTablero***

public static void llenarLetrasEnTablero()

Metodo que va colocando las letras en el tablero según la indicación del usuario

**Parametros:**

Sin parámetros

**Retorna:**

Sin retorno

**CLASE JUGADOR**

**METODOS**

***entregarLetras***

public ArrayList<Character> entregarLetras()

Metodo que entrega letras aleatorias al jugador

**Parametros:**

Sin parámetros

**Retorna:**

ArrayList<Character> que contiene las 10 letras

***entregarValores***

public ArrayList<Integer> entregarValores(ArrayList<Character> s)

Metodo que entrega los valores del puntaje de las letras

**Parametros:**

ArrayList<Character> s, con las letras

**Retorna:**

ArrayList<Integer>, con el valor de las letras

***mostrarLetras***

public void mostrarLetras()

Metodo que muestra las letras con sus respectivos valores de puntaje

**Parametros:**

Sin parámetros

**Retorna:**

Sin retornos

***valorPalabras***

public int[] valorPalabras(ArrayList<String> a)

Metodo que otorga el valor de puntaje a una cadena

**Parametros:**

ArrayList<String> a, arreglo de cadena con el cual se creare el otro arreglo para los puntajes

**Retorna:**

int[], arreglo de cadena el cual tendrá todos los valores de puntaje de la cadena de palabras

***ordenarMenoraMayor***

public int[] ordenarMenoraMayor(int[] valoresPalabras)

Metodo que ordena el arreglo valores de puntaje de menor a mayor

**Parametros:**

int[] valoresPalabras, arreglo de cadena con los valores de puntaje

**Retorna:**

int[], arreglo de cadena el cual tendrá todos los valores de puntaje de manera ordena de menor a mayor.

***mejoresValores***

public int[] mejoresValores(int[] valoresOrdenados)

Metodo que saca los diez mejores valores de puntaje

**Parametros:**

int[] valoresOrdenados, arreglo de cadena con los valores ordenados de menor a mayor

**Retorna:**

int[], arreglo de cadena el cual tendrá los 10 mejores valores de puntaje de manera ordena de menor a mayor.

***obtenerPalabrasyPuntajes***

public void obtenerPalabrasyPuntajes(ArrayList<String> palabras, int[] valoresPalabras, int[] diezMejores)

Metodo que obtiene los 10 mejores valores de puntaje de una lista de palabras con sus valores de puntaje correspondientes

**Parametros:**

- ArrayList<String> palabras, cadena con las palabras a buscar

- int[] valoresPalabras, los valores de puntaje de cada palabra

- int[] diezMejores, cadena con los 10 mejores valores de puntaje del arreglo de cadenas

**Retorna:**

Sin retornos

***ingresarPalabra***

public void ingresarPalabra(String s)

Metodo para ingresar la palabra del jugador, que muestra el valor obtenido por la jugada y el puntaje acumulado del jugador

**Parametros:**

String s, palabra escogida por el jugador

**Retorna:**

Sin retorno

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***quitarLetrasDeLaMano***

public void quitarLetrasDeLaMano(String s)

Metodo que quita la letras de la mano del Jugador

**Parametros:**

String s, palabra con las letras a retirar

**Retorna:**

Sin retorno

**CLASE DICCIONARIO**

**METODOS**

***leerDiccionario***

public void leerDiccionario(String a)

Metodo que lee el diccionario con todas las palabras del español

**Parametros:**

String a, ubicación del archivo .txt con las palabras

**Retorna:**

Sin retorno

**CLASE PALABRAS**

**METODOS**

***ConvertirAStringYPermuta***

public ArrayList<String> ConvertirAStringYPermuta(Jugador p, , ArrayList<Character> letrasEnTablero)

Metodo encargado de generar todas las posibles Variantes de las letras del jugador con una permutación (Se le hizo una modificación para que solo devuelve las variantes que contengan siempre una letra que este en el tablero)

**Parametros:**

**-** Jugador p, objeto de la clase jugador para obtener las letras

**Retorna:**

ArrayList<String>, con todas las posibles palabras con las letras obtenidas

***compararDiccionarioConPermutaciones***

public ArrayList<String> compararDiccionarioConPermutaciones(ArrayList<String> d, ArrayList<String> p)

Metodo que compara todas las posibles palabras con que si se encuentren en el diccionario

**Parametros:**

**-** ArrayList<String> d, arreglo con las palabras del diccionario

**Retorna:**

ArrayList<String> p, arreglo con todas las posibles palabras permutadas

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CLASE TABLERO**

**METODOS**

***llenarTableroInicial***

public void llenarTableroInicial()

Metodo encargado de crear una matriz que hace de tablero, de manera ordenada.

**Parametros:**

Sin parametros

**Retorna:**

Sin retorno

***imprimirTablero***

public void imprimirTablero()

Metodo encargado de mostrar el tablero por pantalla, de manera ordenada

**Parametros:**

Sin parametros

**Retorna:**

Sin retorno

***agregarLetraATablero***

public void agregarLetraATablero(String[][] a ,String casilla, char letra)

Metodo encargado de agregar las letras al tablero

**Parametros:**

-String [] [] a: Indica el tablero

-String casilla: Indica la casilla en donde agregar la letra (Forma de tres dígitos ej: 000)

-char letra: Indica la letra que se va a agregar al tablero

**Retorna:**

Sin retorno

**Reporte de Estrategias Usadas**

En uno de los métodos más importantes que nos dimos cuenta debíamos implementar una estrategia especial, fue en el encargado de entregar las letras de la mano a cada jugador. Sucedía que con el random se entregaban muy pocas vocales, pues la probabilidad de generar consonantes era mucho mayor, por esta razón tuvimos que poner un condicional que en caso de haber menos de 3 vocales añadiera una, sino la tarea de construir palabras con tan pocas vocales se haría imposible.

A la hora de imprimir la mano de letras del jugador con su respectivo puntaje, optamos por imprimir el arreglo de caracteres uno por uno separado por espacios, y en la siguiente línea hacer lo mismo con los valores de las letras; de esta manera, se ve muy organizado y claro que letras se poseen y debajo de cada una se encuentra su valor.

También, cuando imprimíamos los valores y/o ya no eran necesarios para usarse de nuevo, usamos el método clear(), para así reducir la carga del programa y permitir que funcionara correctamente al ejecutarse de nuevo estos métodos.

Para hallar las palabras con 10 mejores puntajes recurrimos a tres métodos, uno que organizara únicamente los valores de cada palabra válida en un arreglo de enteros, yendo este de menor a mayor; para después por medio de otro método obtener los valores de las 10 últimas posiciones del primer arreglo de enteros y creando así otro arreglo con los 10 mejores puntajes. Por último, usamos un método que recibía el arreglo de palabras inicial, con el arreglo de sus valores en el orden inicial y el arreglo con los puntajes de las 10 mejores palabras; con estos tres parámetros recorríamos los arreglos de valores buscando en que posición coincidían, cuando lo hacían se guardaba la palabra que había en la misma posición, que en la del arreglo inicial que coincidió con el de los 10 mejores valores. De esta manera obtenemos las 10 mejores palabras con sus respectivos puntajes.

En el método final de la clase palabras, en la cual se ingresa la palabra deseada para jugar, lo hicimos de tal manera que extrajera las letras de la palabra jugada y las quitara del arreglo de caracteres de la mano del jugador, para que se repusieran por otras distintas y se usaran las letras como en el juego real.

Para los turnos, decidimos crear un método para cada jugador, en el cual se ejecutarían todas las acciones que normalmente tendría el turno; y en el método main creamos un ciclo de 10 turnos donde se ejecutaran los métodos de cada jugador, esto con el fin de ahorrar líneas de código al escribir 10 veces lo que haría cada jugador en su turno.

Cuando empezamos con la construcción del tablero, y la implementan de los métodos que ya teníamos a este, estuvimos pensando la mejor manera de imprimir la matriz por pantalla y que quedara organizada, para esto optamos por usar una matriz de Strings y colocar 3 caracteres en cada posición de la matriz; estos caracteres por defecto serían el numero de la casilla en la que se encuentran.

Para que todos tuvieran los 3 caracteres, se creó un ciclo que, si detectaba que la casilla era menor a 100, añadiera dos ceros antes del número, y si era mayor a 10 y menor a 100, añadiera un cero.

La razón por la que optamos por el uso de los 3 caracteres fue porque, al imprimir la matriz, todas las columnas fueran rectas, ya que si solo se imprimía el numero de la casilla iban a quedar descuadradas las columnas, esto debido a que en las primeras casillas solo habría uno o dos dígitos, mientras que en las últimas habría de tres, lo que haría unas filas más largas que otras.

Al numerar cada casilla del tablero le facilitábamos mucho el trabajo a los jugadores, pues ellos decidirían en que posición colocar la palabra con mucha facilidad. También, para que el tablero mantuviera su tamaño, cuando se colocara una letra en una casilla, esta se colocaría en medio de dos espacios, de esta manera cada casilla conservaría su tamaño de 3 caracteres.

**Manual de Usuario para la primer entrega del juego Scrabble en Java**

**Reglas:**

* El juego está hecho para 2 jugadores.
* Hay 3 turnos en total durante una partida.
* A cada jugador se le entregan 10 letras al azar, con mínimo 3 vocales.
* Cada letra tiene un valor fijo asignado, los valores van así:
  + *1 punto*: **A**, **E**, **O**, **I**, **S**, **N**, **L**, **R**, **U**, **T**
  + *2 puntos*: **D**, **G**
  + *3 puntos*: **C**, **B**, **M**, **P**
  + *4 puntos*: **H**, **F**, **V**, **Y**
  + *5 puntos*: **Q**
  + *8 puntos*: **J**, **X**
  + *10 puntos*: **Z**
* En el tablero siempre va a haber una palabra al empezar.
* El jugador debe formar una palabra por turno haciendo uso de una de las letras ya puestas en el tablero. Con base en esta palabra se dan los puntos teniendo en cuenta el valor de cada letra usada.
* La palabra que forme el jugador debe ser agregada al tablero, de manera que quede anidada con la letra que se usó para su formación.
* Después de cada turno se le entregan las letras faltantes al jugador, hasta que este complete las 10.
* Gana quien consiga el mayor puntaje después de los 3 turnos.

**Instrucciones básicas:**

Al ingresar al juego saldrá un mensaje de bienvenida y a continuación se le pedirá al usuario que ingrese el nombre de los jugadores; se mostrara el tablero en el que se va a jugar, cada casilla de este estará marcada con un número que corresponde a la posición, después empieza el juego. Para cada turno se le mostraran al jugador las letras que le tocaron en su mano con sus respetivos valores en la parte de abajo; durante este turno se le mostraran las 10 mejores jugadas que puede hacer y con esto el jugador debe digitar las letras que va a usar para su turno, y decir en que posiciones del tablero las va a colocar, de esta manera se calcularan los puntos del turno.

Cada turno se le reponen las letras usadas al jugador y se le vuelven a mostrar, para que este pueda ver que hay en su mano de letras.

Luego de haber terminado los tres turnos de cada jugador se mostrarán los resultados, poniendo el nombre del jugador y los puntos que logró durante la partida. Y así se termina el juego.