PRÁCTICA PROGRAMACIÓN I 2022-2023

20302 – PROGRAMACIÓ – INFORMÀTICA I
MIQUEL MASCARÓ PORTELLS

AUTORS

PAU TONI BIBILONI MARTÍNEZ | 46391835-F
MARC LINK CLADERA | 45692714-V

ÍNDICE

1. INTRODUCCIÓN

El objetivo de la práctica final del curso es consolidar los conocimientos que hemos ido adquiriendo en clase. Especialmente, tenemos que demostrar a través del desarrollo de un proyecto completo, el uso de los conceptos siguientes: planteamiento algorítmico, tipo de datos, claridad del código, estructuras de control, tratamiento secuencial, subprogramas, estructuras de datos, programación orientada a objetos, ficheros de texto y diseño descendente, además de trabajar las competencias genéricas de la asignatura: análisis y síntesis, organización y toma de decisiones y comunicar conceptos propios de la informática de manera escrita. En esta práctica hemos hecho uso herramientas que vistas a lo largo del curso, entre estas: el lenguaje de programación Java, el entorno integrado de desarrollo NetBeans, la clase LT, la clase Palabra y otras herramientas que han ido a implementando en nuestros programas.

El Wordle es un juego en línea que tiene como objetivo adivinar una palabra de cinco letras en seis intentos. Inventado en 2021 por Josh Wardle, se popularizó a través de las redes sociales a inicios del 2022, cuando la versión original en inglés fue adaptada al catalán y al castellano. La mecánica de juego es sencilla: para encontrar una palabra objetivo, el usuario introduce, como mucho seis veces, una palabra de cinco letras. El sistema marca con colores las letras introducidas: el fondo gris representa las letras que no se encuentran en la palabra buscada, el fondo amarillo representa las letras que sí que se encuentran en la palabra buscada, pero no en la posición correcta y el fondo verde representa las letras que se encuentran en la palabra buscada y en la posición correcta. De esta forma con las pistas obtenidas mediante la palabra introducida anteriormente el usuario debe pensar cuál podría ser la palabra objetivo.

Además de los requisitos esenciales, hemos incorporado los siguientes aspectos adicionales a la práctica:

- Mostrar las letras de las palabras en mayúsculas, como el juego original. Hemos introducido este aspecto ya que le daba un toque más característico debido a que el wordle original muestra las palabras en mayúscula.

- Dar a elegir al usuario con que idioma quiere jugar. Decidimos incorporar esto ya que le da mucha versatilidad al juego. Dado que ha sido un aspecto que hemos tenido en cuenta desde el principio, ha sido sencillo implementarlo.

- Modo jugador contra jugador: dos jugadores tienen que intentar adivinar la misma palabra objetivo (en dos ordenadores diferentes). Esto lo conseguimos mediante una seed, esta sed indicará al programa que palabra poner como objetivo, de esta forma dos jugadores pueden jugar a encontrar la misma palabra.

2. DISEÑO WORDLE

En primer lugar tenemos el main desde donde llamaremos a la clase menú y hacemos un breve comentario para explicar de qué se trata el juego.
En la clase menú usamos la clase LT distribuida durante el curso, esta nos permitirá obtener entradas por teclado, necesarias para poder acceder a los siguientes apartados del programa. En el constructor llamamos a menuPrincipal para que cada vez que instanciemos una variable de la clase menu directamente salte la interfaz del juego. Una vez puestas todas las opciones del juego (1.Jugar, 2.Estadísticas y s.Salir) pediremos que elija la opción. cuando introduzca por teclado la opción mediante el método 'choose' lo enviaremos a otras clases o subprogramas. Cabe destacar que cada vez que pedimos una entrada por teclado, está, la validamos con sus respectivas características de validación, en este caso revisamos que los datos introducidos sean 1, 2 o s , si no es así, mediante un while saldrá un error y te volverá a preguntar la opción de nuevo.
El método 'choose' tiene un switch al cual le damos 3 opciones (caso 1, caso 2 y el default que significa que si no se elige ninguna de las opciones anteriores el juego se acaba y nos muestra un print diciendo "Adeu", sabemos que entrara al caso default si o si cuando introduzca una 's' ya que antes de entrar en el switch hemos validado que solo pueda entrar un 1 un 2 o una 's').

● CASO 1 DEL SWITCH

En este caso llamamos a nuestra clase Joc, las variables que aparecen son:

- 'LONGITUD_PALABRAS' de tipo int, le damos el valor de 5, que es el tamaño de las palabras por defecto.
- 'LONGITUD_RONDAS' de tipo int, esta nos indicará la cantidad de rondas que se pueden jugar.
- 'lt' de tipo LT para poder introducir datos por teclado.
- 'estadísticas' de la clase Estadísticas que nos permite generar e imprimir estadísticas.
- 'palabrasMetidas' de tipo Paraula[], esta será una array de palabras, en cada casilla de la array guardaremos cada una de las palabras para que posteriormente poemas imprimirlas
- 'nom' de tipo Paraula que guardará el nombre y de esta forma después podremos usar el método toString de la clase Paraula para poder imprimir el nombre.
- 'palabracomparar' de tipo Paraula, en esta variable se guardará la palabra que introduzca por teclado y mediante la clase palabra podemos hacer las validaciones de forma más sencilla.
- 'palabraobjetivo' de tipo Paraula, donde meteremos la palabra objetivo conseguida de forma random.

- ‘dioma’ de tipo char que guardara un solo carácter (‘e’, ‘a’ o ‘c’)
- ‘seed’ de tipo Integer a esta variable desde el principio le asignamos el valor de -1 ya que esta si le asignamos valor será con un random. Como los random devuelven números positivos, si a esta variable le damos un valor negativo significa que su valor no se ha consiguió mediante el random.

Estas serían las variables globales usadas en esta clase.

Al constructor de la clase Joc le pasamos por parámetro una booleana ‘multijugador’ (desde la clase Menú) que nos indicará si quiere jugar en modo multijugador o en modo individual.

Este método nos hará una serie de preguntas para personalizarlo al estilo del jugador, entre estas preguntas encontramos el nombre, el idioma, y si queremos o no una pista (la pista nos dará uno de los caracteres de la palabra objetivo). posteriormente ya entrara a jugar. Todo el juego se basa en un bucle for que tendrá 6 iteraciones (el número de LONGITUD_RONDAS). Dentro de este bucle iremos validando las entradas, comparando con la palabra objetivo y guardando las palabras introducidas en la array de Paraules.

La palabra aleatoria la conseguimos mediante un método que hemos creado llamado PalabraAleatoria la cual nos proporcionarà una array, en la que habrá una palabra situada en una línea random en el caso de que esté jugando el el modo individual o una palabra que estará situada en la línea que corresponde con la seed introducida anteriormente. Nos dimos cuenta que para extraer una palabra random teníamos que hacer dos lecturas al txt, en la primera lectura contamos el numero de lineas (porque el número de líneas varía dependiendo del idioma del diccionario) que hay y en la segunda lectura extraemos la palabra situada en la línea número ‘x’ (siendo ‘x’ el número obtenido mediante el random o el número que pasamos como seed).

Para la lectura y escritura tenemos dos clases FI y FO, clases a las que prácticamente no les hemos hecho cambios, estas han sido aportadas por el profesor (lo único que hemos añadido ha sido un método en la clase Fi para obtener el número de líneas de un fichero).

De nuevo en la clase Joc, si el jugador elige la opción multijugador, el juego nos preguntará por nuestra seed (en este volvemos a hacer una validación de datos por si el usuario introduce un enter), una vez introducida la seed empezara el juego (aparecerá una interfaz con nuestro nombre y el idioma seleccionado) donde nos preguntará por nuestra jugada y tendremos que responder por vía teclado gracias a la clase LT (en este caso hemos hecho una validación de datos para comprobar que la palabra exista en el diccionario del idioma correspondiente (mediante la clase Paraula) y si tiene una longitud de 5 letras).
La mecánica del juego es la siguiente: primero hacemos un subprograma que nos muestre las palabras introducidas anteriormente para que el usuario pueda ver la palabra que a introducido y que letras se han dibujado o no, es decir, si una letra esta en la misma posicion que la palabra objetivo se pintara de color verde, si una letra esta en la palabra objetivo pero en distinta posicion se pintara de color amarillo y si una letra no esta en la palabra objetivo no se pintara de ningun color gracias al metodo printPalabra (este metodo

lo que hace es imprimir caracter a caracter el color que toca gracias a la array de ints que le pasamos por parámetro(es una array de ins porque hemos codificado los colores como números (1 = verde; 2 = amarillo y 3 = gris) que va junto al metodo imprimir (este metodo lo que hace es codificar la array de colores), luego hacemos otro subprograma que nos compare la palabra objetivo y la palabra introducida, si estas dos palabras son exactamente iguales nos dirá que hemos acertado y nos llevará al menú principal, si no son iguales y se acaban los intentos permitidos, el juego nos dirá cual era la palabra objetiva y nos llevara nuevamente al menú principal.

Para el tratamiento de las palabras hemos utilizado la clase palabra:

- El método existAlDic() nos permite validar si la palabra que introduce el usuario existe en el diccionario, este método lo que hace es ir leyendo línea a línea, guardando cada línea en una palabra, gracias al método 'igual' podremos saber si existe o no (si encontramos alguna palabra que es igual significa que la palabra existe en el diccionario).

- El método toMayusculas() nos convierte las letras minúsculas en mayúsculas.

- El método toMinusculas() nos convierte las letras mayúsculas en minúsculas.

- El método getEspecificChar() lo que nos permite es mediante un número que le pasamos por parámetro, obtener el carácter que está en esa posición en la array de la palabra.

- El método igual() lo que hace es comprobar si dos palabras son iguales, comprobando en este el tamaño de la palabra y si los caracteres coinciden entre las dos palabras pasadas por parámetro.

- El método posarArray() lo que hace es recibir una array por parametro y agregarla a la array 'pal' de la clase.

• CASO 2 DEL SWITCH

Cuando entramos en este switch es porque el jugador quiere ver las estadísticas de las partidas anteriores, para ello lo que haces es llamar al método imprimir de la clase 'Estadisticas' para que nos imprima el txt que contiene las estadísticas('estadísticas.txt') recopiladas y almacenadas durante la partida. La información en este fichero está guardada de la siguiente manera:

FECHA Y HORA # IDIOMA # NOMBRE DEL USUARIO #PALABRAS INTRODUCIDAS # PALABRA OBJETIVO

- **CASO 3 DEL SWITCH**

Este caso se activará únicamente si le damos a la opción salir (s), una vez pulsada esta opción el juego se acabará y nos despediremos del usuario .

BIBLIOGRAFÍA

Wordle definición.