JUEGOS ARITMETICOS

Programación III

Integrantes:

- Alan Barraza
- Giovanni Sia

Universidad Nacional de General Sarmiento

JUEGOS ARITMETICOS

En este trabajo se diseñó un juego, el cual consiste en una grilla de 4x4 con valores a los costados, en la que el jugador debe adivinar qué número del 1 al 9 debe poner en cada casillero de la grilla, de modo tal que la suma de los números de cada fila sea igual al valor dado, y que la suma de los números de cada columna sea igual al valor dado.

El juego inicia ejecutando cualquiera de las clases de interfaz.

Para implementar este juego se construyeron diversas clases las cuales están separadas por paquetes.

Paquete "interfaz":

- **InterfazMenu:** Contiene el menú que le da la bienvenida al jugador antes de empezar el juego y en la cual debe ingresar su nombre.
- **InterfazJuego:** Muestra la ventana de juego, donde se ven la grilla, el cronometro, y los controles del juego.
- **InterfazGanador:** Muestra el menú en caso de que el jugador complete correctamente la grilla, con posibilidad de reiniciar el juego, salir o volver al menú.
- **InterfazPuntaje:** Muestra los puntajes de los jugadores.

Paquete "logica":

- **Tablero**: Contiene los valores de las casillas, además de métodos para controlarlos.
- **Puntaje:** Clase que, a través de manejo de archivos, guarda el puntaje de los jugadores (Nombre del jugador y tiempo que tardó).
- **Cronómetro:** Toma el tiempo que le toma al jugador completar el juego.

A la hora de implementar estas clases, nos encontramos con algunas dificultades:

- Se pensaron diversas formas de implementar la grilla del juego, entre las cuales la primera fue diseñar cada casilla de la grilla individualmente, y nos encontramos con que se repetía código constantemente. La otra manera fue diseñarlo a través de una tabla, pero esa idea fue descartada debido a la dificultad que traía manejar los valores de las casillas. La implementación que nos resultó mejor fue la de manejar una matriz de JLabels.
- Otro problema fue al intentar guardar los puntajes ya que no se acumulaban los nombres de los ganadores. Se soluciono a través de la inicialización del objeto FileOutputStream, al cual le faltaba definir en "true" si el objeto hace un append al texto leído o no. Además faltaba el método flush(), el cual permite la modificación del texto correctamente.
- Se presentó otra problemática, la cual fue tener que abrir una nueva ventana. Como no se logró mediante clases Application Window, se usaron clases Jframe. A través de estas clases se creaba un objeto, el cual es la interfaz que se quiere ver, y se elimina la actual por cada vez que se quiere cambiar de ventana.
- Además, en la generación de soluciones que pudiesen ser irresolubles se crea un nuevo tablero en el cual se le agrega al azar números del 1 al 9 para luego sumar cada fila y columna y así armar las soluciones.

- El cronómetro presentaba un problema, el cual consistía en que se movía constantemente, debido a que en ciertos momentos los segundos tenían una unidad, en vez de dos. Entonces, a través de ifs, se agregaba un “0” como String al otro String de datos.
- En la matriz del tablero, se dificulto el hecho de manejarlas sin tener en cuenta las soluciones del juego en la clase “Tablero”, así que en la clase “InterfazTablero” se usó una matriz con las soluciones, y en la clase “Tablero” sin las mismas.

Estas clases presentan métodos que definen su comportamiento, y su interacción con las demás clases. Los métodos más importantes de las mismas son los siguientes:

InterfazTablero()

- **private void manejoValorCasilla(int f, int c, int click):** Recibe como parámetros la posición de la matriz[f][c] de JLabels, y dependiendo de qué botón del mouse se apriete, se cambia el valor de la casilla.

Tablero()

- **public void valoresDefinidos():** Define los valores a los que tendrá que llegar el jugador a través de la suma de las casillas. Creando un tablero auxiliar, y asignándole valores aleatorios del 1 al 9. Luego, se suman estos valores y se van agregando en listas donde se van a guardar como los resultados para cada fila y columna.
- **public boolean gano():** Inicializa una variable boolean en true. Luego, va sumando las casillas tanto de las filas como de las columnas. Si todas las sumas dan como resultado el valor asignado para esa fila o columna, la variable se mantiene en true, de lo contrario cambia a false. Al final, se devuelve la variable, que será true si todas las filas y columnas dan como resultado su valor asignado, o false si alguna no lo hace.

Puntaje()

- **public static void registrarPuntaje(String datos):** Accede a un archivo .txt (o crea uno en caso de que no exista), en el cual se escriben el tiempo que tardaron los jugadores en terminar sus partidas.
- **private static void limpiarArchivo(String archivo):** Limpia el archivo (borra todas sus líneas) para evitar que se repitan puntajes

Cronometro()

- **public void sumarTiempo():** Suma el tiempo del cronómetro utilizando las 4 variables del objeto (h,m,s y cs).
- **private void actualizarDatos(int h, int m, int s, int cs):** Actualiza los datos del cronómetro, el cual es un String. (Evita que haya movimiento en el string por la falta de 0).

En conclusión, después de haber finalizado con el trabajo, y a pesar de no haber podido contar con nuestra compañera Tamara Fernández, se obtuvo el resultado que se deseó, implementando en el mismo todo lo aprendido en clases y a su vez implementando elementos aprendidos a partir de las clases virtuales, y documentación/tutoriales vistos en internet.