Manual Técnico

Se crearon varios métodos para poder organizar un poco el programa:

```
1  package D3r3_k;
2
3  import java.util.InputMismatchException;
6
7  public class Main {
8
9      public static int partida=0;
10     static Historial[] histo = new Historial[10];
11
12     public static void main(String[] args) {
15
16     // METODOS
17     static void menu() {
17
18     public static void nuevoJuego(String nombre, int edad, int alto, int ancho) {
18
19     static void itmRndm1(int rangoAlto, int rangoAncho, String[][] tablero) {
38
39     static void itmRndm2(int rangoAlto, int rangoAncho, String[][] tablero) {
57
58     static void itmRndm3(int rangoAlto, int rangoAncho, String[][] tablero) {
76
77     static void guardar(String nombre, int edad, int puntaje, int movimientos) {
84
85     static void Stats(String nombre, int puntaje, int movimientos) {
93
94     public static void infoMov() {
101    public static void win() {
107    public static void lose() {
113
114
115 }
116
```

En la clase Main se crearon, una variable global y un arreglo global.

Public static int partida=0;

Se almacena la cantidad de partidas jugadas. Al ganar, perder o parar el juego, la variable partida aumenta su valor en 1 para tener un conteo en la posicion del arreglo.

Static Historial[] histo = new Historial[10];

Se crea un arreglo con una capacidad maxima de 10 para almacenar datos, este servira para almacenar los datos finales del juego.

Una vez creadas estas varibes globales, creamos un metodo/funcion menu() y la llamamos a nuestra funcion main()

```
public class Main {

    public static int partida=0;
    static Historial[] histo = new Historial[10];

    public static void main(String[] args) {
        menu();
    }

    // METODOS
```

Empezamos creando una variable Scanner.
Inicializamos las variables que pediremos para la creación de una nueva partida, nombre, edad, ancho, alto, opción y creamos un booleano que se encargara de parar nuestro ciclo while().

Imprimimos el mensaje para que el jugador escoja una opción.

```
17    static void menu() {
18
19        Scanner sc = new Scanner(System.in);
20        String nombre="";
21        int edad=0,ancho=0,alto=0;
22        boolean terminar = false;
23        int opcion=0;
24
25        while(!terminar) {
26            System.out.println(
27                    "\n|--------------------------------------|\n"
28                     +"|          Selecciona una opcion       |\n"
29                     +"|--------------------------------------|\n"
30                     +"|     1. Jugar                         | \n"
31                     +"|     2. Historial                     |\n"
32                     +"|     3. Salir                         |\n"
33                     +"|--------------------------------------|");
```

Utilizamos un try{} catch(){ } que si el usuario escribe una letra, símbolo o palabra en opciones donde se le solicitan caracteres números, este nos imprima un error y vuelva a repetir el proceso de iniciar una nueva partida.

Utilizamos un Switch, para definir las condiciones del menú, dentro del caso 1, creamos nuevamente un booleano que nos servirá para terminar el bucle que se encarga de pedir los datos al jugador y preguntamos los datos del jugador.

```
35        try {
36            opcion=sc.nextInt();
37                switch(opcion) {
38                case 1:
39                    boolean exit= false;
40                    do {
41                        try {
42                            System.out.println(
43                                    "|--------------------------------------|\n"
44                                   +"|              NUEVA PARTIDA           |\n"
45                                   +"|--------------------------------------|\n"
46                                   + "Escribe tu nombre...");
47                            nombre = sc.next();
48                            System.out.println("Escribe tu edad...");
49                            edad = sc.nextInt();
50
51                            System.out.println(
52                                    "|--------------------------------------|\n"
53                                   +"|              NUEVO TABLERO           |\n"
54                                   +"|--------------------------------------|\n"
55                                   + "> Escribe la altura del tablero (Min. 8)...");
56                            alto = sc.nextInt();
57                            boolean dimension1 = false, dimension2 = false;
58                            while (!dimension1) {
59                                if(alto >=8) {
60                                    System.out.println("> Escribe el ancho del tablero (Min. 8)...");
61                                    ancho = sc.nextInt();
62
63                                    while (!dimension2) {
64                                        if(ancho<8) {
65                                            System.out.println("El ancho tiene que ser mayor a 8...");
66                                            ancho = sc.nextInt();
67                                        }else {
68                                            dimension1=true;
69                                            dimension2=true;
70                                            nuevoJuego(nombre, edad, alto, ancho);
71                                        }
72
73                                    }
74                                }else {
75                                    System.out.println("La altura tiene que ser mayor a 8...");
76                                    alto = sc.nextInt();
77                                }
78                            }
79                            exit = true;
80                        } catch (InputMismatchException e) {
81                            System.out.println("\n\n > Escribe un dato correcto... < \n\n");
```

Colocamos las condicionales if(alto>=8) y if(ancho<8) para definir un tamaño mínimo al tablero.

Al cumplir todas las condiciones, este llamara a la función nuevoJuego, enviándole los parámetros: nombre, edad, alto, ancho.

Una vez enviado los datos a la función nuevoJuego, este empieza a crear la partida, generando el tablero, sus muros, muros internos aleatorios y sus espacios vacíos.

```java
public static void nuevoJuego(String nombre, int edad, int alto, int ancho) {
    int nuevoAlto = alto+2;
    int nuevoAncho = ancho+2;
    String[][] tablero = new String[nuevoAlto][nuevoAncho];
    int rangoAlto=alto-1,rangoAncho=ancho-1;
    Scanner op = new Scanner(System.in);

    Random rnm = new Random();

    // CRACION DEL TABLERO
    for (int f = 0; f < nuevoAlto; f++) {
        for (int c = 0; c < nuevoAncho; c++) {
            if(f==0 || f == nuevoAlto-1) {
                tablero[f][c] = "*";
            }else if(c==0 || c==nuevoAncho-1) {
                tablero[f][c] = "*";
            }else {
                tablero[f][c] = " ";
                if(rnm.nextBoolean()) {
                    tablero[f][c] = "*";
                }
                if(rnm.nextBoolean()) {
                    tablero[f][c] = " ";
                }
                if(rnm.nextBoolean()) {
                    tablero[f][c] = " ";
                }
            }

            if(f==(nuevoAlto/2)) {
                tablero[f][c] = " ";
            }else if(f==(nuevoAlto-2)/2) {
                tablero[f][c] = " ";
            }


        }
    }
```

Creamos 3 variables más, la posición del jugador a lo alto y ancho del tablero y un boleando que se encargara de terminar es ciclo si el jugador apareció en un espacio vacío.

Definimos la posición del jugador con dos números aleatorios, creamos un ciclo que nos posicione al jugador en un espacio vacío dentro del tablero y termine el ciclo, de lo contrario, vuelva a generar dos números aleatorios hasta que se cumpla la condición anterior.

```
int playerAlto,playerAncho;
boolean aparece = false;

itmRndm1(rangoAlto, rangoAncho, tablero); // ITEM 1 -> #
itmRndm2(rangoAlto, rangoAncho, tablero); // ITEM 2 -> $
itmRndm3(rangoAlto, rangoAncho, tablero); // ITEM 3 -> @


playerAlto = (int)(rnm.nextDouble()*rangoAlto+1);
playerAncho = (int)(rnm.nextDouble()*rangoAncho+1);

while(!aparece) {
    if(tablero[playerAlto][playerAncho].equals(" ")) {
        tablero[playerAlto][playerAncho] = "V";
        aparece = true;
    }else {
        playerAlto = (int)(rnm.nextDouble()*rangoAlto+1);
        playerAncho = (int)(rnm.nextDouble()*rangoAncho+1);
    }
}

// IMPRESION DEL TABLERO
int puntaje=10;
int movimientos=0;
Stats(nombre,puntaje,movimientos);
for (int f = 0; f < nuevoAlto; f++) {
    for (int c = 0; c < nuevoAncho; c++) {
        System.out.print(tablero[f][c]);
    }
    System.out.println("");
}
infoMov();
```

Creamos dos variables más, que nos servirán para almacenar la puntuación y movimientos que ha realizado el jugador y luego imprimimos el tablero.

Creamos tres variables, infoMovimiento, la que se encarga de parar el ciclo cuando este sea verdadero, opcionMovimiento para pedir un movimiento al jugador y posMid para cuando el jugador entre a un limite del muro y este este vacío aparezca del otro lado del tablero.

Pedimos la opción de movimiento al jugador y lo convertimos en minúscula, seguido creamos un Switch para las condiciones del movimiento.

Si en caso que el jugador haya comido un item, ya sea @, $ o #, este tendrá el mismo funcionamiento, su posición actual será remplazada por un espacio, la posición a la que se moverá se remplazara con el jugador, sumara o restara los puntos según el item y llamara a la función itmRndm1,2,3 según el item que se coma (item1=#, item2=$, item3=@) y aumentara la cantidad de movimientos +1, en caso que el jugador tenga un muero en la posición a la que se quiere dirigir, este no se moverá y solo aumentara sus movimientos, de lo contrario, si la posición a la que se moverá es un espacio, este se moverá libremente y finalmente imprimirá nuevamente el tablero

```
190         // EVENTOS DE MOVIMIENTOS
191         boolean infoMovimiento = false;
192         String opcionMovimiento = "";
193         int posMid = nuevoAncho-1;
194 
195         do {
196             opcionMovimiento = op.next().toLowerCase();
197             switch (opcionMovimiento) {
198             case "w":
199                 if(tablero[playerAlto-1][playerAncho] == "#") {
200                     tablero[playerAlto][playerAncho] = " ";
201                     tablero[playerAlto-1][playerAncho] = "V";
202                     playerAlto-=1;
203                     puntaje-=10;
204 
205                     itmRndm1(rangoAlto, rangoAncho, tablero);
206 
207                     movimientos+=1;
208                 }else if(tablero[playerAlto-1][playerAncho] == "$") {
209                     tablero[playerAlto][playerAncho] = " ";
210                     tablero[playerAlto-1][playerAncho] = "V";
211                     playerAlto-=1;
212                     puntaje+=15;
213 
214                     itmRndm2(rangoAlto, rangoAncho, tablero);
215 
216                     movimientos+=1;
217                 }else if(tablero[playerAlto-1][playerAncho] == "@") {
218                     tablero[playerAlto][playerAncho] = " ";
219                     tablero[playerAlto-1][playerAncho] = "V";
220                     playerAlto-=1;
221                     puntaje+=10;
222 
223                     itmRndm3(rangoAlto, rangoAncho, tablero);
224 
225                     movimientos+=1;
226                 }else if(tablero[playerAlto-1][playerAncho] == "*") {
227                     movimientos+=1;
228                 }else {
229                     tablero[playerAlto][playerAncho] = " ";
230                     tablero[playerAlto-1][playerAncho] = "V";
231                     playerAlto-=1;
232                     movimientos+=1;
233                 }
                Stats(nombre, puntaje, movimientos);
                for(int f=0;f<nuevoAlto;f++) {
                    for(int c=0;c<nuevoAncho;c++) {
                        System.out.print(tablero[f][c]);
                    }
                    System.out.println("");
                }
                break;
```

Al llamar a la funcion itmRndm1,2 o 3, este recibira el rango alto y ancho del tablero y el tablero, luego generara dos numeros aleatorios definiendo la posicion de dicho item, luego generara un ciclo donde si la posicion del objeto en el tablero esta vacia, se posicionara, de lo contario generara dos numeros aleatorios y volvera a empezar el ciclo hasta que se cumpla la condicion. (asi con los 3 objetos)

```
418     static void itmRndm1(int rangoAlto, int rangoAncho, String[][] tablero) {
419         int item1Alto, item1Ancho;
420         Random rnm = new Random();
421         boolean aparecio = false;
422 
423         item1Alto = (int)(rnm.nextDouble()*rangoAlto+1);
424         item1Ancho = (int)(rnm.nextDouble()*rangoAncho+1);
425 
426         while(!aparecio) {
427             if(tablero[item1Alto][item1Ancho].equals(" ")) {
428                 tablero[item1Alto][item1Ancho] = "#";
429                 aparecio = true;
430             }else {
431                 item1Alto = (int)(rnm.nextDouble()*rangoAlto+1);
432                 item1Ancho = (int)(rnm.nextDouble()*rangoAncho+1);
433             }
434         }
435 
436     }
437 
438     static void itmRndm2(int rangoAlto, int rangoAncho, String[][] tablero) {
456 
457     static void itmRndm3(int rangoAlto, int rangoAncho, String[][] tablero) {
```

En caso que el jugador, gane, pierda o pare el juego presionando m, este llamara a la funcion/metodo guardar, definiendo los parametros que utilizara, luego termina el ciclo de los movimientos y finalmente regresa al menu principal.

```
390                 break;
391             case "m":
392                 guardar(nombre,edad,puntaje,movimientos);
393                 infoMovimiento=true;
394                 menu();
395                 break;
396 
397             default:
398                 infoMov();
399                 break;
400             }
401 
402             if(puntaje>=100) {
403                 guardar(nombre,edad,puntaje,movimientos);
404                 win();
405                 menu();
406                 infoMovimiento = true;
407             }else if(puntaje<=0) {
408                 guardar(nombre,edad,puntaje,movimientos);
409                 lose();
410                 menu();
411                 infoMovimiento = true;
412             }
413 
414 
415         } while (!infoMovimiento);
```

En la función guardar, se le solicitara los datos finales de la partida: nombre, edad, puntaje, movimientos. Luego se llamará al constructor de la clase Historial, almacenando los datos en la variable h y mandándole los parámetros que se solicitaron anteriormente, luego se hace una validación donde preguntamos si el arreglo en la posición del numero de partida no es nulo, este nos inserte (los datos almacenados anteriormente en la variable h) al arreglo en dicha posición. Utilizando partida como el iniciador del arreglo y al finalizar la función, aumentamos en 1 la variable partida.

```
476     static void guardar(String nombre, int edad, int puntaje, int movimientos) {
477         Historial h = new Historial(nombre, edad, puntaje, movimientos);
478         if(histo[partida]==null) {
479             histo[partida]=h;
480         }
481         partida++;
482     }
483
```

En la clase Historial, se crearon variables privadas donde se almacenarán los datos que se solicitarán a través de un constructor solicitando los datos, nombre, edad, puntaje y movimientos de la clase principal.

Se crea una función verHistorial que nos permite llamar a todos los datos del arreglo Historial y mostrarlos.

```
1  package D3r3_k;
2
3  public class Historial {
4
5      private String nombre;
6      private int edad;
7      private int puntaje;
8      private int movimientos;
9
10     public Historial(String _nombre, int _edad, int _puntaje, int _movimientos) {
11         this.nombre = _nombre;
12         this.edad = _edad;
13         this.puntaje = _puntaje;
14         this.movimientos = _movimientos;
15     }
16
17
18     void verHistorial() {
19         System.out.println("|-----------------------------------------------------");
20         System.out.println("| Nombre: "+nombre);
21         System.out.println("| Edad: "+edad);
22         System.out.println("| Puntaje: "+puntaje);
23         System.out.println("| Movimientos: "+movimientos);
24         System.out.println("|-----------------------------------------------------");
25     }
26
27 }
28
```

Al presionar la opción 2, se crea un ciclo for, que empieza desde 0, hasta la longitud del arreglo historial, mostrando todos los datos almacenados, si estos no son nulos.

```
                break;
            case 2:
                System.out.println("\n\n"+
                        "|--------------------------------------|\n"
                       +"|               HISTORIAL              |\n"
                       +"|--------------------------------------|\n");
                for (int i = 0; i < histo.length; i++) {
                    if(histo[i]!=null) {
                        histo[i].verHistorial();
                    }else {
                        System.out.print(" ");
                    }
                }
                menu();
                terminar=true;
                break;
            case 3:
                System.out.println("\n\n"
                       +"|--------------------------------------|\n"
                       +"|                ADIOS                 |\n"
                       +"|--------------------------------------|\n");
                terminar=true;
                break;
            }
        } catch (InputMismatchException e) {
            System.out.println("\n > Solo se aceptan numeros <\n");
            sc.next();
        }
    }

}
```

Al presionar la opción 3, se imprime un mensaje y finaliza el programa.

Se agregaron funciones extra para mostrar información más dinámica y ordenada.

```
484     static void Stats(String nombre, int puntaje, int movimientos) {
485         System.out.println(
486                 "|--------------------------------------\n"
487                +"| > Nombre: "+nombre+"          \n"
488                +"| > Puntaje: "+puntaje+"         \n"
489                +"| > Movimientos: "+movimientos+"\n"
490                +"|--------------------------------------");
491     }
492
493     public static void infoMov() {
494         System.out.println(
495                 "|---------------------------------|\n"
496                +"| Movimientos: A-S-D-W            |\n"
497                +"| Salir: M                        |\n"
498                +"|---------------------------------|\n");
499     }
500     public static void win() {
501         System.out.println("\n\n"+
502                 "|---------------------------------|\n"
503                +"|               GANASTE           |\n"
504                +"|---------------------------------|\n");
505     }
506     public static void lose() {
507         System.out.println("\n\n"+
508                 "|---------------------------------|\n"
509                +"|               PERDISTE          |\n"
510                +"|---------------------------------|\n");
511     }
512
```