Documentación Batalla Naval

La siguiente documentación consta de la implementación de los algoritmos de como funciona el juego llamado Batalla Naval, el juego se presenta a nivel de pseudocódigo para posteriormente ser transcrito a un lenguaje de programación, en este caso se utilizará el lenguaje java y se implementará interfaz de usuario con la librería swing, el funcionamiento de cada clase y métodos que contienen las clases se definen a continuación.

Clase MenuPrincipal

Se encarga de mostrar al usuario la parte gráfica del juego, después de registrar el nombre del jugador, se encarga mostrar el menú de opciones a través de botones, seguidamente según el botón presionado se redirigirá a la opción indicada.

```
Inicio Proceso MenuPrincipal

    /*Métodos se explican a continuación*/

FinProceso
```

El método **mostrarVentana()** se encarga de mostrar la ventana principal del juego, seguidamente llama al método correspondiente para registrar el nombre del jugador y al momento de dar click en el botón Registrar, guarda el nombre o nick ingresado por el usuario y muestra el menú principal del juego, inicialmente la opción Iniciar Partida estará bloqueada debido a que primero debe de existir mapas para jugar.

```
InicioProceso mostrarVentana()
    Var nombre

    Llamar método mostrarVentanaPrincipal()
    Llamar Proceso registrarJugador()
FinProceso
```

El método **registrarJugador()** se encarga de guardar el nombre o nick del usuario y posteriormente mostrarlo al ingresar a la opción de puntajes.

```
InicioProceso registrarJugador()
    Llamar método jugador.cambiarNombre(ventana.mostrarNombre())
FinProceso
```

Clase IniciarPartida

En esta clase se ejecuta la función principal del juego, es decir, donde se desarrollará el juego de batalla, donde el usuario podrá elegir un nivel, elegir un tablero y comenzar a jugar.

```
Inicio Proceso IniciarPartida

    Var usuario

    Var tablero
```

```
    /*Métodos se explican a continuación*/

FinProceso
```

El método **elegirNivel()** se encarga de mostrarle al usuario la dificultad en la que desea jugar, siendo estos: principiante, intermedio y titán, al elegir un nivel se le muestra al usuario únicamente los mapas que corresponden a cada tipo de dificultad, luego de seleccionar el mapa se le asignará un inventario correspondiente.

```
Inicio Proceso elegirNivel()
    Var opcion

    Escribir “Elija uno el nivel de dificultad de la partida.”
    Escribir “0. Regresar”
    Escribir “1. Principiante”
    Escribir “2. Intermedio”
    Escribir “3. Titán”
    Leer opcion

    Si (opcion == 0) entonces
        Llamar método mostrarMenu()
    Sino si (opcion ==1 ) entonces
        Llamar método asignarInventario(nivel)
    Sino si (opción == 2) entonces
        Llamar método asignarInventario(nivel)
    Sino si (opción == 3) entonces
        Llamar método asignarInventario(nivel)
    Finsi
FinProceso
```

El método **asignarInventario()** se encarga de asignar un tablero, bombas especiales y bombas normales, el tamaño del tablero y cantidad de bombas especiales y normales se asignarán dependiendo de la dificultad de juego que ha elegido el usuario.

```
Inicio Proceso asiganrInventario(Var nivel)
    Si (nivel == “principiante”) entonces
        Llamar método agregarTablero()
        Llamar método agregarBombasEspeciales()
        Llamar método agregarBombasNormales()
```

```
    Sino si (nivel == “intermedio”) entonces
        Llamar método agregarTablero()
        Llamar método agregarBombasEspeciales()
        Llamar método agregarBombasNormales()
    Sino si (nivel == “titan”) entonces
        Llamar método agregarTablero()
        Llamar método agregarBombasEspeciales()
        Llamar método agregarBombasNormales()
    Finsi
FinProceso
```

El método **mostrarTableroSeleccionado()** se encarga de mostrar en pantalla el mapa seleccionado por el usuario y poder así iniciar partida.

```
Inicio Proceso mostrarTableroSeleccionado()
    Desde i = 0; hasta i = mapa.tamaño(); i++
        Desde j = 0; hasta j= mapa.tamaño(); j++
            Escribir “mapaSeleccionado[i][j]”
        FinDesde
    FinDesde
FinProceso
```

El método **comenzarPartida()** se encarga de mostrar en pantalla las bombas que se encuentran en el mapa, barcos destruidos, mapa con posiciones de barcos y bombas escondidos. El usuario da click sobre una casilla y si no hay nada en esa casilla entonces solo se mostrará agua, si hay una bomba, dependiendo del tipo esta hará una acción de estallar lo que se encuentre a su alrededor dependiendo de sus límites, si es una parte de un barco entonces se mostrará una imagen y se sumará puntos al usuario para luego crear el archivo de registro.

```
Inicio Proceso comenzarPartida()
    Llamar accionBoton()

    Si (numeroBarcos > 0 || numeroBombas > 0) entonces
        Llamar método mostrarCasilla()
        Llamar método cambiarConteoBombas()
    Finsi
FinProceso
```

Clase NuevaPartida

La clase NuevaPartida se encarga de cargar un mapa desde la computadora del usuario a través de un archivo con extensión .th, de usar otra extensión se le mostrará un mensaje que le indicará que no puede cargar ese archivo.

```
Inicio Proceso NuevaPartida
    Var Usuario
    Var vacio = true

    /*Métodos se explican a continuación*/
FinProceso
```

El método **validarArregloVacio()** se encarga de validar si hay mapas existentes, si no hay mapas existentes entonces se el mostrará al usuario un texto de información sobre los datos y formato que debe llevar el archivo con extensión .th para la creación del mapa.

```
Inicio Proceso validarArregloVacio()
    Desde i = 0; hasta i = mapa.tamaño(); i++
        Desde j = 0; hasta j = mapa.tamaño(); j++
            Desde k = 0; hasta k = mapa.tamaño(); k++
                Si (mapa[i][j][k] == null) entonces
                    Llamar método mostrarMensajeDeCargaDeArchivo()
                sino
                    Llamar método mostrarMensjaeDeCargaDeArchivo()
                    vacio = false
                Finsi
            FinDesde
        FinDesde
    FinDesde
FinProceso
```

El método **mostrarPresivualizador()** se encarga de mostrar los mapas cargados, al momento de dar click en cargar mapa se le mostrará una ventana en la que pueda buscar el archivo y si la lectura de archivo es correcta entonces se mostrara un pequeño visualizador del mapa, seguidamente debe seleccionar el mapa actual y presionará el botón siguiente, luego se dirigirá a la clase IniciarPartida para la ejecución del juego de la partida.

```
Inicio Proceso mostrarPrevisualizador()
    Llamar método mostrarJPanel(Componente mapa)
FinProceso
```

El método **crearNuevaPartida()** se encarga de crear nueva partida, al momento de dar click en el botón siguiente se dirige al método elegirNivel() de la clase IniciarPartida()

Inicio Proceso crearNuevaPartida()

Llamar método elegirNivel()

FinDesde

Clase Punteo

Esta clase se encarga de mostrar los punteos y registro de actividades de una partida, además de visualizar en un Jpanel ya sea el registro de puntos o registro de actividades.

InicioProceso Punteo

/*Métodos se explican a continuación*/

FinProceso

El método **mostrarPuntajes()** se encarga de crear un registro de puntos del jugador cada vez que termine una partida, los datos se guardarán en un archivo llamado puntos con extensión .war, cada vez que se termine una partida este archivo se sobre escribirá u se acumularán en el mismo archivo.

Inicio Proceso mostrarPuntajes()

FinProceso

El método **mostrarAcciones()** se encarga de registrar las acciones durante una partida, es decir, cada vez que el usuario de click en una casilla, si en esta hay una bomba y estalla, se escribirá en un archivo llamado acciones.avn, la extensión .avn permite que este arhcivo sea únicamente de lectura, en este archivo se colocarán datos como nombre del jugador, acción realizada, si estallo una bomba o le hizo daño a un barco o falló el tiro, luego se mostrará la posición de la casilla seleccionada y como último dato se mostrará la hora. Este método recibe parámetros como nombre de usuario, acción realizada, posición de casilla en x, posición de casilla en y, hora.

Inicio Proceso mostrarPuntajes(Var nombreUsuario, Var accion, Var posicionX, Var posicionY, Var hora)

FinProceso

Clase ColecciónTablero

Esta clase se encarga de gestionar los mapas guardados después de que el usuario haya cargado un mapa, se guarda en un arreglo y seguidamente se mostrarán en pantalla.

```
Inicio Proceso ColeccionTablero
    /*Métodos se explican a continuación*/
FinProceso
```

El método **mostrarColeccionTableros()** se encarga de verificar si el arreglo está vacio, si esta vacio entonces se mostrará un mensaje al usuario que debe de cargar un archivo para un nuevo mapa.

```
Inicio Proceso mostrarColeccionTableros()
    Desde i = 0; hasta i = mapas.tamaño(); i++
        Desde j = 0; hasta j = mapas.tamaño(); i++
            Desde k = 0, hasta k = mapas.tamaño(); k++
                Escribir mapas[i][j][k].mostrarCasilla()
            FinDesde
        FinDesde
    FinDesde
FinProceso
```

El método **crearTablero(Var cadena)** se encarga de crear un tablero en base a una cadena recibida, para identificar el contenido que tendrá cada casilla se verifica que carácter poseen la cadena recibida como parámetro.

```
Inicio Proceso crearTablero(Var[] cadena)
    Desde i = 0; hasta i = cadena.tamaño;, i++
        Var caracteres[] = cadena.split(“,”)
        Desde j = 0; hasta  j = verificarContenido(); j++
            casillas[i][j].cambiarPosicionX()
            casillas[i][j].cambiarPosicionY()

            Si (caracteres[j] ==  “T”) entonces
                casillas[i][j] == new Torpedo()
            Sino si (caracteres[j] == “I”) entonces
                casillas[i][j] = new Misil()
            Sino si (caracteres[j] == “O”) entonces
                casillas[i][j] == new Hecatombe()
            Sino si (casillas[i][j] == “B1”) entonces
                casillas[i][j] = new Pailebot()
            Sino si (caracteres[j] == “B2”) entonces
```

```
                    casillas[i][j] == new Bergantin()
                Sino si (caracteres[j] == “B3”) entonces
                    casillas[i][j] = new Navio()
                Finsi
            FinDesde
        FinDesde
Fin Proceso
```

El método **crearTableroVacio()** se encarga de crear un tablero que solo mostrará casillas con agua, este mapa será utilizado para mostrar en partida para que cuando el usuario de click en una casilla este ya muestre el contenido de esa casilla.

```
Inicio Proceso crearTableroVacio()
    Desde i = 0: hasta j = mapas.tamaño(); i++
        Desde j = 0; hasta j = mapas[i].tamaño; j++
            Desde k = 0; hasta k = mapas[i][j].tamaño(); k++
                mapas[i][j][k] = new Agua()
            FinDesde
        FinDesde
    FinDesde
FinProceso
```