Universidad de San Carlos de Guatemala
Centro Universitario de Occidente
División de Ciencias de la Ingeniería
Introducción a la programación y computación 1 Sección “A”
Segundo Semestre 2024

ENRIQUE ALEXANDER TEBALAN HERNANDEZ
Carné: 202230026

MANUAL TECNICO PRACTICA #1 2024

{ BIENVENIDO }

Manual Técnico

CREADO EN: NETBEANS IDE 22
LENGUAJE: JAVA
Utilizando Programación Estructural y Orientada a Objetos (POO)

INTRODUCCION:
Este manual técnico describe la implementación de un sistema de juegos en consola que incluye los juegos "Carrera de Caballos", "Anagramas en Consola", y "Battleship en Consola". Se detalla la estructura del código, las clases principales, y se proporcionan algoritmos en pseudocódigo para los subprogramas más importantes del sistema.

Estructura del Proyecto

El proyecto está estructurado en las siguientes clases principales:

- **PRACTICA1_ENRIQUE_TEBALAN_SS2024:** Clase principal que contiene el punto de entrada (main) del programa.
- CarreraCaballos: Implementa el juego de Carrera de Caballos.
- Anagramas: Implementa el juego de Anagramas.
- Battleship: Implementa el juego de Battleship.

- Reportes: Clase que maneja la recopilación y presentación de estadísticas de los juegos.

Cada una de estas clases tiene métodos que gestionan la lógica del juego y la interacción con el usuario.

Pseudocódigo de Subprogramas Importantes

1. Método iniciarJuego de la clase CarreraCaballos

Pseudocódigo:

```
PROCEDIMIENTO iniciarJuego
  reportes.incrementarVecesIniciadoCarreraCaballos()
  Mostrar mensaje "Elige tu caballo:"
  Leer elección del usuario
  Mientras no haya un ganador
    Para cada caballo en la carrera
      Generar número aleatorio para avanzar el caballo
      Si caballo actual ha ganado
        reportes.incrementarVecesGanadoCaballo(color del caballo)
        Si caballo es el del jugador
          reportes.incrementarVecesGanadasHipodromo()
        Sino
          reportes.incrementarVecesPerdidasHipodromo()
        Terminar la carrera
    Fin Para
  Fin Mientras
FIN PROCEDIMIENTO
```

Método iniciarJuego de la clase Battleship

Pseudocódigo:

```
PROCEDIMIENTO iniciarJuego
  reportes.incrementarVecesIniciadoBattleship()
  Si modo de juego es contra IA
    Colocar barcos para IA automáticamente
  Fin Si
  Mientras no haya ganador
    Leer ataque del jugador
    Si el ataque es exitoso
```

```
        reportes.incrementarBarcosDestruidosPorJugador()
      Fin Si
      Si IA está jugando
        Generar ataque de IA
        Si el ataque es exitoso
          reportes.incrementarBarcosDestruidosPorIA()
        Fin Si
      Fin Si
      Verificar si hay un ganador
    Fin Mientras
    Si el jugador ha ganado
      reportes.incrementarVecesGanadasIABattleship()
    Fin Si
FIN PROCEDIMIENTO
```

Método mostrar Reportes de la clase Reportes

```
PROCEDIMIENTO mostrarReportes
    Mostrar "Reporte de Juegos:"
    Mostrar "Veces iniciado Carrera de Caballos: " +
vecesIniciadoCarreraCaballos
    Mostrar "Veces iniciado Anagramas: " + vecesIniciadoAnagramas
    Mostrar "Veces iniciado Battleship: " + vecesIniciadoBattleship
    Mostrar "Veces ganadas en Hipódromo: " + vecesGanadasHipodromo
    Mostrar "Veces perdidas en Hipódromo: " + vecesPerdidasHipodromo
    Mostrar "Veces ganadas contra IA en Battleship: " +
vecesGanadasIABattleship
    Mostrar "Barcos destruidos por el jugador: " + barcosDestruidosPorJugador
    Mostrar "Barcos destruidos por la IA: " + barcosDestruidosPorIA
    Mostrar "Color de caballo más ganador: " + colorCaballoMasGanador + " con
" + vecesGanadasCaballo + " veces ganadas"
FIN PROCEDIMIENTO
```

Método iniciarJuego de la clase Anagramas

Pseudocódigo:

```
PROCEDIMIENTO iniciarJuego
    reportes.incrementarVecesIniciadoAnagramas()
    palabraMezclada = mezclarPalabra(palabraOriginal)
```

```
  Mostrar palabraMezclada
  Mientras intentosRestantes > 0 y palabrasEncontradas < totalPosibles
    Leer intento del jugador
    Si intento es válido
      Añadir intento a palabrasEncontradas
      Mostrar palabrasEncontradas
    Sino
      Restar 1 a intentosRestantes
    Fin Si
  Fin Mientras
  Si palabrasEncontradas = totalPosibles
    Mostrar "¡Ganaste!"
  Sino
    Mostrar "Perdiste. La palabra era: " + palabraOriginal
  Fin Si
FIN PROCEDIMIENTO
```

Procesos en Juego Battleship

```
ALGORITMO colocarBarco(barco, filaInicio, columnaInicio, filaFin,
columnaFin)
   // Verificar que la colocación sea horizontal o vertical
   SI (filaInicio ≠ filaFin Y columnaInicio ≠ columnaFin) ENTONCES
      RETORNAR FALSO // No se permite colocación diagonal
   FIN SI

   // Verificar que las coordenadas estén dentro del tablero
   SI (NO coordenadasValidas(filaInicio, columnaInicio) O NO
coordenadasValidas(filaFin, columnaFin)) ENTONCES
      RETORNAR FALSO
   FIN SI

   largo ← barco.getTamanio()

   // Verificar que el barco encaje en las coordenadas especificadas
   SI (ABS(filaFin - filaInicio + columnaFin - columnaInicio) + 1 ≠ largo)
ENTONCES
      RETORNAR FALSO
   FIN SI
```

```
   // Verificar que no haya solapamiento
   SI (filaInicio = filaFin) ENTONCES // Colocación horizontal
      PARA j DESDE MIN(columnaInicio, columnaFin) HASTA
MAX(columnaInicio, columnaFin) HACER
         SI (tablero[filaInicio][j] ≠ AGUA) ENTONCES
            RETORNAR FALSO // Hay un solapamiento
         FIN SI
      FIN PARA
   SINO // Colocación vertical
      PARA i DESDE MIN(filaInicio, filaFin) HASTA MAX(filaInicio, filaFin)
HACER
         SI (tablero[i][columnaInicio] ≠ AGUA) ENTONCES
            RETORNAR FALSO // Hay un solapamiento
         FIN SI
      FIN PARA
   FIN SI
   // Colocar el barco en el tablero
   SI (filaInicio = filaFin) ENTONCES // Colocación horizontal
      PARA j DESDE MIN(columnaInicio, columnaFin) HASTA
MAX(columnaInicio, columnaFin) HACER
         tablero[filaInicio][j] ← barco.getSimbolo()
         color[filaInicio][j] ← barco.getColor() // Almacenar el color del barco
      FIN PARA
   SINO // Colocación vertical
      PARA i DESDE MIN(filaInicio, filaFin) HASTA MAX(filaInicio, filaFin)
HACER
         tablero[i][columnaInicio] ← barco.getSimbolo()
         color[i][columnaInicio] ← barco.getColor() // Almacenar el color del
barco
      FIN PARA
   FIN SI
   RETORNAR VERDADERO // Barco colocado con éxito
FIN ALGORITMO
```

Este manual técnico proporciona una visión general del sistema, enfocándose en la implementación de la lógica principal de los juegos en consola. Los algoritmos en pseudocódigo cubren las partes más críticas del código, permitiendo a los desarrolladores comprender y modificar el sistema según sea necesario.