Rullo

Asignatura: **Introducción a la programación**

Curso: **2018/2019**

Autor: **Pedro Miguel Carmona Broncano**

DNI: **54 332 868 Z**

Autor: **Ruben Marín Lucas**

DNI: **07 272 889 J**

Grupo de prácticas: **A3a / B1a**

Profesor de prácticas: **Alberto Gómez / Cristina Vicente**

Convocatoria: **Enero 2019**

Índice de contenido

1. Introducción

Este proyecto pretende desarrollar el videojuego Rullo, el cual consiste en un tablero de tamaño ***n x n***, (n varía entre el intervalo [4,8]) donde las filas y las columnas deben sumar lo indicado.

El jugador irá moviéndose con las teclas de dirección en un tablero n x n, irá desactivando casillas con la tecla *Espacio* para conseguir que las casillas de las filas y las columnas sumen el resultado mostrado.

También existe la posibilidad de que el usuario bloquee una serie de casillas para no estropear una fila y columna, si el usuario piensa que esa fila o columna esta resuelta como es debido.

2. Análisis y diseño

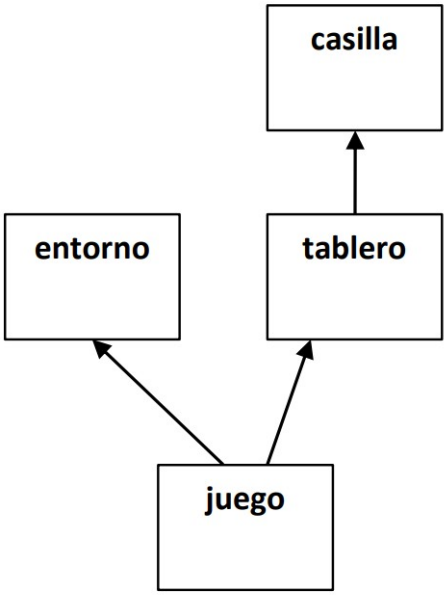
2.1. Análisis

Este software se ha dividido en 4 tipos abstractos de datos distintos (entorno, juego, tablero y casilla).

El TAD Entorno es el que se encarga de toda la implementación gráfica y toda la información se almacena en memoria en el TAD Tablero y el TAD Casilla.

Por último, el TAD Juego es el encargado de la interacción entre los gráficos y la memoria.

2.2. Diagrama modular



El **entorno** gráfico es el encargado de relacionar la interfaz gráfica con la memoria.

Una ***casilla*** estará formada por un registro, donde almacenará el estado de activación y el estado de bloqueo de la casilla, además del valor de esta.

El ***tablero*** estará formado por un registro, donde almacenará una matriz de tipo "Casilla", además de dos vectores que serán la suma de las filas y las columnas.

El ***juego*** estará formado también por un registro, que almacenará el "tablero" y por una variable de tipo entero que se utilizará para acumular los puntos.

La librería **entorno** es la encargada de mostrar en pantalla los cambios.

El ***TAD Casilla*** es el encargado de guardar la información de una casilla, es decir si está activa o no, bloqueada o no, y el valor de la casilla.

El ***TAD Tablero***, es el encargado de gestionar todo lo que ocurre en el tablero, cómo se inicia, cuando se modifica…

El ***TAD Juego*** es el encargado de toda la interacción entre el entorno grafico y la memoria: la entrada de datos por teclado, la salida da datos por pantalla, cuando se acaba el juego...

2.3. TAD Casilla

TAD

Casilla es **iniciarCasilla, ponerValor, cambiarEstadoActivacion, cambiarEstadoBloqueo, obtenerValor, casillaActiva, casillaBloqueada.**

OPERACIONES

iniciarCasilla(c: Casilla)

modifica Casilla.

efecto Inicia la casilla.

ponerValor(c: Casilla, v: entero)

modifica Casilla.

efecto Modifica el valor de la casilla c, cambiándolo por el valor v.

cambiarEstadoActivacion(c: Casilla)

modifica Casilla.

efecto Modifica el estado de activación de la casilla.

cambiarEstadoBloqueo(c: Casilla)

modifica Casilla.

efecto Cambia el estado de bloqueo de la casilla.

obtenerValor(c: Casilla)

retorna entero.

efecto Devuelve el valor de la casilla.

casillaActiva(c: Casilla)

retorna booleano.

efecto Devuelve el estado de activación de la casilla.

casillaBloqueda(c: Casilla)

retorna booleano.

efecto Devuelve el estado de bloqueo de la casilla.

2.4. TAD Tablero

TAD

Tablero es **iniciarAleatorio, iniciarArchivo, copiarDimensionReal, cambiarValorCelda, cambiarEstadoActivacionCelda, cambiarEstadoBloqueoCelda,obtenerValorCelda, obtenerDimensionReal, obtenervSumaFilas, obtenervSumaCol, obtenerEstadoActivacionCelda, obtenerEstadoBloqueoCelda, sumaCasillasFilas, sumaCasillasColumnas, tableroResueltoFilas, tableroResueltoColumnas.**

OPERACIONES

iniciarAleatorio (t:Tablero, MAX_Desactivar: entero)

modifica Tablero.

efecto Inicia el tablero con valores aleatorios y posteriormente suma las filas y las columnas.

iniciarArchivo (t: Tablero, m: TipoMatrizInt, vSumaFilas: TipoVector, vSumaCol: TipoVector)

modifica Tablero.

efecto Inicia el tablero según el archivo de configuración “rullo.cnf”.

copiarDimensionReal(t: Tablero, tamano: entero)

modifica Tablero.

efecto Modifica el valor del campo dimReal del tablero t por la variable tamano.

cambiarValorCelda(t: Tablero, i: entero, j: entero, v: entero)

modifica Tablero.

efecto Modifica el campo “valor” de la casilla en la posición (i,j) por la información de la variable v.

cambiarEstadoActivacionCelda(t: Tablero, i: entero, j: entero)

modifica Tablero.

efecto Modifica el estado de activación de la casilla (i,j).

cambiarEstadoBloqueoCelda(t: Tablero, i: entero, j: entero)

modifica Tablero.

efecto Modifica el estado de bloqueo de la casilla (i,j).

obtenerValorCelda(t: Tablero, i:entero, j:entero)

retorna entero.

efecto Devuelve el valor de la celda en la posición (i,j).

obtenerDimensionReal(t: Tablero)

retorna entero.

efecto Devuelve el tamaño real de t.

obtenervSumaFilas(t: Tablero, i:entero)

retorna entero.

efecto Devuelve el valor de la posición i del vector "t.vSumaFilas".

obtenervSumaCol(t: Tablero, i:entero)

retorna entero.

efecto Devuelve el valor de la posición j del vector "t.vSumaCol".

obtenerEstadoActivacionCelda(t: Tablero, i:entero, j:entero)

retorna booleano.

efecto Devuelve el estado de activación de la celda en la posición (i,j).

obtenerEstadoBloqueoCelda(t: Tablero, i:entero, j:entero)

retorna booleano.

efecto Devuelve el estado de bloqueo de la celda en la posición (i,j).

sumaCasillasFilas(t: Tablero, k:entero)

retorna entero.

efecto Suma las casillas activas de la fila k.

sumaCasillasColumnas(t: Tablero, k:entero)

retorna entero.

efecto Suma las casillas activas de la columna k.

tableroResueltoFilas(t: Tablero)

retorna entero.

efecto Comprueba si la suma de las casillas activas en cada fila coincide con su valor resultado en "t.vSumaFilas".

tableroResueltoColumnas(t: Tablero)

retorna entero.

efecto Comprueba si la suma de las casillas activas en cada columna coincide con su resultado en "t.vSumaCol".

2.5. Programa principal

El programa principal estará formado por la declaración de un Juego y por tres módulos **iniciarJuego, jugar** *y* ***juegoDePruebas.*** Este último llamará a todos los módulos de pruebas.

2.6. Ampliaciones

Para este proyecto se ha elegido las ampliaciones de **Ayuda** *y* **Puntos.**

Para la ampliación **Ayuda** hemos añadido un nuevo caso de comprobación a la estructura *switch ,* en ella se evalúa si se a pulsado la tecla *TY.* Esta ampliación muestra por un breve periodo de tiempo la suma actual de las casillas activas de la fila y la columna donde se encuentre el cursor.

Para la ampliación **Puntos** hemos añadido el campo puntos (variable de tipo entero) en la estructura del TAD_Juego. Estos puntos se modificarán al pulsar la tecla *Tspace (se restarán 3 puntos si se activa una casilla y 2 en caso contrario)*y la tecla *TY (se restarán la mitad de los puntos)* y no podrá ser negativos.

3. Planificación y tareas

Día Reunión	Horas empleadas:	Objetivo:
12/05/18	Desde 15:00 a las 17:30	- Planificación y reparto de tareas. -Implementación a papel y en eclipse del TAD Casilla - Implementación a papel del TAD Tablero
12/08/18	Desde 18:30 a las 19:30	- Cambio en la planificación inicial.
12/10/18	Desde las 20:00 a las 21:30	- Implementación conjunta del TAD Tablero en eclipse.
12/11/18	Desde las 16:30 a las 19:30	Implementación del modulo de implementación aleatoria, y pruebas realizadas con éxito de este modulo.
12/12/18	Desde las 18:00 a las 20:30	- Implementación del modulo de iniciación leyendo desde el archivo de configuración,pruebas realizadas de ese modulo.

12/04/18	Desde las 12:30 a las 14:00y de las 16:30 a las 19:30	- Implementación de la estructura del TAD Juego, su operación inicial y algunas de las funciones de las teclas.
12/15/18	Desde las 18:00 a las 20:00	- Implementación de las funciones de las teclas restantes.
12/18/18	Desde las 17:00 a las 21:00	- Implementación de el modulo para iniciar el juego aleatoriamente, y implementación de la primera ampliación (Tecla Y)
01/14/19	Desde las 12:00 a las 14:00 y de las 16:00 a las 21:00	Mejora de sintaxis en todo el proyecto.
01/15/19	Desde las 11:00 a las 14:00 y de las 16:00 a las 00:00	Corrección de errores y documentación externa.

4. Conclusiones y principales problemas

Desde el día en el que nos dijeron que el proyecto final de la asignatura era realizar un videojuego, estábamos entusiasmados a ponernos a programarlo. Pero en el momento que empezamos, nos dimos cuenta que no era tan sencillo como parecía, desde un primer momento tuvimos problemas para diferenciar entre memoria y entorno gráfico.

Otro de los problemas que hemos tenido ha sido a la de implementar los campos de un TAD en otro diferente al campo que queríamos implementar.

El primer reto de la realización de este trabajo en parejas, fue la organización, fue lo más costoso al principio, pero una vez organizado el trabajo fue sobre ruedas.

Por último, aunque hemos sufrido lo nuestro haciendo este proyecto, no dudaríamos en volver a juntarnos para desarrollar otro proyecto similar o incluso más complicado.