**“UNIVERSIDAD NACIONAL JOSÉ MARÍA ARGUEDAS”**

**FACULTAD DE INGENIERÍA**

**Escuela Profesional de Ingeniería de Sistemas**



**TEMA:** Juego de Tetris

**ASIGNATURA:** INGENIERÍA DE SOFTWARE III

**DOCENTE:** Ing. Iván SORIA SOLÍS.

**ALUMNO:** GERMIOS LEGUÍA ARONI.

**ANDAHUAYLAS-PERÚ**

**2014**

**IMPLEMENTACIÓN DE JUEGO DE TETRIS:**

**INTRODUCCIÓN:**

La aplicación consiste en desarrollar en java el juego del tetris: 4 figuras geométricas, formadas por cuatro bloques cuadrados unidos de forma ortogonal, que caen de la parte superior de la pantalla. El jugador puede hacer girar estas piezas. Cuando una línea horizontal se completa, la línea desaparece y las piezas que están por encima descienden una posición. Las piezas se generan de manera aleatoria. No se crea una nueva mientras haya otra en movimiento. El jugador controla el movimiento de las piezas mediante pulsaciones del teclado (teclas de cursor): puede hacerlas desplazarse a la izquierda o a la derecha (dentro de los límites del tablero), o hacerlas rotar. Las piezas caen con movimiento uniforme, a razón de una línea por segundo, y pueden efectuar, como máximo, cuatro giros por segundo. El juego termina cuando no pueden crearse piezas nuevas porque están ocupadas todas las casillas. Dejan de caer fichas y aparece la frase “FIN DEL JUEGO” en el centro del tablero.

**OBJETIVO:**

El objetivo del proyecto es implementar el juego de Tetris, valiéndose o utilizando lenguajes de programación como java.

## REQUISITOS FUNCIONALES Y NO FUNCIONALES:

### REQUISITOS FUNCIONALES

Describen el comportamiento que debe adoptar el sistema. Qué acciones ha de desarrollar, cómo las debe de llevar a cabo y qué resultado se debe dar para cada una de ellas.

* Interfaz amigable
* Iniciar juego
* LimpiarTablero
* Controles
* Dirección Derecha
* Dirección Izquierda
* Salir del juego

### LISTA DE REQUISITOS NO FUNCIONALES

Estos requisitos son implícitos al sistema pero no son funcionalidades concretas a este. Se descomponen en distintos apartados:

* **Sistema operativo.**

El sistema operativo será win8 para desarrollo e implementación de la aplicación de juego de Tetris.

* **Requerimientos de producto**

Se desarrollara en el lenguaje de programación JAVA, en plataforma de NETBEANS Y JDK DE 64 BITS, por tener un entorno amigable al usuario.

* **Requerimientos organizacionales**
* Implementación de software tendrá una duración de dos meses.
* **Requerimiento externo.**
* El cliente no tendrá derechos a modificar, eliminar, etc.
* **Requerimiento calidad:**
* Para el correcto funcionamiento del juego de Tetris, es necesario que esta se mueva en unos márgenes de calidad adecuados.

**Otros requisitos.**

* **Requisitos que deben cumplir los usuarios**

Los jugadores deberán de manejar adecuadamente el sistema o pc y tener conocimiento del juego para poder jugar fácilmente y ganar.

**Requisitos funcionales y no funcionales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **RU01** | **Necesidad** | **ALTA** | |
| **Titulo** | **Icono ejecutable** | | | |
| **Descripción** | La aplicación debería tener un fichero que permita la ejecución directa de esta, para facilidad del usuario final. | | | |
| **Identificador** | **RU02** | **Necesidad** | | **ALTA** |
| **Titulo** | **Controles** | | | |
| **Descripción** | La aplicación deberá permitir que el usuario pueda utilizar el teclado y el ratón para interactuar con esta. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU03** | **Necesidad** | **Esencial** |
| **Titulo** | **Menús** | | |
| **Descripción** | La aplicación deberá tener un sistema de navegación de pantallas o menús que permita al usuario ir configurando las diferentes opciones que le permita dicha aplicación. Además de iniciar y finalizar el juego. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU3** | **Necesidad** | **Esencial** |
| **Titulo** | **Música de fondo** | | |
| **Descripción** | La aplicación deberá permitir al usuario habilitarla y deshabilitarla. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU5** | **Necesidad** | **Esencial** |
| **Titulo** | **Sistema de Puntuación** | | |
| **Descripción** | La aplicación debe tener un sistema de puntuación basado en el tiempo que le lleva al jugador superar los diferentes niveles. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU6** | **Necesidad** | **Esencial** |
| **Titulo** | **Sistema Operativo** | | |
| **Descripción** | La aplicación deberá funcionar en cualquier versión de Windows xp o superior.  . | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU14** | **Necesidad** | **Esencial** |
| **Titulo** | **Música de fondo** | | |
| **Descripción** | La aplicación deberá permitir al usuario habilitarla y deshabilitarla. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Identificador** | **RU16** | **Necesidad** | **Esencial** |
| **Titulo** | **Música de fondo 3** | | |
| **Descripción** | La aplicación debería tener músicas diferentes para cuando el usuario este jugando en el tablero. | | |

**Diagrama de casos de uso. **

**Casos de uso detallado**

|  |  |
| --- | --- |
| **Identificador** | **CU-01** |
| **Titulo** | Iniciar Juego |
| **Actores** | Jugador |
| **Objetivo** | Iniciar Aplicación |
| **Precondiciones** | La aplicación debe estar instalada en el ordenador en el que se vaya a usar y tenga un sistema operativo compatible con esta. |
| **Post-condiciones** | Aparecerá en pantalla el menú principal del juego. |

|  |  |
| --- | --- |
| **Identificador** | **CU-02** |
| **Titulo** | Configurar Opciones |
| **Actores** | Jugador |
| **Objetivo** | Permitir al usuario acceder a la configuración de opciones como nivel básico, avanzado. |
| **Precondiciones** | Tener en pantalla el menú principal |
| **Post-condiciones** | Se cargará el menú de opciones. |

|  |  |
| --- | --- |
| **Identificador** | **CU-03** |
| **Titulo** | Música |
| **Actores** | Jugador |
| **Objetivo** | Permitir al jugador habilitar o deshabilitar el sonido en el juego |
| **Precondiciones** | Tener en pantalla el menú de opciones. |
| **Post-condiciones** | En el menú de opciones aparecerá el estado de la música seleccionada, On / Off. |

|  |  |
| --- | --- |
| **Identificador** | **CU-04** |
| **Titulo** | Nivel Básico |
| **Actores** | Jugador |
| **Objetivo** | Permitir al jugador elegir entre los diferentes niveles para jugar. |
| **Precondiciones** | Tener en pantalla el menú de opciones. |
| **Post-condiciones** | En el menú de opciones aparecerá la dificultad seleccionada, fácil, normal o difícil. |

|  |  |
| --- | --- |
| **Identificador** | **CU-05** |
| **Titulo** | Nivel Avanzado |
| **Actores** | Jugador |
| **Objetivo** | Permitir al jugador elegir entre los diferentes niveles para jugar. |
| **Precondiciones** | Tener en pantalla el menú de opciones. |
| **Post-condiciones** | Aparecerá en pantalla un recuadro con los controles de manejo del personaje. |

|  |  |
| --- | --- |
| **Identificador** | **CU-06** |
| **Titulo** | Jugar |
| **Actores** | Jugador |
| **Objetivo** | El uso principal de la aplicación, donde el usuario maneja al personaje para capturar los ítems y colocar de manera ordenada. |
| **Precondiciones** | Haber seleccionado el menú principal la opción de “Nivel Básico” o “Nivel Avanzado”. |
| **Post-condiciones** | Aparecerá en la pantalla el nivel con el personaje seleccionado. |

|  |  |
| --- | --- |
| **Identificador** | **CU-07** |
| **Titulo** | Controlar personaje |
| **Actores** | Jugador |
| **Objetivo** | Durante el juego, el usuario controla al personaje con el teclado. |
| **Precondiciones** | Nivel y personajes cargados y visualizados en pantalla. |
| **Post-condiciones** | El usuario controlará al personaje. |

|  |  |
| --- | --- |
| **Identificador** | **CU-08** |
| **Titulo** | Establecer record |
| **Actores** | Jugador |
| **Objetivo** | Cuando el jugador supera un nivel, la aplicación le asigna una puntuación en función del tiempo que le ha llevado. Si es una puntuación de record podrá guardar dicha puntuación. |
| **Precondiciones** | Superar un nivel. |
| **Post-condiciones** | Una vez introducido el nombre continúa la partida. |

|  |  |
| --- | --- |
| **Identificador** | **CU-09** |
| **Titulo** | Salir de aplicación |
| **Actores** | Jugador |
| **Objetivo** | Cerrar la aplicación. |
| **Precondiciones** | En pantalla este el menú principal. |
| **Post-condiciones** | Se cierra totalmente la aplicación. |

**Diagramas de clase en UML.**

****

**ANÁLISIS Y DISEÑO:**

He basado el programa en los siguientes conceptos:

**UN MÓDULO PRINCIPAL (MAIN):** en el que se generan el tablero y las piezas, y se toman las decisiones del juego: respuesta a las órdenes del jugador, movimiento automático de las fichas (caída), tiempo de caída, etc.

**UN TABLERO DE JUEGO:** en el que se dibuja una cuadrícula, sobre la cual se irán 'pintando' las piezas. El tablero se encarga de limpiar las filas llenas, bajar las casillas ocupadas de las filas superiores, y de asignar color de fondo (pintar) a las casillas ocupadas por piezas

**PIEZAS DEL JUEGO:** generadas de forma aleatoria, cada una con un color distinto. Cuando se genera una pieza, lo que hace el programa es dibujar la pieza sobre la cuadrícula (tablero), cambiando el color de fondo de las casillas ocupadas por la pieza por el color de dicha pieza. Para controlar si una casilla está ocupada y poder cambiar la posición de una pieza (girándola o desplazándola a derecha, izquierda o abajo), generar una pieza nueva o controlar el final del juego, lo que hacemos es comparar el color de las casillas que debería ocupar la pieza en la nueva posición: si el color de fondo de la casilla es blanco, está libre. En caso contrario, está ocupada y la pieza no puede desplazarse o generarse. Esto se resuelve preguntando a tablero por el color de fondo de las casillas que deberían ocupar las piezas.

Como cada pieza tiene una forma y un comportamiento distintos, se ha implementado una clase pieza donde se define el comportamiento común a todas. El resto de los métodos que se utilizan serán sobre escritos en cada uno de los distintos tipos de piezas, que heredan de la superclase *pieza*, adaptándolos a su comportamiento particular.