Proyecto de desarrollo de software “TitanChess”

Jose Daniel Amador Salas

José Pablo Brenes Alfaro

Instituto Tecnológico de Costa Rica

Ingeniería en Computación

Santa Clara, 2017

**Contenido**

[Problema que resolver 3](#_Toc494670443)

[Análisis del problema 3](#_Toc494670444)

[Solución Planteada 3](#_Toc494670445)

[Análisis de Resultados 4](#_Toc494670446)

[Recomendaciones 6](#_Toc494670447)

[Conclusiones 6](#_Toc494670448)

[Bibliografía 7](#_Toc494670449)

# **Problema que resolver**

Los estudiantes de computación de nuevo ingreso carecen de conocimientos en los lenguajes de programación y en la implementación de su lógica en el desarrollo de proyectos.

# **Análisis del problema**

Las empresas que se especializan en el software y su mercado como tal, exigen que las personas que ingresan al mercado laboral tengan conocimientos en los diferentes paradigmas de programación actuales. Dentro de los paradigmas más utilizados se encuentra el que comúnmente se conoce como Programación Orientada a Objetos (POO). Entre lenguajes orientados a objetos más demandados actualmente por las empresas, se pueden encontrar C++, C# y Java (James, 2017). Con el fin de brindarle a los estudiantes una manera de desenvolverse en esta área las universidades optan por la asignación de diferentes proyectos en los cuales se deba utilizar dichas tecnologías. Estos proyectos en su mayoría siempre están enfocados a áreas donde se necesita el desarrollo de la lógica y el análisis para la solución de posibles problemas. Por esta razón es que se propone la creación de videojuegos los cuales tienen como requerimiento una gran cantidad de análisis por parte de sus desarrolladores. Algunos de los principales requerimientos que debe tener un desarrollador en cuanto a POO, es el manejo de las interfaces, clases abstracción, encapsulamiento de datos, instalación de clases, entre algunos otros.

# **Solución Planteada**

Se busco el desarrollo de un programa que cumpliera con los requisitos de implementación de todas las características necesarias para poder adentrarse en la POO, y así adquirir experiencia en esta área. Por esta razón se planteó el desarrollo de un video juego MBA, al mejor estilo de los juegos de tablero por turnos que son comunes actualmente. El juego consiste en un tablero dividido en dos sectores iniciales, los cuales representan los territorios de los dos jugadores. Antes de iniciar el juego el usuario que organiza la partida deberá seleccionar los personajes que el usará y los de su contrincante, así como el tamaño de la arena y su respectivo elemento. Cada jugador tendrá la posibilidad de elegir las posicione de iniciales de sus torres las cuales serán el objetivo principal del juego, la eliminación de las torres del enemigo. La lógica del juego consiste en la creación de partidas en las cuales el objetivo de los dos jugadores es la destrucción de las torres enemigas, para esto se podrá mover los personajes seleccionados dentro del tablero y para poder hacer un ataque a una torre enemiga o a un personaje enemigo, los movimientos y los valores de ataque están definidos en la tabla de valores de cada personaje, los cuales esta definidos por el juego pero también pueden ser editados por el administrador del juego, cada usuario se puede registrar y acceder a la creación de una partida. Dentro del juego hay algunas especificaciones que influirán sobre el desarrollo del juego, algunas de estas son: los ataque se calculan aleatoriamente en un rango entre 1 y el valor máximo de daño del ataque definido para ese personaje en específico, los personajes reviven una vez que pasa un turno después de la muerte de este, cada ataque tiene una cantidad de mana que es descontada una vez realizado el ataque. Cada partida también cuenta con algunos pluses para cada jugador, por ejemplo: después de cada turno el jugador aumenta sus habilidades un 25%, así mismo cuando una torre es destruida o un personaje es asesinado se aumentan las habilidades 25%, si el elemento de la arena (puede ser tierra, agua, aire, fuego) es igual al del personaje se obtiene un bonus de 10% más de daño.

Para desarrollar de manera eficiente el proyecto se trabajó con una matriz lógica la cual cumple la función de simular el tablero gráfico. Según (Corredor, 2017) “es una colección de datos del mismo tipo, que se almacenan en posiciones consecutivas de memoria y reciben un nombre común”. De esta manera se respaldaron las posiciones de cada personaje y torre, además de utilizar una única clase abstracta para definir a los personajes y las torres y así poder manipular estos de manera más efectiva. Se implemento además un sistema de paquetes basado en el modelo vista controlador para mantener contenidas todas las clases del programa. Además, se implementó el uso de archivos binarios para así darle persistencia de datos al juego. Se utilizo la creación de objetos para el manejo de datos, por ejemplo: objetos tipo Torre, Titan, Estadísticas, Usuarios, entre otras…

# **Análisis de Resultados**

|  |  |  |
| --- | --- | --- |
| Tarea | Estado (%) | Observaciones |
| Gestión de Usuarios | Completo (100%) | Persistencia de datos mediante archivos binarios |
| Creación de arenas | Completo (100%) | Se pueden crear arenas de tamaño pequeño, mediano, grande. |
| Selección de arenas | Completo (100%) | Se puede cambiar de elemento aire, tierra, fuego y agua |
| Crear, editar, eliminar personaje | Completo (100%) | Se puede eliminar, modificar los personajes existentes o crear uno nuevo. |
| Movimiento de personajes | Completo (100%) | Se pueden desplazar arriba, abajo, adelante. Solo se puede mover un personaje por turno |
| Ataques de personajes | Completo (100%) | Se pueden crear ataques nuevos para cada personaje. |
| Bonus de ataque | Completo (100%) | 10% extra cuando el elemento del personaje e igual al de la arena |
| Bonus del juego | Completo (100%) | Incremento de habilidades cuando muere un elemento y restauración del mana |
| Resurrección de un personaje muerto | Completo (100%) | Cuando un personaje muere revive después de un turno |
| Tabla de atributos del personaje | Completo (100%) | Cada personaje tiene una tabla de valores. |
| Reducción de mana al utilizar una habilidad | Completo (100%) | Se reduce el mana al personaje que ataca, de acuerdo con su valor de consumo establecido |
| Creación de torres | Completo (100%) | Se puede seleccionar la posición de las torres, estas inician con 10000pts de vida |
| Partida terminada | Completo (100%) | Cuando se destruyen todas las torres enemigas se gana el juego. |
| Turnos de jugadores | Completo (100%) | Se puede escoger atacar, o pasar. Además, cuando alguna de |
| Partida | Completo (100%) | Se pueden escoger tres diferentes personajes |
| Estadísticas | Completo (100%) | Se pueden ver las estadísticas del jugador, Muertes en partida, torres destruidas, partidas jugadas, partidas ganadas, partidas perdidas. |

# **Recomendaciones**

* Trabajar un modelo de organización de clases, métodos, funciones y todos los elementos que conforman la estructura del proyecto.
* Crear un mapa secuencial, diagramas de flujo o cualquier otro elemento que pueda ayudar al entendimiento del problema.
* Utilizar alguno de los estándares de declaración de variables y métodos que existen.
* Trabajar con una matriz lógica para respaldar la información del tablero.
* Investigar la utilidad de las diferentes características de los lenguajes orientados a objetos.

# **Conclusiones**

Se concluye que la implementación de proyectos de desarrollo es una de las maneras más efectivas para aportar conocimientos a los estudiantes. Además de ser una herramienta super eficiente en cuanto a desarrollo de la lógica se refiere. Por otro lado, la implementación de los diferentes características de los lenguajes de programación orientados a objetos facilitan el desarrollo de algoritmos eficientes. La programación orientada a objetos representa uno de los principales recursos que tiene el mercado de software en la actualidad, por esta razón se impulsa el desarrollo de la lógica en esta área. Para el desarrollo de la solución planteada, se obtuvo conocimientos en la persistencia de datos, serialización, herencia, clases abstractas, encapsulamientos de datos.

# **Bibliografía**

James, L. (2017). *Conoce los 9 lenguajes de programación más demandados en el mercado*. [online] Noticias Universia Colombia. Available at: http://noticias.universia.net.co/educacion/noticia/2016/02/23/1136601/conoce-9-lenguajes-programacion-demandados-mercado.html.

Corredor, C. (2017). *logica de programacion "vectores", matrices*. [online] Camcorredor.blogspot.com. Available at: http://camcorredor.blogspot.com/2011/08/logica-de-programacion-vectores.html.