

**Asignatura:**

**ISC-210-T-001 Prog. Aplicada**

**Trabajo de Curso:**

**Kunai Hit**

**Autores:**

**2017-0861 Joseph De Los Santos**

**2017-0502 Ruben O. Diaz**

[Matrícula y Nombre del Estudiante 3]

**Tutor:**

**Prof. Miguel T. Moronta**

**Entregado en Fecha:**

lunes, 01 de abril de 2019

**PONTIFICIA UNIVERSIDAD CATÓLICA MADRE Y MAESTRA – PUCMM  
FACULTAD DE CIENCIAS DE LAS INGENIERÍAS  
Departamento de Ingeniería en Sistemas y Computación – ISC**

# Resumen / Abstracto del Problema

Se pide hacer un juego que contenga las siguientes especificaciones:

1. Físicas hechas por el programador.
2. Almacenamiento persistente.
3. Animaciones.

El juego que se eligio es un juego llamado “Knife Hit” el juego consiste en lanzar una cierta cantidad de cuchillos a un tronco, si se lanza un cuchillo y este golpea a otra que ya haya sido lanzado el jugador pierde, cuando los cuchillos se acaban y no se ha golpeado ningún cuchillo el jugador pasara a otro nivel.

El juego que se diseñó tiene como nombre “Kunai Hits” la trama del juego es idéntica al de “Knife Hit” con ligeros cambios. En el juego “Kunai Hits” se tiene la posibilidad de elegir una dificultad (fácil, medio o difícil). Durante la partida aparecen “Power Ups” con forma de pájaros. En la dificultad fácil no se tiene power ups ni obstáculos. El power Up azul duplica la puntuación de los siguientes dos cuchillos y se pueden acumular. El anti Power Up de color rojo hace que el jugador automáticamente pierda la partida.

Tabla de contenido

[Resumen / Abstracto del Problema 1](#_Toc409432871)

[Análisis 2](#_Toc409432872)

[Cuerpo del Informe 2](#_Toc409432873)

[Código Fuente 2](#_Toc409432874)

[Condiciones de Error 2](#_Toc409432875)

[Notas al Profesor 2](#_Toc409432876)

[Conclusiones 2](#_Toc409432877)

[Recomendaciones 2](#_Toc409432878)

[Referencias Bibliográficas 2](#_Toc409432879)

# Análisis

Objetivos de la asignación:

1. Programar un videojuego sencillo.
2. Utilizar animaciones sencillas en el videojuego.
3. Programar las físicas del juego.
4. Programar detección de colisiones.
5. Almacenamiento persistente

El videojuego “Knife Hit” es un juego sencillo con una jugabilidad sencilla y un ambiente un tanto “arcade”. El juego consiste en lanzar cuchillos a un pedazo redondo de madera, la cual será destruida después de ser impactada por todos los cuchillos.

El blanco va a estar rotando y moviéndose con obstáculos dependiendo de la dificultad elegida. El obstáculo que utilizamos son aves que si son impactadas por el cuchillo se pierde este obstáculo está disponible desde la dificultad media, para la dificultad difícil el objetivo se estará moviendo de lado a lado y las aves se generaran más seguido.

El blanco retiene los cuchillos impactados y son destruidos con el mismo al impactar el ultimo cuchillo. Las fórmulas primordiales que se utilizan son las de movimiento rectilíneo uniforme, movimiento rectilíneo uniformemente variado, y rotación y traslación.

Para el almacenamiento persistente se guardaran en un archivo XML las mejores puntuaciones con el nombre de su respectivo jugador, estarán ordenadas de mejor a peor.

# Cuerpo del Informe

El cuerpo del informe se estructura en dos capítulos:

- Capítulo 1 – Descripción del Negocio

Aquí usted expondrá todo lo referente a la investigación realizada sobre el negocio. Deben incluirse las reglas de negocio que permitan validar la aplicación.

Ejemplos de algunas reglas de negocio:

* Una asignatura tiene uno o varios profesores.
* Todas las personas tienen un local asociado.

- Capítulo 2 – Solución Propuesta

Este capítulo tiene la finalidad de describir la solución propuesta ante la problemática planteada. Incluir:

* Diagramas de flujo, pseudo-códigos o diagramas de clases, casos de uso, actividades, de estado, de secuencia o cualquier otra herramienta que explique cuáles algoritmos, librerías, funciones o componentes se utilizaron para la resolución del problema.
* Organización del proyecto en capas lógicas, indicando los paquetes o librerías implementados en cada capa (nombre y descripción del contenido).
* Descripción de los patrones utilizados en la modelación. Si se han implementado algunos patrones de diseño en sus diagramas, utilice este espacio para citar acerca de los mismos sus nombres, descripción general y finalidad de su uso en esta solución.
* Capturas de pantalla o ejemplos de uso. Muestre el proceso a seguir para los principales lugares en su proyecto, así como datos de prueba y flujos de trabajo. Puede auxiliarse de capturas de pantalla, ejemplos de salida o vídeos que ilustren su objetivo.

Debe describir de una manera natural, concisa y clara por qué prefirió un algoritmo y no otro. No "corra" el programa. Es muy contraproducente describir instrucción por instrucción todo el código; se deben agrupar las sentencias que guarden relación y explicar el motivo de las mismas como un todo. Trate de no mencionar cosas sobreentendidas.

Es la sección más oportuna para hacer un comentario de por qué hizo algo de una manera específica. Si su código tiene muchas funciones, módulos o librerías, describa los más relevantes por separado..

# Código Fuente

Bajo este título se incluye el código fuente del programa a entregar. Deberá siempre anexar al documento el o los archivo(s) fuente para poderlos compilar de manera adecuada. Puede adjuntar en documento Zip toda la solución, en caso de que sea necesario. En caso de que su proyecto tenga múltiples archivos de código fuente, o resulte impráctico colocarlo dentro de esta sección, bastará con enviar el código junto con este reporte sin incluirlo en esta sección.

# Condiciones de Error

Por lo general, hay motivos que pueden causar que el usuario de un programa obtenga una anomalía o que el algoritmo usado tenga ciertas restricciones de uso. Ejemplo de esto es cuando a un programa que opera con números positivos se le digitan negativos. Ocurre también con funciones que no son capaces de operar con un número muy grande (por ejemplo, el factorial).

Estas anomalías deben enumerarse en esta sección del trabajo. Deberá incluir ejemplos de corrida con errores en esta sección donde se contemple el mismo claramente y se pueda visualizar cualquier mensaje de error desplegado. Tenga cuidado al decir que su programa no tiene condiciones de error. Quedará luego a consideración del profesor valorar las condiciones de error.

# Notas al Profesor

En caso de que desee hacer un señalamiento al profesor acerca de su trabajo, incluirlo bajo este título al final. Puede incluir cualquier otro aspecto que el equipo considere necesario exponer o explicar.

Por ejemplo: aspectos tenidos en cuenta para el diseño de la interfaz, filosofía de trabajo en la aplicación, tratamiento de errores, ayudas, aspectos novedosos tenidos en cuenta en la implementación, organización del trabajo en equipo, cronogramas de trabajo (plan y real), entrevistas (planificadas, reales, plantillas utilizadas, tipos de usuarios/clientes entrevistados) , estándar de código utilizado, ente otros..

# Conclusiones

Aquí se abordan las conclusiones del trabajo realizado, no de la asignatura, ni sobre los beneficios que le reporta a un estudiante la realización de este proyecto de curso. Deben redactarse teniendo en cuenta:

* Estado de cumplimiento de los objetivos planteados en la introducción.
* Observaciones, mediciones, estilos, etc., puestos en práctica durante la ejecución del proyecto. Por ejemplo: conclusiones, que consideren de interés a partir de sus propias experiencias, respecto a la organización del trabajo en equipo.

# Recomendaciones

Se redactan teniendo en cuenta aspectos que quedaron pendientes, mejoras que en el futuro se pueden realizar a la aplicación, posibilidades de implantación, entre otros.

# Referencias Bibliográficas

Especifique las referencias bibliográficas utilizadas durante la elaboración del proyecto.