COMPUTER ENGINEERING

(CFIA ACCREDITED SEAL)

Project II:

From Mars to Saturn

Professor:

Milton Villegas Lemus

The Earth is the cradle of Humanity.
But one doesn't always live in the cradle.

Konstantin Tsiolkovsky

The best computer is a man,
and it's the only one that can be mass-produced by unskilled labor.

Wernher von Braun

The future of humanity is going to bifurcate in two directions:
Either it's going to become multiplanetary,
or it's going to remain confined to one planet and
eventually there's going to be an extinction event.

Elon Musk

1. Objetivo General

- Diseñar una solución computacional relacionada con conceptos de recursividad, almacenamiento en disco, con el fin de aplicar la teoría en problemas específicos, considerando las buenas prácticas de programación.

Objetivos Específicos

- Desarrollar un proyecto software de pequeña escala, aplicando recursividad para repetición en Python.
- Aplicar las habilidades adquiridas en los talleres y experimentadas de forma individual.
- Realizar la documentación requerida para el proyecto.

2. Antecedentes

Como se revisó en la especificación del Proyecto I en la historia de los videojuegos un clásico es el juego Space Invaders que vino consolidar lo que se ha clasificado como genero Shoot 'em up.

Se debe recordar que el juego fue creado por el ingeniero japonés Tomohiro Nishikado quien diseñó y desarrolló el hardware necesario para producir Space Invaders. Quizás de los elementos más destacables es que incluye en el videojuego de disparos un tipo clase de avance o progreso en el tiempo de juego mediante la conclusión de niveles, proporcionando en sí el efecto adictivo. En este proyecto se va a conservar la misma aproximación en el diseño, es decir, el núcleo de la dinámica de juego será la misma, lo que cambia es la ambientación, temática y escenarios que dejan a la creatividad de los autores.

El Juego es para ser desarrollado en grupos de dos estudiantes, pero esto no es mandatorio.

Descripción:

Como parte del desarrollo de un nuevo producto en una organización para la cual usted trabaja, se le encomendó a usted la creación de una aplicación prueba de concepto que permita comprobar las capacidades del producto. Mientras usted desarrolla dicha aplicación, la organización decide que sería conveniente realizar otra aplicación similar. Las pruebas a realizar tienen el propósito de evaluar el desempeño

en escenarios en que la cantidad de elementos en pantalla es mayor, y no saben si esto puede tener un efecto significativo sobre el desempeño en comparación a otros computadores similares.

Para el momento en que los requisitos para esta nueva aplicación de prueba son definidos, usted ya ha terminado la primera aplicación. Se considera que su experiencia le permitiría completar la nueva aplicación más rápidamente que otro de los trabajadores disponibles, por lo cual se le asigna a usted esta tarea. Se le provee la siguiente especificación de requisitos a cumplir:

Especificaciones técnicas:

- Resolución de pantalla a escogencia del programador
- Programa desarrollado en python 3 o posterior
- Uso de Pygame o Tkinter como biblioteca de interfaz gráfica
- Controles con las cuatro flechas del teclado para movimiento
- Biblioteca de sonido vlc-python
- Temática a escogencia del programador. Debe ser coherente.

Mecánica de juego:

En forma general, el juego consiste en esquivar proyectiles u obstáculos. Si la temática fuera similar al juego anterior, el jugador podría ser representado por una nave, y los proyectiles a esquivar podrían ser meteoritos.

El jugador controla su avatar utilizando las flechas direccionales del teclado. El objetivo del juego es sobrevivir a varios proyectiles que rebotan alrededor de la pantalla. Entre mayor tiempo sobreviva un jugador, mayor puntaje consigue.

El juego ocurre en una pantalla de fondo estático, el cual debe ser acorde a la temática seleccionada.

El jugador posee en total 3 vidas. Cada impacto de proyectil le restará una vida al jugador. Al llegar a 0 vidas el jugador pierde el juego.

Proyectiles:

Los proyectiles son objetos que se mueven alrededor de la pantalla de juego. El movimiento de estos objetos es determinado por una componente vertical y una componente horizontal. Cada vez que un proyectil colisiona con el extremo de la pantalla, invierte el signo de estas componentes y cambia la magnitud de cada uno por un numero generado aleatoriamente. Lo anterior significa que los meteoritos se

encontrarán rebotando alrededor de la pantalla en trayectorias relativamente impredecibles.

Al decidir qué rango utilizar para la generación de números aleatorios, recuerde que la prioridad es que el juego sea entretenido.

Los proyectiles no colisionan entre sí. Si colisionan con el jugador, el jugador pierde una vida y el proyectil se destruye

Niveles:

El juego consiste de tres niveles distintos. La cantidad de proyectiles a esquivar en cada nivel es mayor que en el nivel anterior. Queda a decisión del desarrollador la esta variación de cantidad. Recuerde que el juego debe ser retador, pero no imposible. En sí, se esperan al menos un nivel relativamente fácil, uno intermedio, y un nivel difícil.

Puntaje:

La cantidad de puntos por cada segundo varía según cada nivel. En el nivel fácil, es 1 punto por segundo. En el nivel intermedio, son 3 puntos por segundo. Finalmente, en el nivel difícil, se le confieren 5 puntos al jugador por cada segundo transcurrido. Cada nivel dura un minuto.

Si al finalizar el juego (complete o no el nivel) el puntaje de jugador supera alguno de los mejores anteriores, se le comunica por medio de un mensaje cuál posición obtuvo y con cuántos puntos llegó a dicha posición.

Sonido:

Cada nivel deberá tener su propio tema musical acorde al tema escogido. La pantalla principal también tiene su propio tema musical.

Cada colisión de un proyectil contra los extremos de la pantalla causa un sonido de impacto. Este sonido debe ser acorde al tipo de objeto seleccionado como proyectil, por ejemplo: si se usa un meteorito se espera el sonido de una roca chocando, si se usa un cubo de metal, se espera un sonido de un metal chocando.

Pantallas:

Toda pantalla que no sea la de inicio debe proveer alguna manera de volver a la pantalla de inicio

Pantalla de inicio:

En esta pantalla el jugador puede acceder a cualquier otra pantalla del juego por medio de botones. Además de los accesos a otras pantallas, se muestra una caja de texto y una selección de nivel. La caja de texto (Entry) se usará para permitirle al jugador ingresar el nombre con el que se registrará su puntaje. Los selectores de

nivel permitirán empezar por el nivel que el jugador escoja, aunque el valor por defecto será el nivel fácil.

Pantalla complementaria o About:

En esta ventana se mostrará la información complementaria. Se deben incluir los siguientes datos:

- País de producción
- Universidad y Carrera
- Asignatura, año que cursa y grupo
- Nombre del profesor
- Versión del programa
- Autor(es) y Fotos
- Autores de módulos modificados/utilizados por usted
- Instrucciones o datos que considere importantes para el uso del programa

Pantalla Mejores puntajes

Esta pantalla contiene una lista de los 10 mejores puntajes obtenidos. Estos puntajes se leen desde un archivo .txt, el cual es procesado como un archivo secuencial.

Cuando un jugador termina una partida, debe cargar el archivo .txt, comparar los resultados guardados y actualizar el archivo en caso de ser necesario. Si luego de terminar una partida se obtiene uno de los 10 mejores puntajes, esto tiene que mostrarse al volver a la pantalla principal y abrir la ventana de mejores puntajes.

Se proveen dos formas alternativas de ordenar la lista de mejores puntajes. Una interacción hace que la lista se ordene en forma descendente según puntaje usando el algoritmo Quicksort, otra interacción ordena según nombre de jugador utilizando Insertion sort. La implementación de éstas funciones las debe realizar con recursión. Como desarrollador usted puede decidir si estas interacciones son botones, presiones de teclado, o eventos capturados sobre el elemento gráfico en el que se encuentra la lista.

NO se permite el uso de bibliotecas de manejo de bases de datos o automatización de lectura de archivos secuenciales. Los datos de puntajes deberán ser manejados únicamente utilizando las funciones de acceso a archivos vistas en clase. Lo mismo aplica para bibliotecas de serialización de datos a archivos. Se espera que usted implemente un formato de texto propio.

Pantalla de juego

La pantalla de juego tendrá un fondo agradable a la vista y acorde al tema escogido. Además de los elementos de juego ya abarcados como proyectiles y jugador, deberán mostrarse los siguientes elementos de interfaz gráfica:

- Puntaje
- Tiempo transcurrido
- Nombre del jugador
- Vidas del jugador
- Botón de regreso a pantalla principal (automáticamente aborta el juego, procesar el puntaje es opcional en este caso)

4. Aspectos Generales

- Se puede usar Tkinter o Pygame para desarrollar la interfaz gráfica.
- Cualquier indicio de copia o plagio significará un cero en la nota y la aplicación de las sanciones especificas en reglamento RREA.

- La documentación archivo(s) formato **.doc** consta de dos partes: Documento Técnico Ejecutivo y Documento Administrativo del Proyecto (Aplica solo para grupos).
    - El **Documento Administrativo Trabajo en Equipo** del Proyecto, es donde se establecen claramente las reglas del grupo, los roles de cada integrante, asignaciones o actividades, responsable, fecha preliminar y efectiva de entrega, la completitud de la misma. En este documento se entregan los rubros de evaluación para los integrantes y la evaluación justificada por semana respectiva.
    - El **Documento Técnico Ejecutivo** (archivo **.pdf**) deber contar con lo siguiente: Introducción, Conclusiones, Recomendaciones, Diagramas de la Arquitectura, Diagramas de Módulos, Plan de Pruebas, Análisis de Resultados y literatura o fuentes consultadas.
- Se debe enviar el software el día Miércoles 16 de Junio de 2021, además, por aparte en dos asignaciones específicas, enviar también la documentación: 1. **Documento Administrativo Trabajo en Equipo** y 2. **Documento Técnico Ejecutivo** (incluir bitácora) vía el Classroom, a más tardar a las 23:30. Se deben entregar todos los componentes software necesarios para probar todos los comandos de forma que demuestre pleno funcionamiento.
- Debe indicarse un archivo readme. txt con la versión de Python (>=3.3) a utilizar para la revisión, alguna otra indicación que se considere importante. Este archivo contiene el nombre y la descripción de una función o procedimiento especial designado mi_auto_doc(), que genera de forma automática (con print de Pyhton), todas las autodocumentaciones de los principales módulos (mínimo 5).

- Cada archivo base debe enviarse el nombre construido de la siguiente forma: Iniciales de los nombres (de los autores) en mayúsculas, guion bajo primer apellido, guion bajo segundo apellido y extensiones txt y doc.
- Cualquier duda, omisión o contradicción en la especificación, se debe aclarar con el profesor y esta se difundirá, a través del Classroom de Google.
- Toda la documentación formará parte de la defensa.
- El código debe estar suficientemente documentado de tal forma que los autores(as), se puedan orientar fácilmente durante la defensa.
- El proyecto se debe defender previa cita con el profesor y el asistente. El profesor o asistente publicará unas fechas de defensa. Ud debe inscribirse para defender el proyecto, de no asistir a la defensa, únicamente obtendrá la nota correspondiente a la documentación del proyecto.
- Las funciones de ordenamiento dentro de la lógica del programa deben ser desarrolladas recursivamente, no se aceptará con iteración. Desviarse de ésta directriz implicará la anulación de esa parte del código mediante comentarios en él y su correspondiente pérdida de puntos.
- Cualquier clase de copia de código será sancionada de acuerdo con el reglamento vigente y se llevará hasta la consecuencia de amonestación con carta al expediente. Código adoptado para el manejo de interfaz debe especificarse claramente la fuente y reconocer los créditos. Está prohibida la copia de código que involucre la solución lógica general del algoritmo.
- Si en la defensa no demuestra su autoría con el dominio propio de esa calidad solo se le otorgarán los puntos correspondientes a los obtenidos en la documentación.

4.1. Evaluación

Nombre_____Carnet_____

Nombre_____Carnet_____

Project I: Moon to Mars

Tabla de evaluación

Rubro	Valor	Obtenido
Aspectos Generales (28 pts)		
Pantalla Inicial • Captura del nombre • Acceso a otras pantallas • Botón/acceso a inicio de juego • Selección de dificultad inicial	4	
Pantalla Información complementaria • Información completa • Fotos de los programadores • Retorno a pantalla de inicio	3	
Pantalla de mejores puntajes • Muestra mejores puntajes • Se actualiza correctamente • Mejores puntajes se mantienen al cerrar y abrir el juego • Implementa algoritmo de ordenamiento Quicksort (según valor de puntaje) • Implementa algoritmo de ordenamiento Insertion sort(según nombre de jugador) • Permite cambiar el criterio de orden de lista y el mismo se actualiza correctamente	18	
Sonidos • Sonidos asociados a colisiones de proyectiles • Música de fondo en pantalla de juego y pantalla principal • Manejo de cambios de música de fondo al cambiar de pantalla	3	
Aspectos de juego (34 pts)		
Pantalla de juego • Muestra indicador de puntaje • Muestra indicador de tiempo • Muestra vida del jugador • Muestra nombre del jugador • Se le comunica al jugador cuando pierde/gana • Si el jugador obtiene uno de los 7 mejores puntajes, se le comunica el puntaje y su posición	8	

• Botón de regreso a pantalla principal • Muestra fondo agradable y acorde al tema seleccionado		
Jugador • Jugador puede moverse en las 4 direcciones sin salirse de la pantalla • Animación de sprite del jugador • Jugador pierde al tener su vida reducida a 0	6	
Dificultad • Aumenta cantidad de proyectiles según dificultad • Tiempo de juego de un minuto por nivel • Aumenta cantidad de puntos obtenidos por segundo según dificultad • Conserva la cantidad de vidas y puntaje obtenido en nivel anterior al cambiar de nivel	8	
Colisiones • Detecta colisiones jugador – proyectil • Proyectil cambia su trayectoria al colisionar según lo indicado en la especificación • Proyectil es eliminado al impactar con jugador	12	
Documentación 38 pts		
Introducción	1	
Conclusiones	5	
• Recomendaciones	4	
Análisis de resultados • Diagrama de módulos • Plan de pruebas	4	
Bitácora, Programación Recursiva/Iteración	6	
Literatura o Fuentes	1	
Módulo de Generación de Autodocumentación de los Módulos Principales	2	
Auto documentación de módulos sigue formato	3	
Documentación Interna en partes de los módulos más importantes	2	
Documentación de Reglas de Grupo y Roles	5	
Actividades, fechas de entrega y coevaluación	5	

Nota: Para las personas que realicen el proyecto individual los puntos de los dos últimos rubros se distribuirán de la siguiente forma, en: conclusiones (4 para un total de10), recomendaciones (4 para un total de 9) y bitácora (2 para un total de 6).