Computer Engineering

(CEAB Substantially Equivalent Program and CFIA Accredited Seals)

Project I:

Stack and Tail Recursion

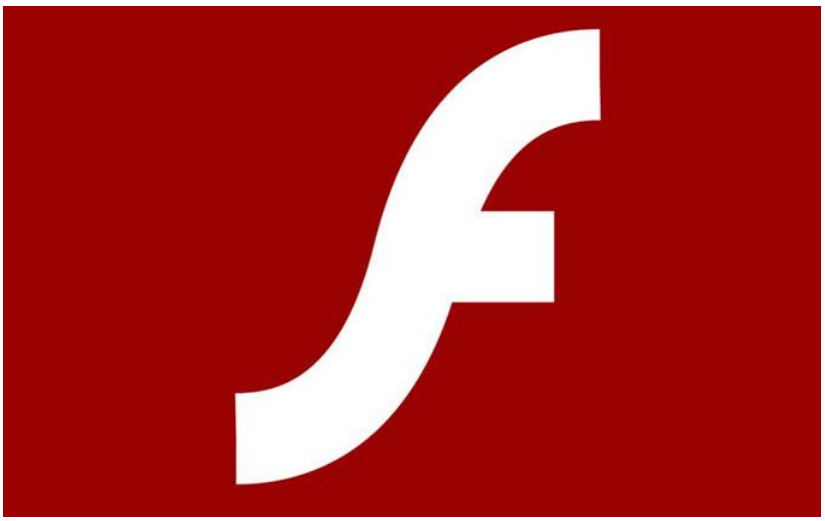
Flashback with Python

To Flash’s happy days: BulletHead

Professor:
Milton Villegas Lemus

"Failure is an option here.
If things are not failing, you are not innovating enough."
"When something is important enough,
you do it even if the odds are not in your favor."
Elon Musk

“Solve it. Solve it quickly, solve it right or wrong. If you solve it wrong,
it will come back and slap you in the face, and then you can solve it right.
Lying dead in the water and doing nothing is a comfortable alternative
because it is without risk, but it is an absolutely fatal way to manage a business.”
Thomas J. Watson



Objetivo General

- Construir un proyecto que integre conceptos y etapas básicas del desarrollo de Software. Aplicar conceptos básicos de proyectos propios de Ingeniería en Computadores.

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.

Antecedentes

El precursor de Flash fue un producto llamado SmartSketch desde el año 1993. lanzado por la compañía FutureWave Software fundada por Charlie Jackson y Jonathan Gay. Se dice que Jonathan Gay comenzó a programar en una apple II en donde desarrolló una aplicación en lenguaje Basic de Space Invaders. SmartSketch tuvo un giro al incorporar animación entre otras cosas en 1995 en lo que después la empresa Macromedia adquiriría para lanzar Flash 1.0 en 1996.

Flash como plataforma de software multimedia fue consolida por la empresa Adobe Systems desde 2005, usada entre otras cosas para animaciones, aplicaciones tanto de escritorio como para dispositivos móviles como llegó a permitir streaming de sonido y video, llegó a gozar de muy buena popularidad en el ambiente creativo de desarrollo de páginas web de grandes marcas como Nike Motorola, Disney y HBO ya que permitía interactividad.

Entre los Juegos desarrollados en esta plataforma AngryBirds y Adventure Quest.

BulletHead es un videojuego basado en Flash, de la compañía Adobe, para ser ejecutado en navegador o buscador de internet. El juego forma parte del género “shoot ‘em up” donde el jugador controla, como típicamente en aquella época de forma individual, un objeto o personaje donde se dispara contra multitud de enemigos.

. Es similar en concepto al clásico Space Invaders, sin embargo, incluye algunas mecánicas distintas que lo hacen particular. El juego fue publicado originalmente en mayo del 2010.

En este juego hay un personaje principal, controlado por el usuario, el cual dispara proyectiles que siguen una trayectoria vertical. Desde la parte superior de la pantalla descienden enemigos varios, los cuales buscan invadir el planeta del personaje principal. Para evitar esta invasión, el personaje principal debe abatirlos con disparos y evitar colisionar con ellos.

El juego tiene 20 niveles en total. El jugador tiene a su disponibilidad varios “power ups” que dejan caer los enemigos de manera esporádica.

La compañía que creó BulletHead, Nitrome, fue fundada en agosto del 2004 como un estudio independiente de desarrollo de videojuegos, por dos diseñadores gráficos ingleses, Matthew Annal y Heather Stancliffe.

Nitrome se convirtió en uno de los estudios más reconocidos en la escena de juegos basados en Flash. Varios de sus juegos fueron licenciados a gigantes de esta escena, un ejemplo de estos siendo Miniclip.

Descripción del proyecto

Para este proyecto, usted deberá desarrollar un juego similar en concepto a BulletHead, con diseño y temática distintivos propios de su de su creación. En el mismo, deberá utilizar programación recursiva para poder replicar un comportamiento similar al que se observa en el juego original.

Imagen 1. Pantalla de información complementaria de BulletHead

Descripción de pantallas:

I. Pantalla Principal

Desde la pantalla similar a la mostrada en la imagen 2, se puede acceder al resto de pantallas del programa. Tiene un fondo agradable y música apta para la temática del juego. Captura el nombre del jugador antes de iniciar una partida, y permite seleccionar el nivel de dificultad inicial.

Imagen 2. Pantalla principal de juego original

II. Pantalla Información complementaria y opciones

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
- Autores de módulos modificados/utilizados por usted
- Instrucciones o datos que considere importantes para el uso del programa

Además de esto se muestran las teclas con las que se controla al personaje principal y se permite al usuario cambiarlas. Estas modificaciones no se guardan en un archivo aparte, sino que permanecen guardadas en memoria
Se debe permitir retornar a la pantalla inicial.

III. Pantalla de mejores puntajes

En esta pantalla, desde un **archivo secuencial**, se carga la información correspondiente a los mejores puntajes. Se deben mostrar las cinco mayores puntuaciones obtenidas hasta el momento, y deben estar ordenadas de mayor a menor, de forma descendente.

Si durante la corrida del programa se llega a obtener un puntaje mayor, el mismo debe aparecer actualizado la próxima vez que se vuelva a abrir esta pantalla.

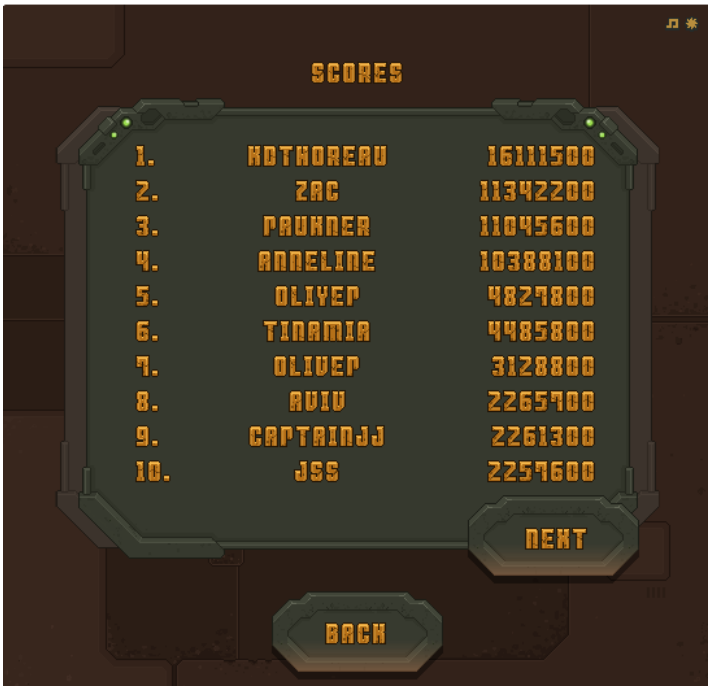


Imagen 3. Pantalla de mejores puntajes en el juego original

IV. Pantalla de juego

Además de la funcionalidad del juego en sí, en esta pantalla se debe mostrar:

- Puntaje actual
- Tiempo de juego
- Cantidad de vidas restantes
- Nombre del jugador

Descripción técnica del juego:

Jugador

El jugador puede realizar las siguientes acciones

- Disparar
- Saltar
- Moverse a la izquierda
- Moverse a la derecha

Si la tecla de disparo se mantiene presionada; el jugador no dispara hasta que la tecla de disparo se libere, de esta manera, se mantiene interesante la mecánica de disparo,

puesto que se quiere imitar un comportamiento similar al de un láser. El objetivo principal es que el juego sea divertido, por lo cual este tipo de comportamientos deben evitarse

En cuanto al movimiento, el jugador no debe ser capaz de sobrepasar los límites de la pantalla de juego ni de saltar mientras se encuentra en el aire

En total, el jugador posee tres vidas, en caso de perderlas todas, se le comunica que ha perdido el juego. Si su puntaje resulta ser uno de los mayores hasta el momento, se le comunica por medio de un mensaje de texto y se actualiza el archivo que contiene los puntajes.

Enemigos

Hay tres tipos de enemigos que se crean de manera dinámica en la parte superior de la pantalla

- **Misil en caída libre:** Este enemigo es el más veloz. No dispara, pero si impacta al jugador, le quita 2 puntos de vida. Tienen un punto de vida

- **Enemigo común:** Este enemigo posee 3 puntos de vida al igual que el jugador, dispara una vez por segundo. Cada disparo de este enemigo le resta 1 punto de vida al jugador. No desciende directamente, sino que “serpentea” por la pantalla

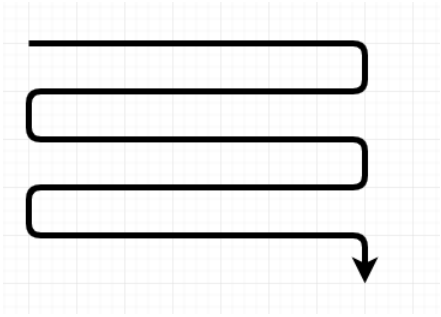


Imagen 4. Trayectoria de enemigo común

- **Enemigo aplastador:** Este enemigo aparece en la parte superior de la pantalla, desciende rápidamente hasta el piso, se mueve a uno de los bordes horizontales de la pantalla, y dispara en el sentido contrario una vez. El jugador debe saltar para evitar ser golpeado por este enemigo o por su disparo. Cualquiera de los dos ataques le resta un punto de vida al jugador. Cuando el jugador evita el ataque, el enemigo desaparece. Tiene 5 puntos de vida

Imagen 5. Ejemplo de enemigo común

Si el jugador abate un misil, obtiene 1 punto, si abate un enemigo común, obtiene 5 puntos, si abate un enemigo aplastador obtiene 20 puntos, si evita a un enemigo aplastador obtiene 2 puntos

Niveles

Hay tres niveles de dificultad en total. La transición entre estos se da de manera natural durante el juego, aunque desde la pantalla principal se puede seleccionar comenzar desde cualquier nivel. Cada nivel tiene una duración de 40 segundos, y la diferencia respecto a los demás es la cantidad de enemigos que aparecen por unidad de tiempo. El aumento en dificultad debe ser notable, pero tampoco debe volver al juego imposible de ganar.

Notas:

- El Proyecto es individual. Se debe enviar el software el día **Miércoles 15 de Julio de 2020**, por aparte en una asignación específica enviar la documentación (incluir bitácora) vía el Classroom, a más tardar a las 23:30. Debe 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 3).
- Cada archivo base debe enviarse el nombre construido de la siguiente forma: Iniciales de los nombres (del autor) 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.
- La documentación (archivo .doc) debe contar con lo siguiente: **Introducción**; **Conclusiones**; **Recomendaciones**; **Análisis de Resultados**, dentro de este punto incluir: Diagramas de Módulos, Plan de Pruebas de los módulos principales; y **Literatura o Fuentes Consultadas**.
- Toda la documentación formará parte de la defensa.
- El código debe estar suficientemente documentado de tal forma que el autor(a), se pueda 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 repetición 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.

Literatura y Fuentes Consultadas

[1] Recuperado Feb(2020).http://www.gamesindustry.biz/articles/2013-05-17-miniclip-interview
[2] Recuperado Feb(2020). https://nitrome.fandom.com/wiki/Bullethead
.
[3] . Recuperado Feb(2020). http://www.nitrome.com/games/bullethead/
[4] Recuperado Jun(2020). https://code.tutsplus.com/articles/a-nostalgic-rummage-through-the-history-of-flash--active-6733

Tabla de evaluación

Rubro	Valor	Obtenido
Aspectos Generales (20 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 • Retorno a pantalla de inicio	4	
Pantalla de Mejores Puntajes • Muestra mejores puntajes • Se actualiza correctamente • Mejores puntajes se mantienen al cerrar y abrir el juego	12	
Aspectos de juego (48 pts)		
Pantalla de escenario de batalla • Muestra fondo de la temática escogida • Muestra puntaje • Muestra tiempo de juego • Muestra nombre del jugador • Se le comunica al jugador cuando pierde o gana • Si el puntaje del jugador es uno de los mejores 5, se lo comunica inmediatamente al perder o ganar y se le dice en cual posición quedó	12	
Movimiento de jugador • Jugador puede moverse en ambas direcciones • Jugador no se sale de los límites de la pantalla • Jugador puede saltar cuando se encuentra en el suelo • Jugador no puede saltar mientras se encuentra en el aire	8	
Colisiones • Enemigos reciben daños al ser impactado por un proyectil del jugador • Jugador pierde una vida al ser impactado por un enemigo	10	
Enemigos • Se crean enemigos de manera dinámica • Se implementan los tres patrones de movimiento especificados • La cantidad de enemigos se distribuye adecuadamente en la duración del nivel	9	

Dificultad (3 niveles mínimo) • Aumenta cantidad de enemigos al subir el nivel • Frecuencia de disparo limitada (no hay efecto "láser" al lanzar un proyectil) • Tiempo mínimo de 40s entre cada nivel	9	
Documentación 32 pts		
Introducción	1	
Conclusiones	6	
Recomendaciones	5	
Análisis de resultados • Diagrama de módulos • Plan de pruebas	4	
Bitácora	4	
Literatura o Fuentes	1	
Módulo de Generación de Autodocumentación de los Módulos Principales	2	
Auto documentación de módulos sigue formato	6	
Documentación Interna en partes de los módulos más importantes	3	