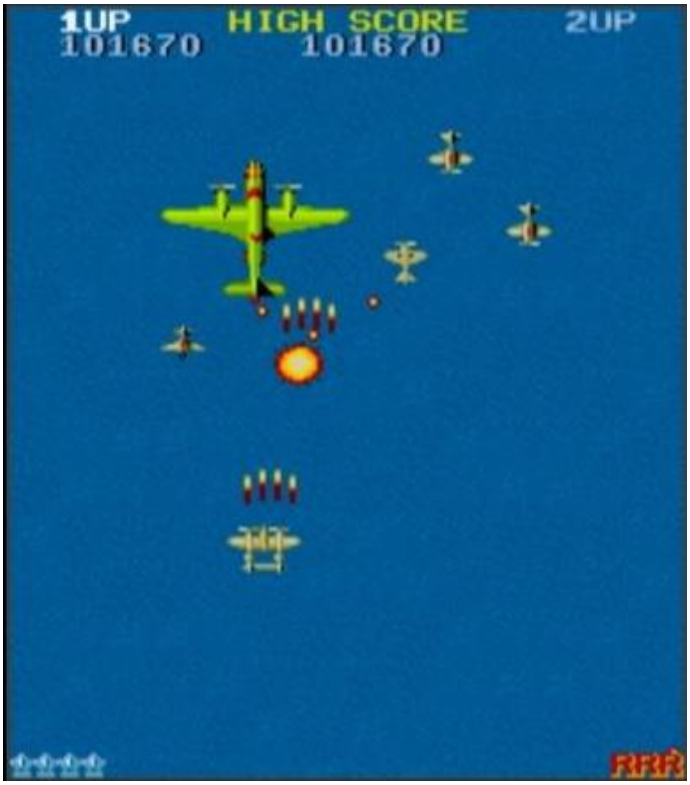
Proyecto Final: 1942
Programación
2022 - 2023

Para el Proyecto final los estudiantes implementarán una versión simplificada del juego 1942. El objetivo del juego es pilotar un avión de combate *Lockheed P-38* y abatir a todos los aviones enemigos.

1. Introducción al juego: Reglas Básicas



El juego a implementar es una versión simplificada del primer nivel de la versión original del juego 1942. Para completar el nivel, el avión debe alcanzar la parte superior de la pantalla evitando a todos los aviones enemigos y sus disparos, mientras abate tantos enemigos como sea posible. Para ello se cuenta con un máximo de 3 vidas o intentos.

A continuación se detallan los diferentes tipos de enemigos con los que habrá que combatir:

Enemigo Regular: Aparecerán en grupos de varios aviones, sin una formación específica, y podrán dispararte. Una vez que alcanzan la mitad de la pantalla podrán desaparecer y volver por la parte superior de la pantalla.

Enemigo Rojo: Aparecerán en grupo siguiendo una formación. Habitualmente dan 2 o 3 vueltas hasta que desaparecen. Normalmente no disparan. Si se les destruye, obtendrás ciertas ventajas, como la capacidad de disparar doble.

Bombardero: Es más grande que el resto de aviones. Aparece solo, dispara varias veces y requiere varios impactos para ser destruido.

Superbombardero: es el avión más grande, requiere muchos impactos para ser destruido y tiene la capacidad de disparar varios proyectiles simultáneamente.

La puntuación aumenta en función del número de aviones destruidos. Los bombarderos proporcionan mayor puntuación que los aviones regulares.

Hay varios sitios web donde se puede jugar a este juego. Seguiremos como ejemplo la siguiente versión:

https://www.retrogames.cc/arcade-games/1942-first-version.html

Podréis encontrar un video completo del juego original en el siguiente enlace:

https://youtu.be/1yMGtlNjcMY

2. Trabajo a desarrollar

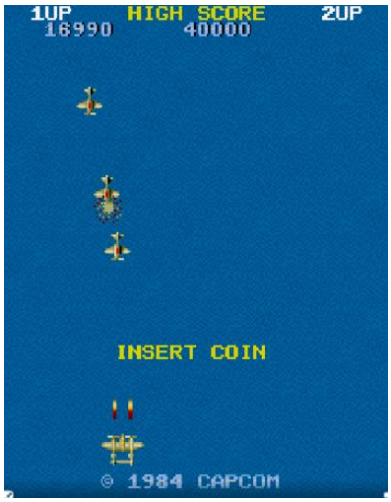
Habrá que programar el juego, incluyendo su interfaz gráfica. El diseño gráfico será una tarea colaborativa en la cual toda la clase creará los gráficos como ficheros pyxres o png usando el editor de gráficos de pyxel y creando con ellos un repositorio común en Aula Global.

Los alumnos podrán cambiar dichas imágenes por las suyas propias. Se recomienda no invertir demasiado tiempo en esto, puesto que se valorará la correcta implementación del juego utilizando el paradigma de programación orientada a objetos, y no la apariencia visual del mismo.

Para implementar el juego de forma correcta es esencial haber realizado un buen diseño de las clases a utilizar. Antes de escribir ningún código, debéis estar seguros de que vuestro diseño de clases es el correcto comentándolo con vuestro/a profesor/a de laboratorio. No comencéis ninguna tarea de implementación de código hasta que el diseño sea aprobado por vuestro/a profesor/a.

Una vez que la jerarquía de clases ha sido definida, la implementación será una tarea incremental, completando las funcionalidades del juego paso a paso. Se recomienda seguir los siguientes pasos, aunque cada alumno podrá elegir su propia estrategia de implementación, no habiendo ningún control semanal ni seguimiento de ello.

2.1 Sprint 1: Objetos e Interfaz Gráfica



Crear una clase para cada elemento del juego con los atributos adecuados. Toda la lógica del juego deberá ser implementada dentro de cada clase. Será penalizado incluir más código del necesario dentro del programa principal.

Crear el nivel: Colocar tu avión, crear una lista con los enemigos, que no aparecerán inicialmente, sino que aparecerán progresivamente. Al menos deberán crearse 20 aviones regulares, 5 aviones rojos, 2 bombarderos y 1 superbombardero, pero se podrán también introducir variantes tratando de imitar el juego original. Añadir también elementos adicionales del juego, como islas. Añadir también marcadores para mostrar la puntuación obtenida durante el juego y otra para mostrar la puntuación más alta.

2.2 Sprint 2: Movimientos básicos del avión y disparos

Permitir al avión moverse en cualquier dirección y disparar. Animar también las hélices del avión.

2.3 Sprint 3: Enemigos

Hacer aparecer a los enemigos e implementar sus movimientos básicos. Puedes hacerlos aparecer aleatoriamente o siguiendo el patrón original del juego. Hacerlos disparar aleatoriamente.

En este sprint no es necesario que el avión interactúe con los enemigos, ni tampoco animar sus movimientos.

2.4 Sprint 4: Juego básico

Implementar el juego básico:

- Cuando un avión enemigo es alcanzado, será destruido y la puntuación aumentará. Los bombarderos y los superbombarderos necesitarán varios disparos para ser destruidos.
- Si el avión es alcanzado por un disparo enemigo, será destruido y se consumirá una vida.
- Presionando la tecla z el avión hará un loop para evitar ser abatido.

Implementar en este sprint las animaciones de los aviones enemigos.

2.5 Sprint 5: Elementos extra (opcional)

Añadir bonus extra cuando se destruyen todos los aviones rojos de una vez: disparos dobles, loops extra, etc.

Añadir nuevas funcionalidades al juego: música y efectos de sonido, una tabla con las puntuaciones más altas, animaciones en el comienzo y final del juego, etc. Verificad con el/la profesor/a de prácticas antes de hacerlo para recibir la orientación adecuada.

3. Normas de Entrega

Las siguientes normas son **obligatorias**. No seguirlas puede suponer que el trabajo no sea evaluado. El proyecto final se debe realizar en grupos de 2 alumnos. Cada grupo debe colgar en Aula Global un fichero zip que contenga la memoria y el código fuente antes de la fecha límite. El nombre del fichero zip debe seguir el siguiente formato: "iniciales1-iniciales2.zip" (por ejemplo, si los alumnos son Lucía Pérez Gómez y Juan García Jiménez, el nombre del fichero debe ser lpg-jgj.zip).

La memoria, en **formato PDF,** contendrá un máximo de **10** páginas, que deberán incluir al menos:

- Portada, tabla de contenidos y un breve resumen del documento.
- Descripción de las clases diseñadas, incluyendo los atributos y métodos principales.
- Descripción general de los principales algoritmos utilizados.
- Descripción del trabajo desarrollado, funcionalidades incluidas, partes no implementadas, y funcionalidades extra llevadas a cabo.
- Conclusiones:
    - Resumen final del trabajo realizado.
    - Principales problemas encontrados.
    - Comentarios personales, valoración y aspectos a mejorar del proyecto final.

4. Evaluación

El Proyecto final será valorado entre 0 y 2.5 puntos. Cada sprint de 1 a 4 será valorado con 0.5 puntos. La memoria será valorada con los 0.5 restantes. La parte opcional será valorada con otros 0.5 puntos extra, con 0.1 puntos por cada funcionalidad relevante añadida, aunque nunca se podrá superar la puntuación máxima de 2.5 puntos.

Los siguientes aspectos serán tenidos en cuenta a la hora de valorar el proyecto final:

- Apropiado diseño de clases.
- Correcto funcionamiento del juego y seguimiento adecuado de las normas fijadas en este documento.
- Calidad del código implementado.
- Uso exclusivo de los contenidos y principios impartidos durante el curso.
- Inclusión de comentarios en el código.
- Descripción general del código implementado en la memoria, acorde a las normas especificadas en la sección 3.

- Cumplimiento de las normas de entrega (Sección 3).

Se aplicarán cuando procedan las siguientes penalizaciones:

- 0.25 puntos si no se han incluido suficientes comentarios. Todos los métodos y clases deberán ser debidamente comentados. También deberían incluirse comentarios dentro del código de los distintos métodos.
- 0.5 puntos si no se usan métodos, si se incluye código repetitivo en vez de bucles, si no se siguen las buenas prácticas de programación inculcadas durante el curso, si se lleva a cabo un mal diseño de las clases, si no se usan listas para representar a los enemigos, si se usan breaks, …
- 0.25 puntos si se incluyen atributos innecesarios, atributos públicos que deberían ser privados, atributos creados fuera del método init, o no uso de determinadas propiedades cuando deberían ser utilizadas.
- 0.25 puntos si se tiene código fuera de las clases (con la excepción de un módulo para incluir las constantes y el programa principal) o por tener métodos demasiado largos (como norma general un método debería ocupar una sola pantalla sin necesidad de hacer scroll).

Las copias serán duramente penalizadas: tanto los copiadores como los copiados serán excluidos automáticamente de la evaluación continua, aparte de posibles sanciones administrativas que la Universidad pudiera ejercer. Dos proyectos podrán ser considerados como copia incluso si su grado de parecido se restringe al código de un único método. Se utilizarán herramientas automáticas de detección de plagios.

El Proyecto final será evaluado mediante una presentación oral en la que ambos alumnos deberán defender y exponer su trabajo ante el profesor.