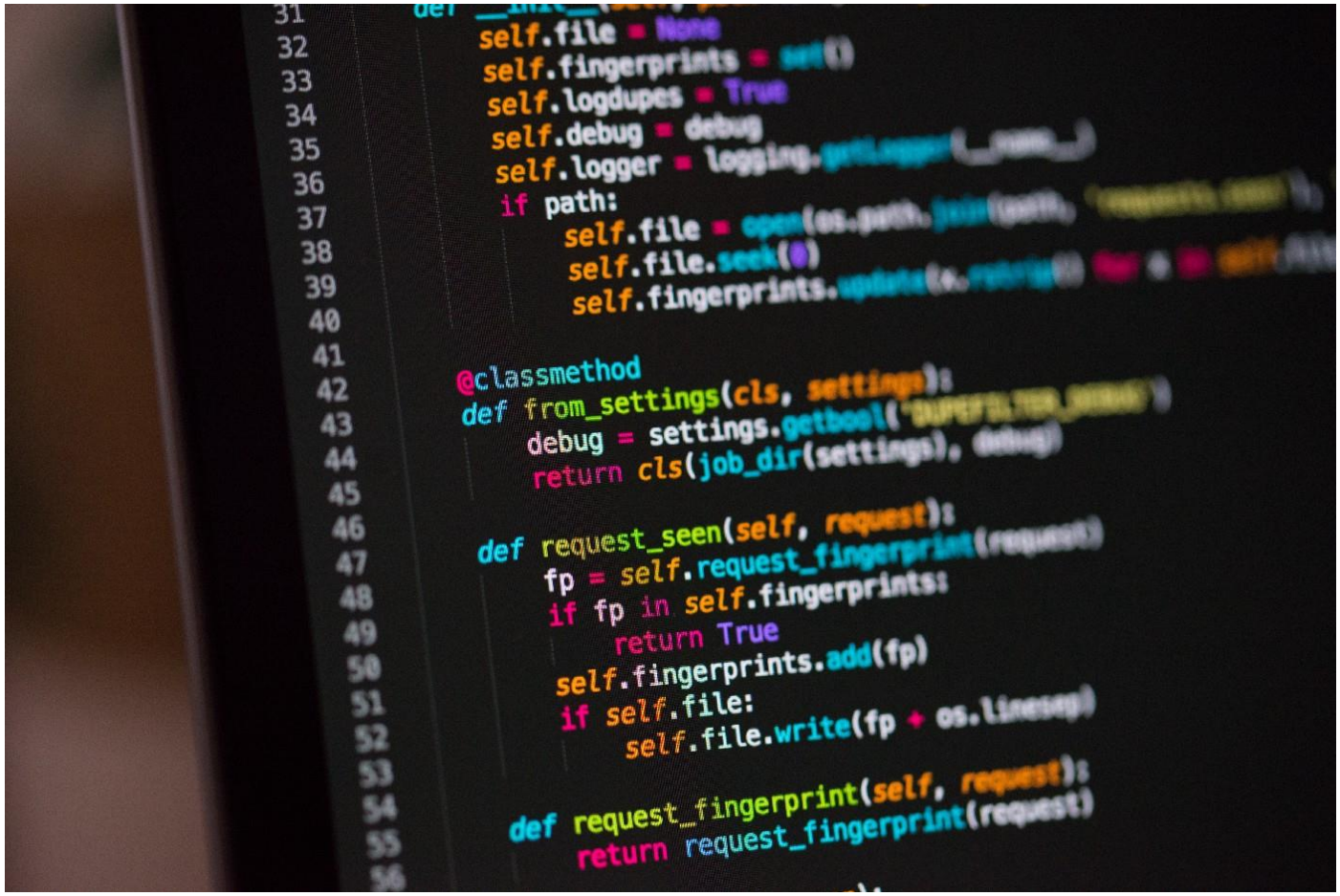
# Documentación sobre nuestro proyecto final:

# Draw Adventure




## CUCEI

**(Centro Universitario de Ciencias Exactas e Ingenierías)**

Fundamentos de programación

Integrantes del equipo:

Leonardo Zaid Pacheco Sánchez

Fabiola González Mora

César Noé Ascencio Palma

# Índice

# Abstract

In this project we will create a RPG style video game called Draw adventure based on final fantasy, we will be using visual studio code, and it´ll be programmed with python, in conjunction with the pygame module, which allows python to run graphic interfaces and add sounds and animations. In this report we will show in a detailed manner how we were able to complete this Video game step by step.

# Introducción

Se estuvo buscando una forma de realizar un proyecto con el cual poder utilizar todos los métodos y nuestros conocimientos aprendidos a lo largo del semestre en la materia introducción a la ingeniería. Por lo que platicando se llegó al acuerdo de que realizando un videojuego se podría implementar todo lo aprendido a lo largo de la materia de igual manera con conocimientos previos y habilidades de cada uno de los integrantes y mucha investigación se hizo posible realizar el videojuego. El cual se realizó en su mayoría en el periodo vacacional de semana santa y pascua.

El videojuego se le dio el nombre de Draw Adventure (Aventura de dibujo) el cual es un tipo de juego rpg por turnos, en el que se te da la opción de atacar o tomar una poción para recuperar puntos de vida y seguir con el juego hasta el momento de la victoria o de la derrota. De igual manera en este juego tanto el usuario como los enemigos tiene dos estadísticas principales que es la vida y el daño. La vida es la cantidad de daño que puede recibir antes de morir y el daño es el daño que se le puede llegar a dar al adversario y este se le resta de su vida máxima.

# Planteamiento del problema

Se buscaba implementar todo lo que se llegó a ver en la clase de introducción a la programación. Se llegó a la decisión de que se haría un videojuego para el cual se ocupaba una extensa investigación para poder realizarlo y que tenga una buena funcionalidad, se llevó a cabo investigaciones tanto en páginas de internet como videos tutoriales de programación en YouTube. De igual manera se crearon las animaciones y la música para el videojuego.

# Objetivo

Se analizó detalladamente todo lo visto en la clase y se propuso la creación de un videojuego en el cual se utiliza todo lo visto en dicha clase. Un videojuego que sea completamente funcional con características de RPG con el cual se pueda mostrar como los personajes pueden llegar a la victoria o a la derrota y volver a reiniciar el juego una vez finalizado.

# Metodología

## Organización

Primero se habló de cómo nos pondríamos de acuerdo para investigar sobre lo que se necesitaba para saber lo que se necesitaba para el videojuego. Se llegó al acuerdo de que cada uno investigaría por su parte y cuando hiciéramos una reunión virtual cada uno mostraría lo que se investigó y veríamos si con eso era suficiente para empezarlo en la cual se dijo que para poder crear una interfaz o una ventana donde se mostraría el juego se necesitaba una librería llamada pygame, la cual se inténtalo atravesó del pip. También se habló sobre cómo

podrías hacer que todos viéramos a la vez que creamos o editamos el código y para ello se encontró una extensión de visual studio code llamada Live Share la cual con esa extensión se facilitó el trabajo que hacíamos reuniones virtuales y a la vez podíamos editar el código en tiempo real. También se investigó el tipo de programación que llevan los videojuegos desarrollados en Python. Se llegó a la conclusión que era programación orientada a objetos y se investigó a fondo sobre la programación orientada a objetos.

## Planificación

Una vez que se tenía toda la información para empezar con la creación del juego se empezó a hablar sobre cómo se llevaría a cabo y sobre que trataría, es decir, de qué trataría se pensó de varias formas la primera opción fue hacerlo únicamente por medio de texto e inputs para el usuario, en el cual no hubieran referentes gráficos o animaciones, luego se llegó a la conclusión de implementar animación y si se nos complicaba hacerlo lo haríamos como se había planeado al principio, solamente inputs y texto.

Se realizaron investigaciones de tipo exploratoria en la cual se averiguo todo lo que era desconocido como las clases y sus atributos, los efectos de sonido, las animaciones hechas por imágenes el cómo cargarlas, agregarlas y darles una ubicación en la interfaz del videojuego, el cómo se usaba live share, cómo agregar botones, el cómo convertir una imagen a texto y se analizó la información viendo cual sería útil y cuál no, también se investigó la forma en que está hecho un videojuego de tipo rpg y como se conforman sus componentes hasta comprender la forma de su funcionamiento.

También se realizó una investigación descriptiva y explicativa en las cuales se compartió la información obtenida y se explicó

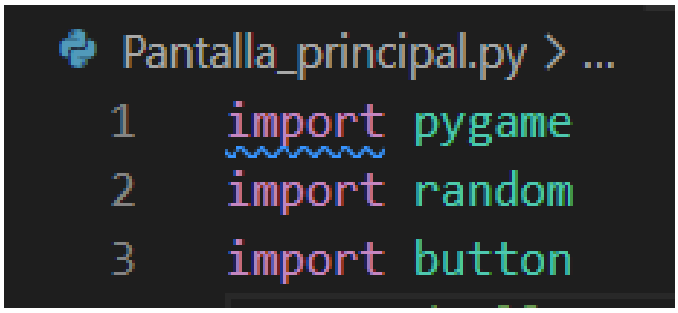
lo que se comprendió de dicha información. De igual manera se hizo una investigación correlacional en la cual se revisó y compartió la información obtenida. Con toda la investigación y conocimientos previos se empezó hacer y se terminó realizando el videojuego.

De igual manera se dialogó sobre la animación que tendría y se llegó a la conclusión de que sería un caballero contra dos bandidos.

El caballero tiene un lápiz que hace la función de una espada y los bandidos son 2 personajes que hacen referencia al juego del ahorcado.

## Ejecución

Primero se importan las librerías que son las de pygame, random; además de de un archivo extra llamado Button.




Pygame, es una librería que tiene todo tipo de facilidades para programar juegos en Python, como hacer interfaces gráficas, agregar animaciones y efectos de sonido y la librería de random solo se encarga de elegir números aleatorios u otras funciones relacionadas con la aleatoriedad.

Button es un archivo extra que se encarga de las colisiones del cursor para así poder seleccionar qué hacer en tu turno.

button.py > ...

```python
import pygame

#button class
class Button():
    def __init__(self, surface, x, y, image, size_x, size_y):
        self.image = pygame.transform.scale(image, (size_x, size_y))
        self.rect = self.image.get_rect()
        self.rect.topleft = (x, y)
        self.clicked = False
        self.surface = surface

    def draw(self):
        action = False

        #get mouse position
        pos = pygame.mouse.get_pos()

        #check mouseover and clicked conditions
        if self.rect.collidepoint(pos):
            if pygame.mouse.get_pressed()[0] == 1 and self.clicked == False:
                action = True
                self.clicked = True

        if pygame.mouse.get_pressed()[0] == 0:
            self.clicked = False

        #draw button
        self.surface.blit(self.image, (self.rect.x, self.rect.y))

        return action
```

Después se empezó a crear un código en el cual se definen las variables principales del juego, las cuales son el tamaño de la pantalla o interfaz, los cuadros por segundo del juego que son FPS, el nombre de la ventana del juego.

```python
clock = pygame.time.Clock()
fps = 60

#Ventana del juego
bottom_panel = 150
screen_width = 800
screen_height = 400 + bottom_panel

screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption('Draw adventure')
```

Después se crearon todas las funciones necesarias para poder correr el juego, además se crearon clases y atributos de estas clases del personaje, enemigos y de las pociones para poder agregar todo en un ciclo principal “while” donde corre todo el juego. Los atributos que creamos en la clase de los peleadores es su posición en la pantalla, vida, vida máxima, ataque, y número de pociones.

```python
#clase peleador
class Fighter():
    def __init__(self, x, y, name, max_hp, strength, potions):
        self.name = name
        self.max_hp = max_hp
        self.hp = max_hp
        self.strength = strength
        self.start_potions = potions
        self.potions = potions
        self.alive = True
        self.animation_list = []
        self.frame_index = 0
        self.action = 0 #0:idle, 1:attack, 2:hurt, 3:dead
        self.update_time = pygame.time.get_ticks()
        #cargar las animaciones idle
        temp_list = []
        for i in range(5):
            img = pygame.image.load(f'Recursos/{self.name}/Idle/{i}.png')
            img = pygame.transform.scale(img, (img.get_width() * 0.20, img.get_height() * 0.20))
            temp_list.append(img)
        self.animation_list.append(temp_list)
        #cargar las animaciones de ataque
        temp_list = []
        for i in range(3):
            img = pygame.image.load(f'Recursos/{self.name}/Attack/{i}.png')
            img = pygame.transform.scale(img, (img.get_width() * 0.20, img.get_height() * 0.20))
            temp_list.append(img)
        self.animation_list.append(temp_list)
        #Cargar imagenes de recibir dano
        temp_list = []
        for i in range(4):
            img = pygame.image.load(f'Recursos/{self.name}/Hurt/{i}.png')
            img = pygame.transform.scale(img, (img.get_width() * 0.20, img.get_height() * 0.20))
            temp_list.append(img)
        self.animation_list.append(temp_list)
        #Cargar imagenes de muerte
        temp_list = []
        for i in range(6):
            img = pygame.image.load(f'Recursos/{self.name}/Death/{i}.png')
            img = pygame.transform.scale(img, (img.get_width() * 0.20, img.get_height() * 0.20))
            temp_list.append(img)
        self.animation_list.append(temp_list)
        self.image = self.animation_list[self.action][self.frame_index]
        self.rect = self.image.get_rect()
        self.rect.center = (x, y)
```

Al final implementamos todos estos elementos en un ciclo while principal el cual se encarga de iniciar el videojuego y de finalizarlo cuando se alcancen las condiciones de game over, como vencer a los enemigos o que ellos derroten al jugador.

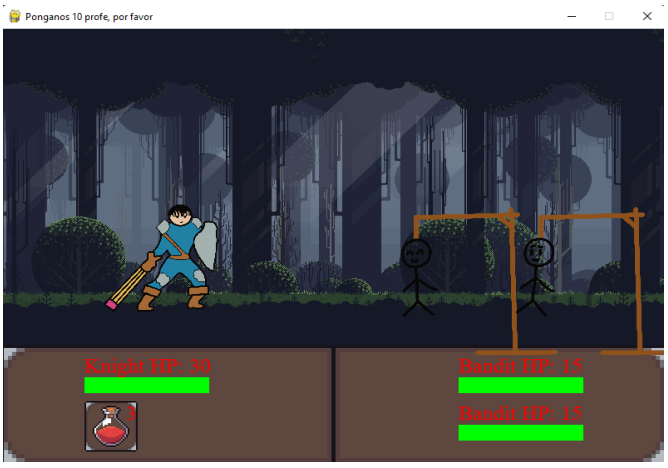
```
#Ciclo while principal
pygame.mixer.music.play()
run = True
while run:

    clock.tick(fps)

    #Aqui llamo la funcion para mostrar el fondo
    draw_bg()

    #Aqui llamo todas las funciones referentes al panel de abajo
    draw_panel()
    knight_health_bar.draw(knight.hp)
    bandit1_health_bar.draw(bandit1.hp)
    bandit2_health_bar.draw(bandit2.hp)

    #Aqui llamo las funciones para mostrar los personajes
    knight.update()
    knight.draw()
    for bandit in bandit_list:
        bandit.update()
        bandit.draw()

    #Mostrar el texto del daño hecho
    damage_text_group.update()
    damage_text_group.draw(screen)

    #Acciones controladas por el jugador
    #Resetear las varriables de accion
    attack = False
    potion = False
    target = None
    #Aqui nos aseguramos que el cursor del mouse sea visible
    pygame.mouse.set_visible(True)
    pos = pygame.mouse.get_pos()
    for count, bandit in enumerate(bandit_list):
        if bandit.rect.collidepoint(pos):
            #Esconder el cursor
            pygame.mouse.set_visible(False)
            #Mostrar espada en vez de cursor
            screen.blit(sword_img, pos)
```

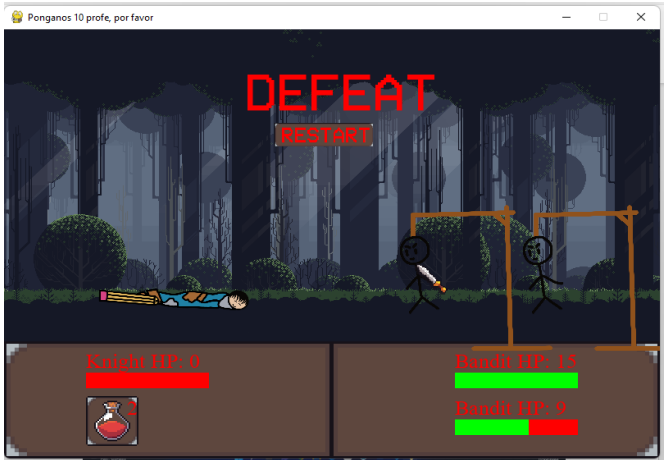
```
                else:
                    current_fighter += 1

        #Si todos los luchadores terminaron su turno se resetea al jugador
        if current_fighter > total_fighters:
            current_fighter = 1

    #Revisar si todas las bandidas murieron para aplicar la condicion de game over 1, que es la victoria
    alive_bandits = 0
    for bandit in bandit_list:
        if bandit.alive == True:
            alive_bandits += 1
    if alive_bandits == 0:
        #Quitar musica poco a poco
        pygame.mixer.music.fadeout(3000)
        game_over = 1

    #Revisar si el juego terminó
    if game_over != 0:
        if game_over == 1:
            screen.blit(victory_img, (250, 50))
        if game_over == -1:
            screen.blit(defeat_img, (290, 50))
        if restart_button.draw():
            knight.reset()
            for bandit in bandit_list:
                bandit.reset()
            current_fighter = 1
            action_cooldown
            game_over = 0


    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
        if event.type == pygame.MOUSEBUTTONDOWN:
            clicked = True
        else:
            clicked = False

    pygame.display.update()

pygame.quit()
```

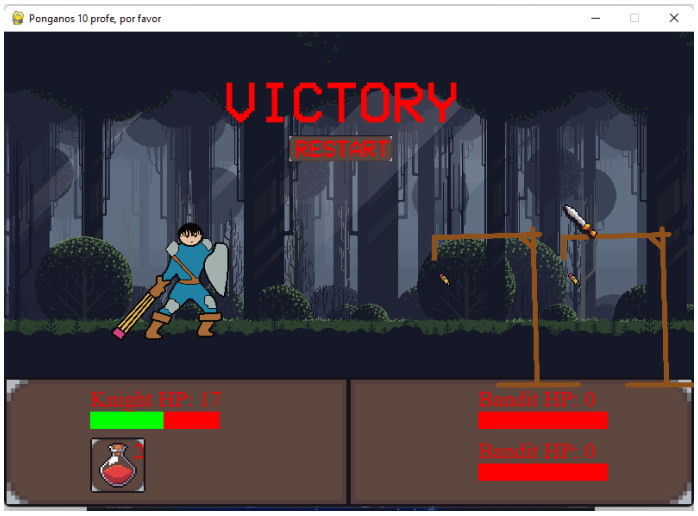
# Resultados

Después de un largo proceso que llevó en la creación se logró la creación del videojuego el cual se trata de juego rpg por turnos, en el que tienes la opción de atacar o tomar una opción para recuperar puntos de vida. En el que tienes una estadística de fuerza al igual que los enemigos que junto con un valor aleatorio de -5 a 5 es lo que determina el daño que le harás a los enemigos y que ellos te harán a ti.

Aquí se muestran imágenes del videojuego. La primera es de cuando recién inicias el videojuego, la segunda cuando pierdes y la tercera cuando ganas.



# Conclusiones

Durante el trayecto de este proyecto hemos aprendido muchas cosas importantes sobre la programación y métodos para la creación del proyecto. Una de las cosas más interesantes que nos dejó toda la investigación que llevamos a cabo fue la mejora de nuestra capacidad para adquirir nuevos conocimientos, la capacidad para lograr plasmar nuestra idea y poder solucionar los errores que se nos cruzaban en nuestro camino para lograr que funcionara el programa. Al realizar con éxito la ejecución del programa nos dimos cuenta de que, tal vez el hacer un videojuego no era tan fácil como lo creíamos al principio, en esa etapa en la que tuvimos la vaga idea al iniciar el proyecto, pero lo terminamos con gran éxito, aprendimos muchas cosas que antes de éste proyecto no sabíamos y que sabemos que nos ayudarán mucho en un futuro tanto en nuestra formación académica como en nuestra formación profesional, fue un largo camino lleno de tropiezos y complicaciones, pero al final logramos hacerlo funcionar juntos como equipo

# Referencias


- **Lozano Gómez, J. J. (s. f.). Programación orientada a objetos (POO) en python. J2logo. Recuperado 11 de mayo de 2022, de https://j2logo.com/python/tutorial/programacion-orientada-a-objetos/**

- **Sánchez Alberca, A. (s. f.). Programación orientada a objetos. Aprende con Alf. Recuperado 11 de mayo de 2022, de https://aprendeconalf.es/docencia/python/manual/objetos/**

- **Programación orientada a objetos. (2021, 17 agosto). IBM. https://www.ibm.com/docs/es/spss-modeler/SaaS?topic=language-object-oriented-programming**

- **Onda electromagnética. (s. f.). Scientific Committees. Recuperado 16 de mayo de 2022, de https://ec.europa.eu/health/scientific_committees/opinions_layman/artificial-light/es/glosario/mno/onda-electromagnetica.htm**

- **Guagliano, C. (2019). Programación en Python II: Programación orientada a objetos. RedUsers.**