Project Specifications

Specifications:

- Objective:
  - To create a simple game that would bring a nostalgia factor and release stress in this pandemic times.
- About:
  - The game, “Attack on Virus”, is designed to be a lot like the old shooting game “Space Invaders”. Where the player or in this case doctor throws syringes up into the air to kill the virus by holding the space bar and moving right or left with the A and W keys respectively. As there is no end the players could play as long as the virus doesn’t touch you. Players will also get points based of how many viruses they kill. As time progress the viruses descend will keep on going faster till it finally touches you.

Design Solution:

- Theme:
  The theme and idea of the game came from frustration of being stuck during quarantine when I was infected with covid. As a way to channel that frustration the theme was made to be viruses. I know it is very simple, but it was the best I could do.
- Code:
  I’ve split the code into a class and driver file. With one containing all the custom classes that was made and another was made to run the modules. Named “VirusClasses.py” and “VirusDrivers.py”.

  VirusClasses.py:

- The purpose of this file was to contain all the needed classes such as the ones that store each sprite needed in the game. This is also a much neater way of organizing things
- The module that was used was pygame.

```
1  import pygame, random, time
2    from pygame.locals import *
3    from pygame import mixer
```

  The mixer is to add audio so that players would be able to listen to music as they play.
- Contents of the file:
  The file consists of 6 classes: enemy, enemyspawner, boss, doctor, syringe and background.
    - Enemy class:

```
class Enemy(pygame.sprite.Sprite):
    def __init__(self, x , y):
        super().__init__()
        self.imageList = ["img/virus.png" , "img/virus1.png"] #gets the image the to be
        self.currentImagePosition = 0
        self.currentImage = pygame.image.load(self.imageList[self.currentImagePosition])
        self.surf = pygame.Surface((40,40))
        self.x = x
        self.y = y
        self.rect = self.surf.get_rect(center = (self.x ,self.y))
        self.direction = "left"
        self.imageReset = 0
```

```
def move(self, destroyed, playerScore, speed):
    if self.direction == "left":
        self.rect.move_ip(-speed, 0)
    if self.direction == "right":
        self.rect.move_ip(speed, 0)

#To tell that it moves left
def left(self):
    self.rect.top += 32
    self.direction = "left"

#To tell that it moves right
def right(self):
    self.rect.top += 32
    self.direction = "right"
```

- The init function contains the sprites of the enemies, where it is located, the direction it will move and how big is its surface/" hitbox". The super().__init__() there is to contain all the functions of the Sprite module hence making it easier to use.

- The move function here is to tell the sprites to move to the left if self.direction is equal to left and right if its equal to the right

- The left function and right function tell it so that when it slowly descends, they don't overlapped each other when they move to the left and the right.

```
def reset(self):
    self.rect = self.surf.get_rect(center = (self.x, self.y))

#Continuouly draws the sprites onto the window every 60 FPS
def draw(self, window):

    if self.imageReset == 60 and self.currentImagePosition == 1:
        self.currentImagePosition = 0
        self.imageReset = 0

    if self.imageReset == 60 and self.currentImagePosition == 0:
        self.currentImagePosition = 1
        self.imageReset = 0

    self.imageReset +=1
    self.currentImage = pygame.image.load(self.imageList[self.currentImagePosition])
    window.blit(self.currentImage, self.rect)
```

-The reset function lets it reset to its first position when the game is over and resets.

-The draw function draws the enemy sprite onto window that is later created. It would alternate from the 2 sprite images every 60 FPS and then it would be resetted.

- Boss Class
  Has Similar codes to the enemy function and works similarly like the enemy class too.

- EnemySpawner

```
class EnemySpawner(pygame.sprite.Sprite):
    virusList = []
    def __init__(self):
        self.edgeBuffer = 50
        self.virusBuffer = 20
        self.virusWidth = 32

        for i in range (self.edgeBuffer, 500, self.virusWidth + self.virusBuffer):
            for j in range (self.edgeBuffer, 300, self.virusWidth + 32):
                self.virusList.append(Enemy(i,j))
```

- The enemyspawner class is made to give some space around the viruses so that it's not closely compact together and give them equal space between each sprite and the edge of the screen so it doesn't do running off it.
    - Doctor

```
class Doctor(pygame.sprite.Sprite):
    def __init__(self, screenWidth, screenHeight):
        super().__init__()
        self.image = pygame.image.load("img/doctor.png")
        self.surf = pygame.Surface((32, 32))
        self.rect = self.surf.get_rect(midbottom = (screenWidth / 2.0, screenHeight))

    def move(self, screenWidth, screenHeight):
        pressed_keys = pygame.key.get_pressed()

        if self.rect.left > 0:
            if pressed_keys[K_LEFT]:
                self.rect.move_ip(-5, 0)

        if self.rect.right < screenWidth:
            if pressed_keys[K_RIGHT]:
                self.rect.move_ip(5, 0)

    def draw(self, window):
        window.blit(self.image, self.rect)
```

-The init function of the doctor class its quite like that of the enemy class as it contains the sprite and its size/ 'hitbox". It would also determine where the syringe would be shoot from.

- The move function of the doctor class allows the return of whatever key is pressed in this case its to go left and right and to make sure it doesn't go off the screen, the doctor is made to move if the left rectangle its in is larger than 0 and as for the right as long as its smaller than the screenWidth. And the draw function draws it to the window.

- Syringe Class

```python
class Syringe(pygame.sprite.Sprite):
    def __init__(self, doctor):
        super().__init__()
        self.image = pygame.image.load("img/syringe.png")
        self.surf = pygame.Surface((10, 10))
        self.rect = self.surf.get_rect(center = (doctor.rect.midtop))
        self.fired = False

    def move(self):
        self.rect.move_ip(0, -5)

    def draw(self, window):
        window.blit(self.image, self.rect)
```

-The init function for the syringe class is also similar as the rest and it is made to shoot from the center of the doctor.

-The move function makes it move up to shoot and then Draw function draws it onto the window.

- Background Class

```python
class Background():
    def __init__(self, DISPLAYSURFACE):
        self.backgroundImage = pygame.image.load("img/bgIMG.jpeg")
        self.rectBGImage = self.backgroundImage.get_rect()
        self.moveSpeed = 0.5
        self.x1 = 0
        self.y1 = 0
        self.x2 = -(self.rectBGImage.width + self.moveSpeed)
        self.y2 = 0

    #updates the speed of the background as it moves
    def update(self):
        self.x1 += self.moveSpeed
        self.x2 += self.moveSpeed

        if self.x1 >= self.rectBGImage.width:
            self.x1 = -(self.rectBGImage.width + self.moveSpeed)

        if self.x2 >= self.rectBGImage.width:
            self.x2 = -(self.rectBGImage.width + self.moveSpeed)

    #renders the image of the background
    def render(self, DISPLAYSURFACE):
        DISPLAYSURFACE.blit(self.backgroundImage,(self.x1 , self.y1))
        DISPLAYSURFACE.blit(self.backgroundImage,(self.x2, self.y2))
```

-The init of the background class is to initialize the background it starts the function and draws the background

-The update function will update the position and the checking whether it is at the right side of the screen

-The render function would draw the images again once it figures out the image is of the screen and resets it by x1,y1 and x2.y2

VirusDriver.py

- This file is made to import all the classes that were made and use them to create the game.
- Imports

```
import pygame, random, time
from pygame.locals import *
from pygame import mixer
from Virus_Classes import Enemy
from Virus_Classes import EnemySpawner
from Virus_Classes import Doctor
from Virus_Classes import Syringe
from Virus_Classes import Background
```

I've use 1 overall function to start the game a main() function

```
def main():
    pygame.init()
    FPS = 60
    clock = pygame.time.Clock()
    speed = 1
    score = 0
    wait = 0
    imageReset = 0

    #colors
    red = (255, 0 ,0)
    black = (0, 0, 0)
    green = (0, 255, 0)
    white = (255, 255, 255)

    #font
    font = pygame.font.SysFont("Verdana", 60)
    smallfont = pygame.font.SysFont("Verdana", 20)
    gameOverFont = smallfont.render("Game over", True , black)
```

-First we run pygame.init() to initialize all imported modules. Then we add the game mechanics like FPS, score, speed, etc. We also add the colors and fonts of our choosing

```
#font
font = pygame.font.SysFont("Verdana", 60)
smallfont = pygame.font.SysFont("Verdana", 20)
gameOverFont = smallfont.render("Game over", True , black)

#window
screenWidth = 500
screenHeight = 600

backgroundImage = pygame.image.load("img/bgImage.png")
DISPLAYSURFACE = pygame.display.set_mode((screenWidth, screenHeight))
pygame.display.set_caption("Attack on Virus!!")

INCREASE_SPEED = pygame.USEREVENT + 1
pygame.time.set_timer(INCREASE_SPEED, 3000)

#music
mixer.init()
mixer.music.load("audio.mp3")
mixer.music.set_volume = 0.8
mixer.music.play
```

-Next choose the window height and width before choosing the background image and setting events such as the DSPLAYSURFACE and INCREASE SPEED.

-Mixer is used to add the music/audio

```
#start game
background = Background(DISPLAYSURFACE)
doctor = Doctor(screenWidth, screenHeight)
syringeList = []
EnemySpawner()
enemyGroup = pygame.sprite.Group()

for i in EnemySpawner.virusList:
    enemyGroup.add(i)

while True:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()

        if event.type == INCREASE_SPEED:
            speed += 0.1

    background.update()
    background.render(DISPLAYSURFACE)
```

-From background to enemygroup we are calling the initialize function while the for loop starts appending enemies into the list.

-The while true function here searches the events in the games for the increase speed when it is called it increments it by 0.1 while the quit event just stops running the code.

```
#to check on the doctor
    doctor.draw(DISPLAYSURFACE)
    doctor.move(screenWidth, screenHeight)

#score
    scoreText = smallfont.render("Score: " + str(score), True, white)
    DISPLAYSURFACE.blit(scoreText, (0,0))
```

-Doctor variable is used here to check the size of the player

-The score here is to display the amount of score you got.

```
enemy
    if len(enemyGroup) == 0:
        EnemySpawner.virusList = []
        EnemySpawner()
        enemyGroup = pygame.sprite.Group()
        for i in EnemySpawner.virusList:
            enemyGroup.add(i)
```

-Checks the length of the enemygroup and if it is 0 the spawner function would be called and start spawning enemies.

```
pressed_keys = pygame.key.get_pressed()
if pressed_keys [K_SPACE] and wait > 25:
    syringeList.append(Syringe(doctor))
    wait = 0

for i in syringeList:
    if i.rect.top < 2:
        syringeList.remove(i)
    i.move()
    i.draw(DISPLAYSURFACE)

for enemy in enemyGroup:

    enemy.move(False, 0, speed)
    enemy.draw(DISPLAYSURFACE)
```

-First the if function checks whether or not the space bar is being pressed or not and if so it would shoot a syringe every 25 frame interval before resetting back to 0.

The first for loop checks how many syringes have been fired and if the top of the rectangle is less than 2 it would remove the syringe

```
for enemy in enemyGroup:

    enemy.move(False, 0, speed)
    enemy.draw(DISPLAYSURFACE)

    if enemy.rect.left < 0:
        enemy.right()
    if enemy.rect.right > 500:
        enemy.left()
    if enemy.rect.bottom > 600:
        DISPLAYSURFACE.fill(green)
        DISPLAYSURFACE.blit(gameOverFont, (200,250))
        DISPLAYSURFACE.blit(scoreText, (215,300))
        pygame.display.update()
        time.sleep(4)
        EnemySpawner.virusList = []
        EnemySpawner()
        enemyGroup = pygame.sprite.Group()

        for i in EnemySpawner.virusList:
            enemyGroup.add(i)
        for resetEnemy in enemyGroup:
            resetEnemy.reset()
        score = 0
        speed = 1
```

-For each enemy inside the group it will move them at the same time and Draw () would draw it at the displaysurface.

-If the enemy hitbox on the left is less than 0 than they would go right and vise-versa but if the bottom of the hitbox hits the bottom, then it would be game over. The game over screen would be displayed and everything would be resetted.

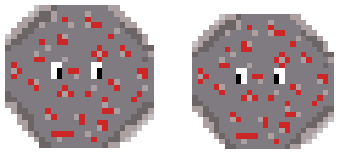
```
    for syringe in syringeList:
        if pygame.sprite.spritecollide(enemy, syringeList, False):
            enemy.kill()
            syringeList.remove(syringe)
            score = score + 10

pygame.display.update()
imageReset +=1
wait += 1
clock.tick(FPS)
```

-The last for loop would check if the enemy sprite and the syringe sprite collides it would “kill” the virus and remove the syringe while adding 10 points for every virus killed

-And last call on the loop with the main function.

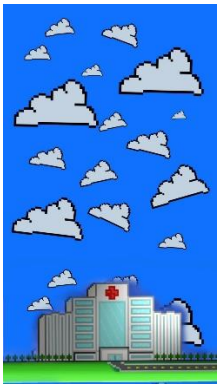
- **Music:**
  I tried to find a royalty free music to use for this game and came across one in this website https://www.bensound.com/royalty-free-music/electronica
- **Art:**
  As for the art that was used, they were most made by me using pixilart or combining pictures:



The viruses

The doctor

the syringe


The background

(Due to them being pixel art the picture size is too small and slightly blurry)