A

Mini Project Report On
OOPCGL Lab

CLASS**: SE II**

Batch**: E1**

GUIDED BY

Prof. Shweta Shah

DEPARTMENT OF COMPUTER TECHNOLOGY

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

DHANKAWADI, PUNE-43

SAVITRIBAI PHULE PUNE UNIVERSITY

AY: 2022-23

Submitted By:

Student Name - **Vedant Aher** Roll No – **21103**

Student Name - **Poonam Chapke** Roll No – **2113**

1) Title: ‘Space Invaders Game’

2) Problem Definitions

Design and implement game.

3) Hardware Requirements:

RAM: 4GB

4) Software Requirements:

Pycharm version 3.7

5) Theory

Python although not fully able to create AAA games that are suitable or being accepted by gamers m nowadays but is capable of being used to build functioning and fun, entertaining game

Python coding language is used in this game project. Various modules are used for various functions as per the requirement.

Python is the most versatile language, and it makes its presence almost in every field including Web-development, Machine Learning, Artificial Intelligence, GUI Application as well as Game Development.

Python provides a built-in library called pygame, which is used to develop the game. Once we understand the basic concepts of Python programming, we can use the pygame library to make games with attractive graphics, suitable animation, and sound.

Pygame is a cross-platform library that is used to design video games. It includes computer graphics and sound libraries to give the standard game experience to the user.

Graphical User Interface (GUI) and Command Line Interface (CLI) are the two means of interaction between a user and an electronic device. A GUI is a graphical representation in which the users can interact with software or devices through clickable icons. A CLI is a console or text-based representation in which the user types commands into a terminal to operate and navigate the software or devices.

Examining CLI and GUI requires an examination of their features, advantages.

GUI represent potential user actions. Although easier to use, a GUI is considered slower due to more visual output, which requires more memory than a text-based CLI.

A GUI offers control over a computer's files and operating system, but advanced tasks may require the command line. GUIs are excellent for general computer use and for users that want to use a machine without prior knowledge of its interface. More users are comfortable with a GUI because it is practical and usable.

Command Line Interface

A CLI uses a keyboard for text input of commands that are required to navigate the machine. As previously indicated, CLI requires commands to be used. Users need to know these commands to use a CLI. This is generally why CLI is less used than a GUI. However, CLI requires a smaller amount of
RAM and CPU processing time.

Choosing to use a GUI or a CLI is dependent on the purpose of computer navigation. Many users prefer a GUI because it is easier and more comfortable. Many users like a visual representation of their actions on a computer or other device. GUIs are the default when computer interaction is considered and is the
go-to because of their usability.

Initially, the Pygame module is imported which leads automatically it to add varied functions such as image addition for background work then the player work and transform of respective need

Step 1: Check for Python Installation
To install Pygame, Python must be installed already in your system. To check
whether Python is installed or not in your system, open the command prompt and give the command as shown below.
Step 2: Check for PIP installation
PIP is a tool that is used to install python packages. PIP is automatically installed with Python 2.7. 9+ and Python 3.4+. Open the command prompt and enter the command shown below to check whether pip is installed or not.
Step 3: Install Pygame
Step 2: Check for PIP installation
PIP is a tool that is used to install python packages. PIP is automatically installed with Python 2.7. 9+ and Python 3.4+. Open the command prompt and enter the command shown below to check whether pip is installed or not.
Step 3: Install Pygame
To install Pygame, open the command prompt and give the command as shown below: pip install pygame
Step 4: Check Whether PyGame is Working or not
Now open a new terminal and import the Pygame library to see whether it is working fine or not in our system. The library is imported successfully means we got successful.

What are Event Objects?

Any time the user does one of several actions (they are listed later in this chapter) such as pressing a keyboard key or moving the mouse on the program's window, a pygame. event. The event object is created by the Pygame library to record this "event". (This is a type of object called Event that exists in the event module, which itself is in the pygame module.) We can find out which events have happened by calling the pygame. event.get() function, which returns a list of pygame. events.Event objects (which we will just call Event objects for short).

The list of Event objects will be for each event that has happened since the last time pregame. event.get() function was called. (Or, if pygame.event.get() has never been called, the events that have happened since the start of the program.)

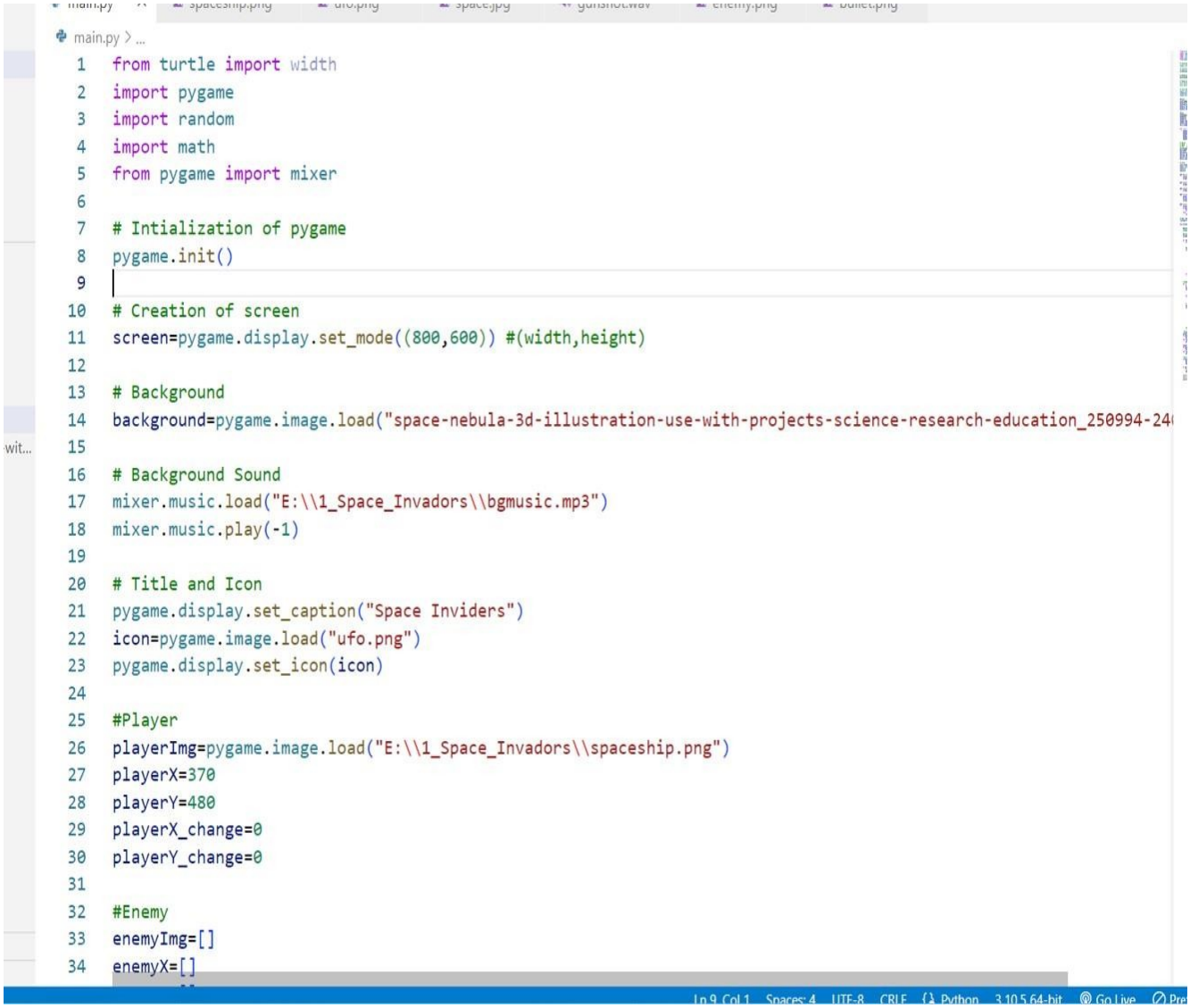
What is a quit event?

Event objects have a member variable (also called attributes or properties) named type which tells us what kind of event the object represents. Pygame has a constant variable for each of the possible types in the pregame. locals modules.

If the Event object is a quit event, then the pygame. quit() and sys. exit() functions are called.

The pygame. quit() function is sort of the opposite of the pregame.init() function: it runs code that deactivates the Pygame library.

Your programs should always call pygame.quit() before they call sys.exit() to terminate the program. Normally it doesn't matter since Python closes it when the program exits anyway. But there is a bug in IDLE that causes IDLE to hang if a Pygame program terminates before pygame. quit() is called.



main.py > ...

```python
from turtle import width
import pygame
import random
import math
from pygame import mixer

# Intialization of pygame
pygame.init()

# Creation of screen
screen=pygame.display.set_mode((800,600)) #(width,height)

# Background
background=pygame.image.load("space-nebula-3d-illustration-use-with-projects-science-research-education_250994-24

# Background Sound
mixer.music.load("E:\\1_Space_Invadors\\bgmusic.mp3")
mixer.music.play(-1)

# Title and Icon
pygame.display.set_caption("Space Inviders")
icon=pygame.image.load("ufo.png")
pygame.display.set_icon(icon)

#Player
playerImg=pygame.image.load("E:\\1_Space_Invadors\\spaceship.png")
playerX=370
playerY=480
playerX_change=0
playerY_change=0

#Enemy
enemyImg=[]
enemyX=[]
```

main.py > ...

```python
def show_score(x,y):
    score=font.render("Score : "+str(score_value),True,(255,255,255))
    screen.blit(score,(x,y))

def player(x,y):
    screen.blit(playerImg,(x,y))

def enemy(x,y,i):
    screen.blit(enemyImg[i],(x,y))

def fire_bullet(x,y):
    global bullet_state
    bullet_state="fire"
    screen.blit(bulletImg,(x+16,y+10))

def isCollision(enemyX,enemyY,bulletX,bulletY):
    distance=math.sqrt((math.pow(enemyX-bulletX,2))+(math.pow(enemyY-bulletY,2)))
    if distance < 27:
        return True
    else:
        return False

# Game loop
running=True

while(running):

    #RGB-Red,Green,Blue
    screen.fill((0,0,0))

    #Background
    screen.blit(background,(0,0))

    for event in pygame.event.get():
```

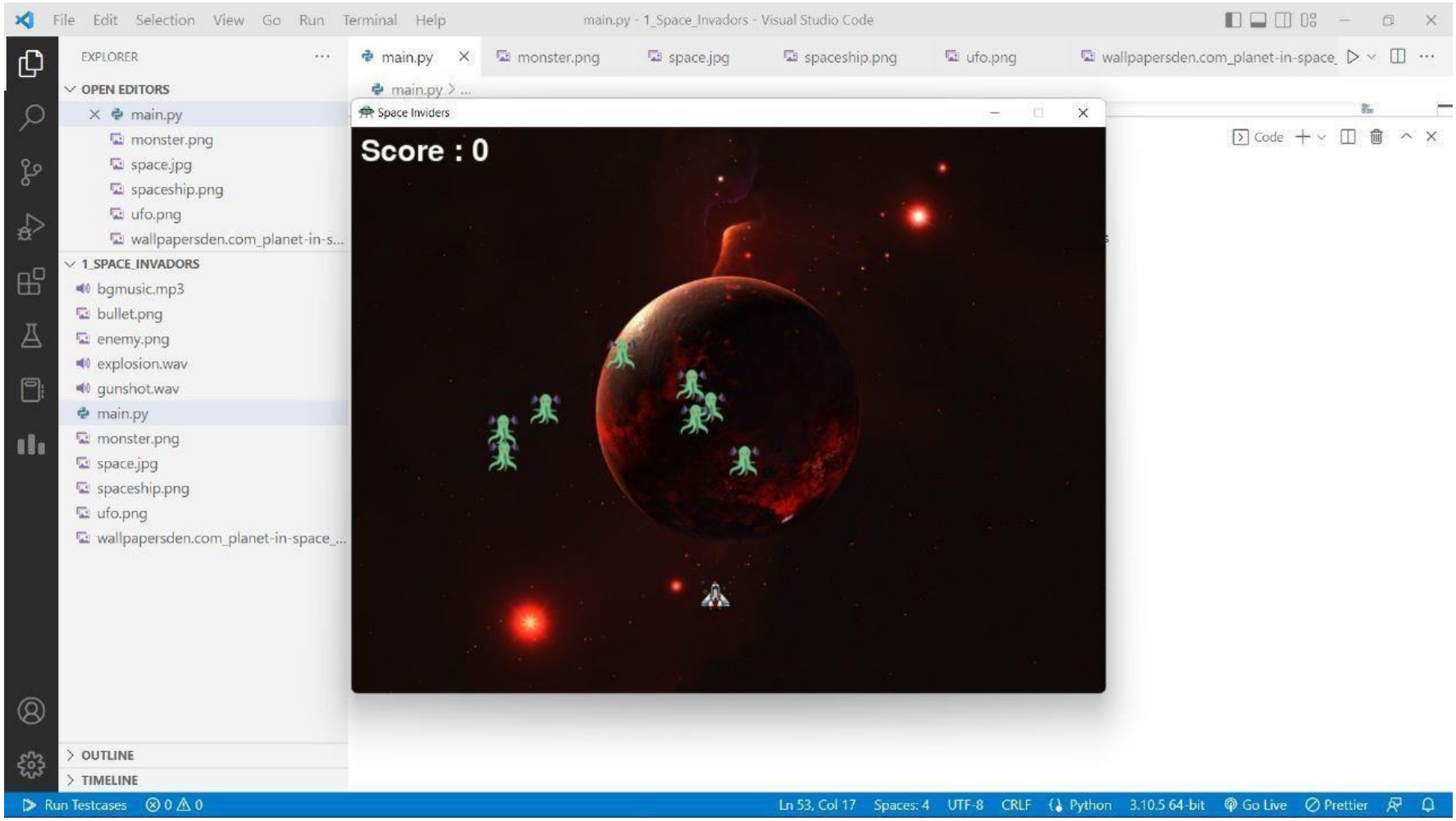
main.py > ...

```python
    playerY+=playerY_change
    if playerY<=0:
        playerY=0
    elif playerY>=536:
        playerY=536

    # Bullet Movement
    if bulletY<=0:
        bulletY=480
        bullet_state="ready"

    if bullet_state is "fire":
        fire_bullet(bulletX,bulletY)
        bulletY-=bulletY_change

    player(playerX,playerY)
    show_score(textX,textY)
    pygame.display.update()
```


File Edit Selection View Go Run Terminal Help
main.py - 1_Space_Invadors - Visual Studio Code
EXPLORER
OPEN EDITORS
main.py
monster.png
space.jpg
spaceship.png
ufo.png
wallpapersden.com_planet-in-s...
1_SPACE_INVADORS
bgmusic.mp3
bullet.png
enemy.png
explosion.wav
gunshot.wav
main.py
monster.png
space.jpg
spaceship.png
ufo.png
wallpapersden.com_planet-in-space_...
OUTLINE
TIMELINE
main.py
monster.png
space.jpg
spaceship.png
ufo.png
wallpapersden.com_planet-in-space_
main.py > ...
Code
Space Inviders
Score : 0
Run Testcases
0 0
Ln 53, Col 17
Spaces: 4
UTF-8
CRLF
Python
3.10.5 64-bit
Go Live
Prettier

Conclusion: We learnt basics of PyGame and successfully implemented OOPs concepts.

References:

- Geeks for Geeks.
- Reema Thareja, “Python Programming Using Problem Solving Approach” Oxford University Press