

Chapter

Balloon Pop Game

AIM:

The aim of the game is to pop as many balloons as possible within a limited time frame while managing a set number of chances. Players score points by clicking on balloons before they disappear off the screen. The game challenges players to demonstrate quick reflexes and accuracy in targeting balloons while keeping track of the remaining time and chances. Ultimately, the goal is to achieve the highest score possible before the game ends.

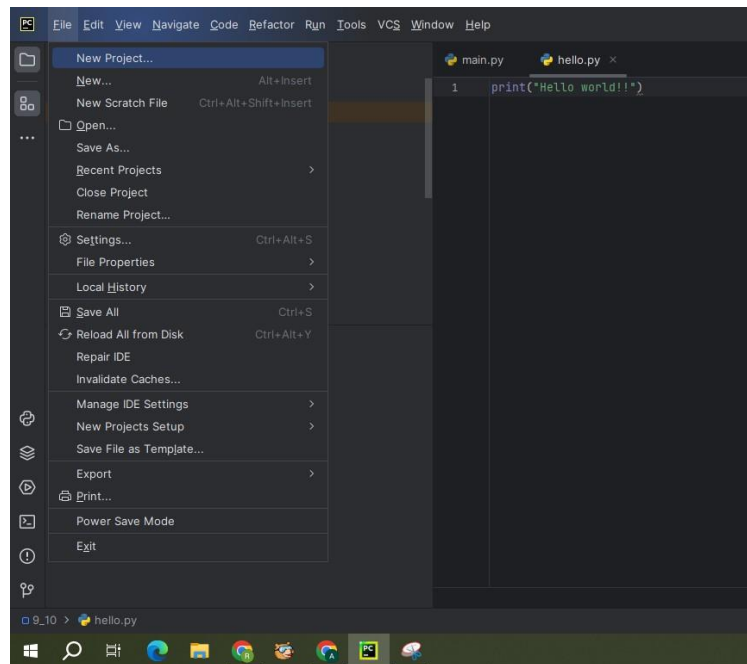
Introduction:

The provided code is for a balloon pop game built using the Pygame library in Python. It begins by initializing Pygame and setting up the screen dimensions. Balloons in the game are represented by a balloon class, which includes attributes like radius, color, position, and speed. The main game logic is encapsulated within the mainfunction, which contains a game loop allowing for the game to be restarted after completion.

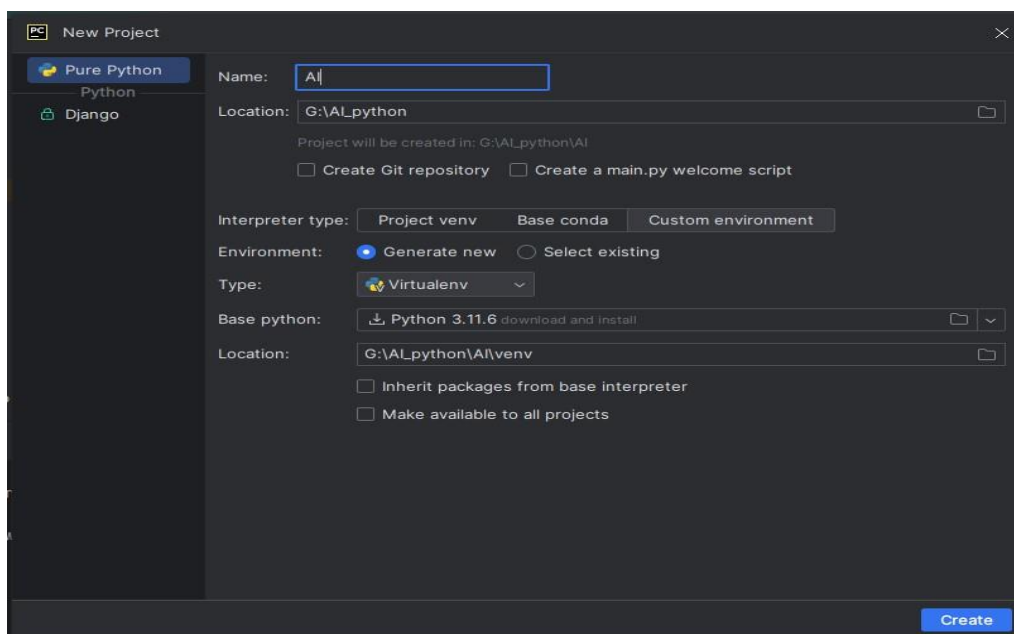
Events such as mouse clicks are handled within this loop, enabling players to pop balloons. New balloons are randomly generated with a certain probability during gameplay. The screen displays the current score, remaining time, and chances. When the game ends, either due to time running out or no remaining chances, a game over screen is shown with the final score, alongside a restart button for players to initiate a new game session. This restart mechanism waits for player input before resetting the game. Overall, the code follows a structured approach to game development, encompassing initialization, event handling, rendering, and game over logic to deliver an engaging balloon popping experience.

Procedure:

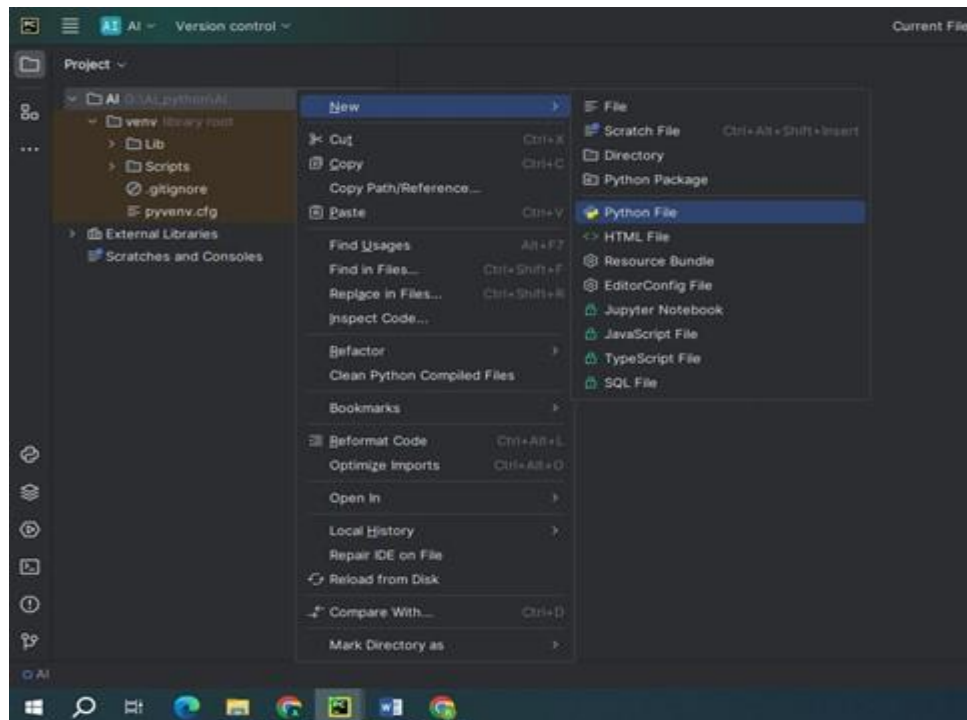
- Open PyCharm IDE software.
- Go to Menu —> File —> New Project.



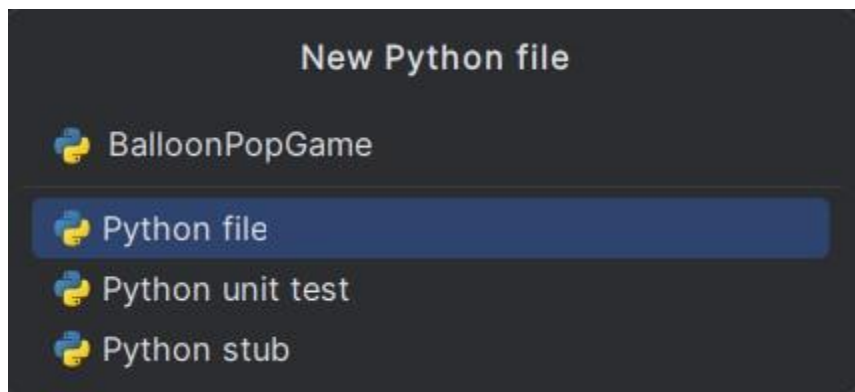
- A window will be appeared as below. Change the Name and Location of the project as per our requirement. Select custom environment for interpreter type. Select Generate new for Environment. Select Virtualenv for Type. Select latest version of python for Base python i.e., Python 3.11.6. And click on create



- Once our project is created, right click on project. Go to —> New —> Python File.



- Give the name to the python file: BalloonPopGame.



- Once the file is created, copy below given code to balloon pop game.

Balloon Pop Game code:

```
import pygame
import sys
import random

# Initialize Pygame
pygame.init()

# Set up screen dimensions
SCREEN_WIDTH = 800
SCREEN_HEIGHT = 600
screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
pygame.display.set_caption("Balloon Pop Game")
```

```

# Colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)

# Balloon parameters
BALLOON_RADIUS = 30
BALLOON_SPEED = 1
BALLOON_COLORS = [(255, 0, 0), (0, 255, 0), (0, 0, 255)] # Red, Green, Blue

# Game duration (in seconds)
GAME_DURATION = 60
CHANCES = 3

class Balloon:
    def __init__(self):
        self.radius = BALLOON_RADIUS
        self.color = random.choice(BALLOON_COLORS)
        self.x = random.randint(self.radius, SCREEN_WIDTH - self.radius)
        self.y = SCREEN_HEIGHT + self.radius
        self.speed = BALLOON_SPEED

    def move(self):
        self.y -= self.speed

    def draw(self):
        pygame.draw.circle(screen, self.color, (self.x, self.y),
self.radius)

def main():
    restart_game = True

    while restart_game:
        balloons = []
        score = 0
        clock = pygame.time.Clock()
        start_time = pygame.time.get_ticks()
        chances = CHANCES
        game_over = False

        while not game_over:
            screen.fill(WHITE)

            # Check game duration
            current_time = pygame.time.get_ticks()
            elapsed_time = (current_time - start_time) // 1000
            remaining_time = max(0, GAME_DURATION - elapsed_time)

            # Check if chances are over
            if chances == 0:
                game_over = True

            # Handle events
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    pygame.quit()
                    sys.exit()
                elif event.type == pygame.MOUSEBUTTONDOWN and not
game_over:
                    mouse_x, mouse_y = pygame.mouse.get_pos()

```

```

        missed = True
        for balloon in balloons:
            if balloon.x - balloon.radius <= mouse_x <=
balloon.x + balloon.radius and \
                                balloon.y - balloon.radius <= mouse_y <=
balloon.y + balloon.radius:
                balloons.remove(balloon)
                score += 1
                missed = False
                break
        if missed:
            chances -= 1

    # Create new balloons
    if random.random() < 0.02 and not game_over:
        balloons.append(Balloon())

    # Move and draw balloons
    for balloon in balloons:
        balloon.move()
        balloon.draw()

    # Display score
    font = pygame.font.SysFont(None, 36)
    score_text = font.render("Score: " + str(score), True, BLACK)
    screen.blit(score_text, (10, 10))

    # Display timer
    timer_text = font.render("Time: " + str(remaining_time), True,
BLACK)
    screen.blit(timer_text, (SCREEN_WIDTH - 120, 10))

    # Display chances
    chance_text = font.render("Chances: " + str(chances), True,
BLACK)
    screen.blit(chance_text, (SCREEN_WIDTH // 2 - 60, 10))

    pygame.display.flip()
    clock.tick(60)

    # Game over
    screen.fill(WHITE)
    font = pygame.font.SysFont(None, 36)
    game_over_text = font.render("Game Over! Score: " + str(score),
True, BLACK)
    screen.blit(game_over_text, (SCREEN_WIDTH // 2 - 150, SCREEN_HEIGHT
// 2 - 50))
    pygame.display.flip()

    # Add restart button
    restart_button = pygame.Rect(300, 400, 200, 50)
    pygame.draw.rect(screen, BLACK, restart_button)
    restart_text = font.render("Restart", True, WHITE)
    screen.blit(restart_text, (350, 410))
    pygame.display.flip()

    # Wait for restart button click
    restart_clicked = False
    while not restart_clicked:
        for event in pygame.event.get():
            if event.type == pygame.MOUSEBUTTONDOWN:

```

```

mouse_x, mouse_y = pygame.mouse.get_pos()
if restart_button.collidepoint(mouse_x, mouse_y):
    restart_clicked = True

pygame.quit()

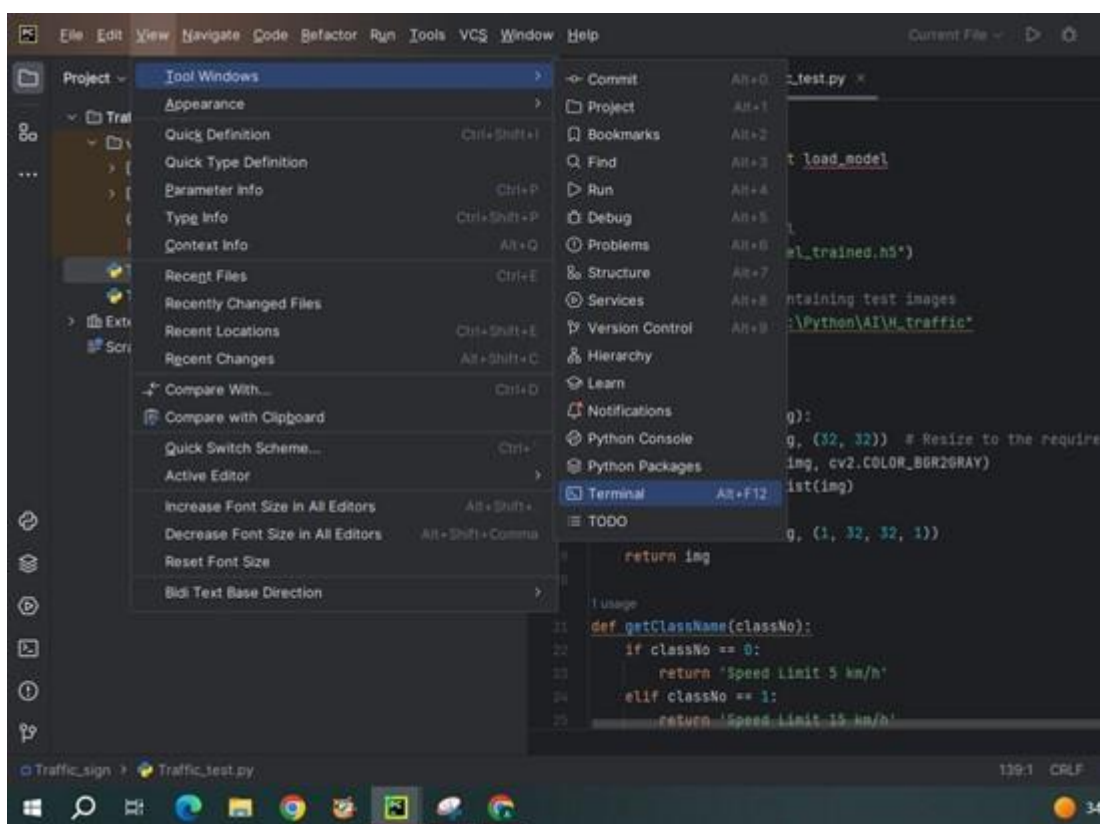
if __name__ == "__main__":
    main()

```

Libraries to install:

Ensure you have the following libraries installed before running the code:

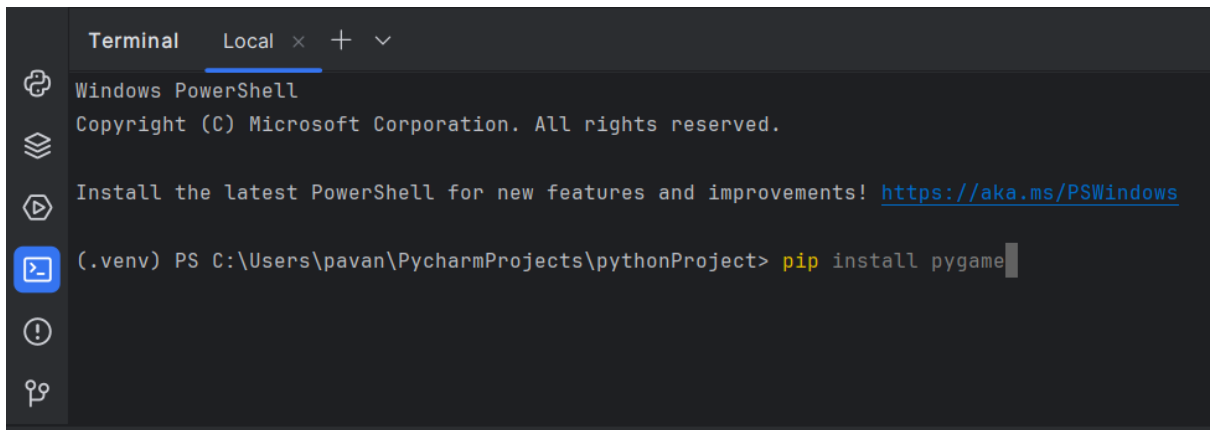
Go to Menu —> View —> Tool Windows —> Terminal



- **Pygame:**

Pygame is a set of Python modules designed for writing video games. It is built on top of the Simple DirectMedia Layer (SDL), a cross-platform multimedia library. Pygame provides functionalities for handling various aspects of game development, including graphics, sound, input devices like keyboard and mouse, and more. Install Pygame library, type the below command in terminal:

pip install pygame

A screenshot of a Windows PowerShell terminal window. The title bar shows 'Terminal' and 'Local' with a close button. The terminal content includes the Windows PowerShell logo, copyright information for Microsoft Corporation, a message about installing the latest PowerShell, and a command prompt showing the command 'pip install pygame' being entered in a virtual environment at the path 'C:\Users\pavan\PycharmProjects\pythonProject'.

- **Random:**

Actually, random is not a library that you install with pip; it's a built-in module in Python. The random module provides functions for generating random numbers and making random selections. Install to type the below command in terminal:

`pip install random`

- **SYS:**

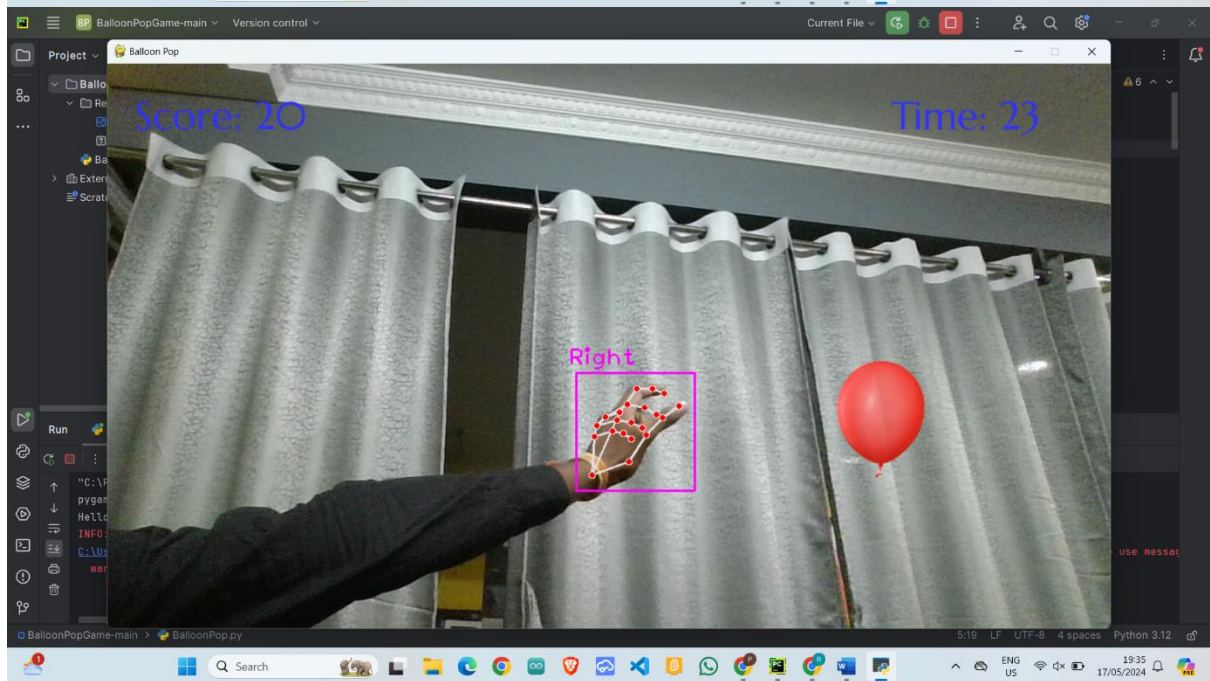
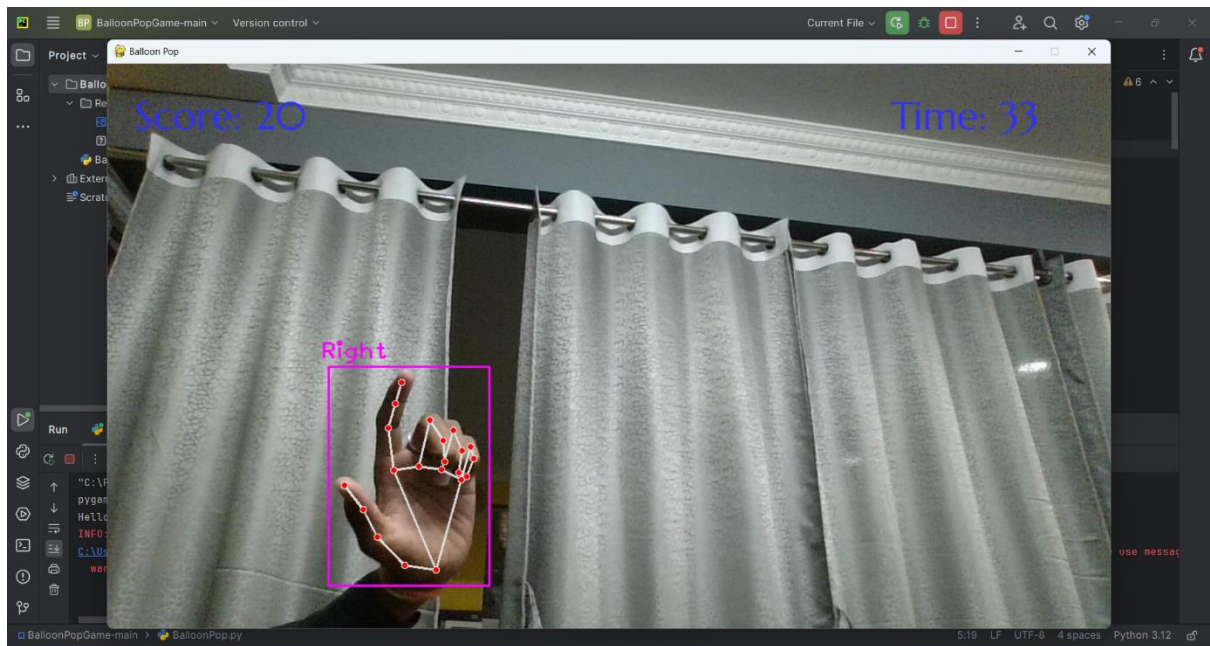
SYS is not a library that you install with PIP. It's actually a built-in module in Python, so it's available without the need for installation. The SYS module provides access to some variables used or maintained by the Python interpreter and to functions that interact strongly with the interpreter. Install to type the below command in terminal:

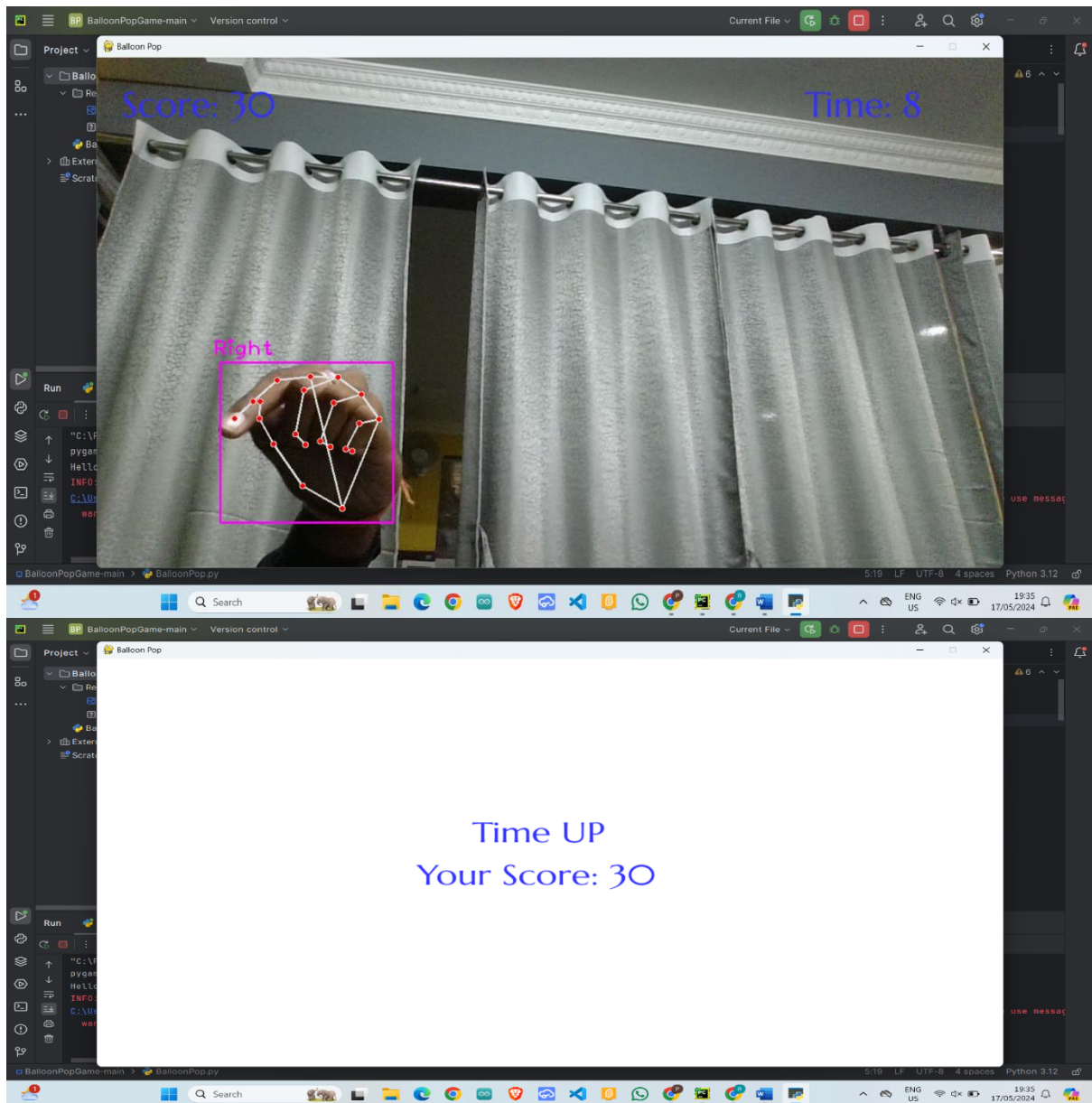
`pip install sys`

Running:

- Run the BalloonPopGame code

Output:





Conclusion:

At the end of a balloon pop game, you typically receive a score based on how many balloons you managed to pop within the given time limit or with the provided resources (like darts or clicks). Some games might also offer rewards or bonuses based on your performance. You can then compare your score with friends or try to beat your own record in subsequent rounds. Overall, it's a simple and entertaining game that can provide a quick burst of fun.