**Annexure-6**

**Copyright Submission Form** (Computer Software)

**Title of the Software/Application: *YOU ARE MY SUNSHINE***

**Name of authors:** R…. Joshi, Dr Bhaveshkumar Choithram Dharmani

**Name of Student/Scholar:** YASWANTH.BATTEPATI

**Registration Number: 12415739**

**Program Name:** B. Tech. in ECE

**Name/ UID of Supervisor:** Dr Bhaveshkumar Choithram Dharmani (23409)

**School: School of Electronoics and Electrical Engineering**

**Name and UID of the all co-authors/Co-Supervisor:** Dr ……… (12345)

**Affiliation details (with Pin code) of co-authors/Co- Supervisor (if External): ……………..**

**Language of the Work:** Python

**Summary, Uniqueness and Utility of work:** (150-250 words)

**This Pygame-based code creates an engaging game called "Collect the Coins." The code initializes Pygame and the Pygame mixer for handling game graphics and audio, respectively. Background music is loaded and set to loop indefinitely, adding an auditory dimension to the game.**

**The game window is set to 800x600 pixels, with the title "Collect the Coins." The player, controlled by arrow keys, is represented by a custom image that moves within the game window. Coins appear at random positions, and the player earns points by collecting these coins. Obstacles move randomly around the screen, and the player must avoid colliding with them.**

**The game loop continuously handles events, updates player and obstacle positions, checks for collisions, and redraws all elements on the screen. The game ends and displays a "Game Over" screen when the player collides with an obstacle.**

**Additionally, the game provides an instructional screen to guide players on how to start the game and play. The player's score is displayed on the screen and updates as coins are collected. This code is unique for its use of custom assets for the player, obstacles, and background, enhancing visual appeal. The background music adds to the auditory experience, making the game more immersive.**

**Dynamic obstacles provide a unique challenge with their random directions and speeds, ensuring each playthrough is different. Detailed instructions improve the user experience by clearly explaining the game's objectives and controls. The dedicated game loop, consistent frame rate control, and event handling contribute to a smooth and responsive gameplay experience.**

**This combination of simplicity and engaging features makes the game an excellent foundation for further game development projects.**

**Flow Diagram:**

**+--------------------------+**

Initialize Pygame

- init()

- mixer.init()

**| Initialize Pygame |**

**| - init() |**

**| - mixer.init() |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

Load Background Music

- load(music\_path)

- play(-1)

**| Load Background Music |**

**| - load(music\_path) |**

**| - play(-1) |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

| Screen Settings |

| - set\_mode() |

| - set\_caption()

**| Screen Settings |**

**| - set\_mode() |**

**| - set\_caption() |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

| Load Images and | | Initialize Variables

**| Load Images and |**

**| Initialize Variables |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

| Show Instructions |

| - fill(WHITE) |

| - render(text) |

| - blit(text) |

| - display.flip() |

| - event loop (wait for |

| KEYDOWN or QUIT) |

**| Show Instructions |**

**| - fill(WHITE) |**

**| - render(text) |**

**| - blit(text) |**

**| - display.flip() |**

**| - event loop (wait for |**

**| KEYDOWN or QUIT) |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

**| Game Loop |**

**| |**

**| +---------------------+ |**

| | Event Handling | | | | - KEYDOWN, QUIT

**| | Event Handling | |**

**| | - KEYDOWN, QUIT | |**

**| +---------------------+ |**

**| | |**

**| v |**

**| +---------------------+ |**

| Update Player | | | | Position | | |

**| | Update Player | |**

**| | Position | |**

**| +---------------------+ |**

**| | |**

**| v |**

**| +---------------------+ |**

| Check Collisions

**| | Check Collisions | |**

**| +---------------------+ |**

**| | |**

**| v |**

**| +---------------------+ |**

**| | Update Obstacles | |**

| Update Obstacles |

**| +---------------------+ |**

**| | |**

**| v |**

**| +---------------------+ |**

Render Everything | | | | - Background | |

| | - Player | | | | - Obstacles | |

| | - Coin | |

| | - Score | | |

**| | Render Everything | |**

**| | - Background | |**

**| | - Player | |**

**| | - Obstacles | |**

**| | - Coin | |**

**| | - Score | |**

**| +---------------------+ |**

**| | |**

**| v |**

**| +---------------------+ |**

Frame Rate Control|

**| | Frame Rate Control| |**

**| +---------------------+ |**

**| | |**

**| v |**

**| Game Over? |**

|| Game Over? |

| - if collision: |

|display\_game\_over| | - else: continue

**| - if collision: |**

**| display\_game\_over|**

**| - else: continue |**

**+--------------------------+**

**|**

**v**

**+--------------------------+**

End Program |

**| End Program |**

**+--------------------------+**

**Source Code/Object Code:**

import pygame  
import random  
import sys  
  
# Initialize Pygame  
pygame.init()  
  
# Initialize the mixer for audio  
pygame.mixer.init()  
  
# Load the background music  
background\_music\_path = r"C:\Users\yaswanth\Downloads\WhatsApp Audio 2024-11-28 at 10.22.53 AM.mpeg"  
try:  
 pygame.mixer.music.load(background\_music\_path)  
 pygame.mixer.music.play(-1) # Loop the music indefinitely  
except pygame.error as e:  
 print(f"Error loading background music: {e}")  
  
# Screen settings  
WIDTH, HEIGHT = 800, 600  
screen = pygame.display.set\_mode((WIDTH, HEIGHT))  
pygame.display.set\_caption("Collect the Coins")  
  
# Colors  
WHITE = (255, 255, 255)  
YELLOW = (255, 255, 0)  
BLACK = (0, 0, 0)  
RED = (255, 0, 0)  
  
# Player settings  
player\_size = 50  
player\_image\_path = r"C:\Users\yaswanth\Downloads\WhatsApp Image 2024-11-28 at 4.45.34 PM (1).jpeg"  
player\_image = pygame.image.load(player\_image\_path)  
player\_image = pygame.transform.scale(player\_image, (player\_size, player\_size))  
player = pygame.Rect(WIDTH // 2, HEIGHT // 2, player\_size, player\_size)  
player\_speed = 10  
  
# Coin settings  
coin\_size = 30  
coin = pygame.Rect(  
 random.randint(0, WIDTH - coin\_size), random.randint(0, HEIGHT - coin\_size), coin\_size, coin\_size  
)  
  
# Moving obstacle settings  
obstacle\_size = 60  
obstacle\_image\_path = r"C:\Users\yaswanth\Downloads\WhatsApp Image 2024-11-28 at 4.45.34 PM.jpeg"  
obstacle\_image = pygame.image.load(obstacle\_image\_path)  
obstacle\_image = pygame.transform.scale(obstacle\_image, (obstacle\_size, obstacle\_size))  
obstacles = []  
obstacle\_speeds = []  
max\_obstacles = 5 # Set the fixed number of obstacles  
  
# Initialize obstacles  
for \_ in range(max\_obstacles):  
 obstacles.append(  
 pygame.Rect(  
 random.randint(0, WIDTH - obstacle\_size),  
 random.randint(0, HEIGHT - obstacle\_size),  
 obstacle\_size,  
 obstacle\_size,  
 )  
 )  
 obstacle\_speeds.append((random.choice([-3, 3]), random.choice([-3, 3])))  
  
# Load background image  
background\_image\_path = r"C:\Users\yaswanth\Downloads\WhatsApp Image 2024-11-28 at 4.46.58 PM.jpeg"  
background\_image = pygame.image.load(background\_image\_path)  
background\_image = pygame.transform.scale(background\_image, (WIDTH, HEIGHT))  
  
# Score  
score = 0  
font = pygame.font.Font(None, 36)  
end\_font = pygame.font.Font(None, 48)  
  
# Clock for frame rate control  
clock = pygame.time.Clock()  
  
  
def show\_instructions():  
 *"""Display the instructions screen."""* while True:  
 screen.fill(WHITE)  
  
 # Display instructions  
 title\_text = end\_font.render("Welcome to Collect the Coins!", True, (0, 0, 255))  
 instructions\_text = font.render("Avoid obstacles and collect the yellow coin.", True, BLACK)  
 controls\_text = font.render("Use arrow keys to move.", True, BLACK)  
 start\_text = font.render("Press ENTER or SPACE to start the game.", True, RED)  
  
 # Center the text  
 screen.blit(title\_text, (WIDTH // 2 - title\_text.get\_width() // 2, HEIGHT // 2 - 100))  
 screen.blit(instructions\_text, (WIDTH // 2 - instructions\_text.get\_width() // 2, HEIGHT // 2 - 50))  
 screen.blit(controls\_text, (WIDTH // 2 - controls\_text.get\_width() // 2, HEIGHT // 2))  
 screen.blit(start\_text, (WIDTH // 2 - start\_text.get\_width() // 2, HEIGHT // 2 + 50))  
  
 pygame.display.flip()  
  
 # Wait for user input to start the game  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 sys.exit()  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_RETURN or event.key == pygame.K\_SPACE: # Start game  
 return  
  
  
def display\_game\_over(final\_score):  
 *"""Display the 'Game Over' screen with final score."""* screen.fill(WHITE)  
 game\_over\_text = end\_font.render("Better Luck Next Time!", True, RED)  
 score\_text = font.render(f"Your Score: {final\_score}", True, BLACK)  
 screen.blit(game\_over\_text, (WIDTH // 2 - game\_over\_text.get\_width() // 2, HEIGHT // 2 - 50))  
 screen.blit(score\_text, (WIDTH // 2 - score\_text.get\_width() // 2, HEIGHT // 2 + 10))  
 pygame.display.flip()  
 pygame.time.wait(3000) # Wait for 3 seconds  
 pygame.quit()  
 sys.exit()  
  
  
# Show instructions before starting the game  
show\_instructions()  
  
# Game loop  
running = True  
while running:  
 # Handle events  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 sys.exit()  
  
 # Move player with arrow keys  
 keys = pygame.key.get\_pressed()  
 if keys[pygame.K\_UP] and player.top > 0:  
 player.y -= player\_speed  
 if keys[pygame.K\_DOWN] and player.bottom < HEIGHT:  
 player.y += player\_speed  
 if keys[pygame.K\_LEFT] and player.left > 0:  
 player.x -= player\_speed  
 if keys[pygame.K\_RIGHT] and player.right < WIDTH:  
 player.x += player\_speed  
  
 # Check for collision with the coin  
 if player.colliderect(coin):  
 score += 1  
 # Reposition the coin  
 coin.x = random.randint(0, WIDTH - coin\_size)  
 coin.y = random.randint(0, HEIGHT - coin\_size)  
  
 # Move obstacles  
 for i, obstacle in enumerate(obstacles):  
 obstacle.x += obstacle\_speeds[i][0]  
 obstacle.y += obstacle\_speeds[i][1]  
  
 # Reverse direction if an obstacle hits the screen edges  
 if obstacle.left <= 0 or obstacle.right >= WIDTH:  
 obstacle\_speeds[i] = (-obstacle\_speeds[i][0], obstacle\_speeds[i][1])  
 if obstacle.top <= 0 or obstacle.bottom >= HEIGHT:  
 obstacle\_speeds[i] = (obstacle\_speeds[i][0], -obstacle\_speeds[i][1])  
  
 # Check for collision with the player  
 if player.colliderect(obstacle):  
 display\_game\_over(score) # Call game over function  
  
 # Draw everything  
 screen.blit(background\_image, (0, 0)) # Draw background image  
 screen.blit(player\_image, player.topleft) # Draw player as image  
 pygame.draw.ellipse(screen, YELLOW, coin) # Coin  
  
 # Draw obstacles  
 for obstacle in obstacles:  
 screen.blit(obstacle\_image, obstacle.topleft)  
  
 # Display the score  
 score\_text = font.render(f"Score: {score}", True, BLACK)  
 screen.blit(score\_text, (10, 10))  
  
 # Update the display  
 pygame.display.flip()  
  
 # Control the frame rate  
 clock.tick(30)