Lesson 6: Setting up Your Game

Objectives:

- Create a space-themed game where the player navigates a spaceship to collect items and avoid obstacles.

Project Breakdown

Game Concept and Setup

Objective:
- Navigate the spaceship, collect stars, and avoid asteroids.

Setup:
- Initialize the Pygame window and create a game loop.

Creating Game Objects

Spaceship: Class to represent the player's spaceship.

Stars and Asteroids: Classes to represent collectible stars and obstacles.

Player Movement

Use keyboard inputs to move the spaceship.

Collision Detection

Detect collisions between the spaceship and other objects (stars and asteroids).

Scoring System

Increase score for collecting stars and decrease for hitting asteroids.

Game Loop and Updates

Continuously update the game state, draw objects, and check for collisions.

Part 1: Setting Up Your Game

Objective: Initialize Pygame and create a game window.

1. **Initialize Pygame**: Write a Python program to initialize Pygame and create a window.
2. **Create a Game Loop**: Implement a game loop that keeps the window open until the user closes it.
3. **Fill the Background**: Fill the window with a background color.

Example Code:

```
import pygame
from pygame.locals import *
import math
import os
import random

# Set SDL audio driver to avoid driver error in console
os.environ['SDL_AUDIODRIVER'] = 'dsp'

# Initialize Pygame
pygame.init()


# Set up the display
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("Space Adventure")

# Main game loop
running = True
while running:
    for event in pygame.event.get():
        if event.type == QUIT:
            running = False

    screen.fill((0, 0, 0))  # Fill the screen with black
    pygame.display.flip()  # Update the display

pygame.quit()
```

Questions:

1. What function is used to initialize Pygame?

2. How do you set the window title in Pygame?

3. What is the purpose of the `game loop`?

Part 2: Creating Game Objects

Objective: Define classes for the spaceship, stars, and asteroids.

Instructions

1. **Define a Spaceship Class**: Create a class to represent the player's spaceship.
2. **Define Star and Asteroid Classes**: Create classes to represent collectible stars and obstacles.
3. **Draw Objects**: Implement methods to draw these objects on the screen.

Example Code:

```
class Spaceship:
  def __init__(self, position):
    self.position = position
    self.color = (0, 255, 0) # Green

  def draw(self, screen):
    pygame.draw.rect(screen, self.color, (*self.position, 50, 30))

class Star:
  def __init__(self, position):
    self.position = position
    self.color = (255, 255, 0) # Yellow

  def draw(self, screen):
    pygame.draw.circle(screen, self.color, self.position, 10)

class Asteroid:
  def __init__(self, position):
    self.position = position
    self.color = (255, 0, 0) # Red

  def draw(self, screen):
    pygame.draw.circle(screen, self.color, self.position, 20)
```

Questions:

1.) What is a class in Python?

2.) How do you create an instance of a class?

3.) What method do you use to draw the spaceship on the screen?

Part 3: Moving the Spaceship

Objective: Implement player controls to move the spaceship.

Instructions

1. **Player Movement**: Use keyboard inputs to move the spaceship.
2. **Update Position**: Update the spaceship's position based on user input.

Example Code:

```
keys = pygame.key.get_pressed()
if keys[K_LEFT]:
    spaceship.move(-5, 0)
if keys[K_RIGHT]:
    spaceship.move(5, 0)
if keys[K_UP]:
    spaceship.move(0, -5)
if keys[K_DOWN]:
    spaceship.move(0, 5)
```

Questions:

1. How do you detect key presses in Pygame?

2. How do you update the position of an object?

Part 4: Collision Detection

Objective: Detect collisions between the spaceship and other objects.

Instructions

1. **Collision Function**: Write a function to detect collisions.
2. **Update Game Loop**: Check for collisions in the game loop and respond appropriately.

Example Code:

```
import math

def detect_collision(obj1, obj2):
    dx = obj1.position[0] - obj2.position[0]
    dy = obj1.position[1] - obj2.position[1]
    distance = math.sqrt(dx**2 + dy**2)
    return distance < 20  # Adjust based on object sizes
```

Questions:

1. What is collision detection?

2. How do you calculate the distance between two points?

Part 5: Scoring System

Objective: Implement a scoring system for the game.

Instructions

1. **Initialize Score**: Create a global variable for the score.
2. **Update Score**: Write a function to update the score based on collisions.
3. **Display Score**: Show the score on the screen.

Example Code:

```
score = 0


def update_score(points):
    global score
    score += points
    print("Score:", score)

# Update the game loop to include collision detection and scoring
for star in stars[:]:
    if detect_collision(spaceship, star):
        stars.remove(star)
        update_score(10)
```

Questions:

1. How do you create and update a global variable in Python?

2. How do you display text on the Pygame window?

ANSWER KEY:

Part 1: Setting Up Your Game

Questions and Answers:

1. **What function is used to initialize Pygame?**
   - `pygame.init()`
2. **How do you set the window title in Pygame?**
   - `pygame.display.set_caption("Space Adventure")`
3. **What is the purpose of the game loop?**
   - The game loop keeps the window open and continuously updates the game state, processes user input, and renders graphics until the user closes the window.

Part 2: Creating Game Objects

Questions and Answers:

1. **What is a class in Python?**
   - A class in Python is a blueprint for creating objects. It defines a set of attributes and methods that the created objects will have.
2. **How do you create an instance of a class?**
   - You create an instance of a class by calling the class as if it were a function,

     e.g., `spaceship = Spaceship(position).`
3. **What method do you use to draw the spaceship on the screen?**
   - `spaceship.draw(screen)`

Part 3: Moving the Spaceship

Questions and Answers:

1. **How do you detect key presses in Pygame?**
   - Using `pygame.key.get_pressed()`
2. **How do you update the position of an object?**
   - By changing the object's position attribute based on the detected key presses and then redrawing the object in its new position.

Part 4: Collision Detection

Questions and Answers:

1. **What is collision detection?**
   - Collision detection is the process of determining when two or more objects in a game intersect or come into contact.
2. **How do you calculate the distance between two points?**
   - Using the distance formula: `distance = math.sqrt((x2 - x1)**2 + (y2 - y1)**2)`

Part 5: Scoring System

Questions and Answers:

1. **How do you create and update a global variable in Python?**
   - By declaring the variable outside of any function and using the `global` keyword inside functions that modify the variable, e.g.,

```
score = 0
def update_score(points):
    global score
    score += points
```

2. **How do you display text on the Pygame window?**
   - By rendering the text to a surface using `pygame.font.Font` and `render` methods, then blitting the text surface onto the main screen.