Name student_1: LESLIE TIENTCHEU NOUMOWE
Name student_2: RANDRIAMAMONJISOA HAJA

PROJECT NUMBER 4: Réseaux Euclidiens et Python

Introduction

The project involves the development of a Pygamebased application with a graphical user interface (GUI). The application is structured around multiple scenes, each serving a specific purpose within the context of a game or interactive experience. The primary features include a home screen with interactive buttons, an introductory scene requiring specific key presses, an encryption scene with a difficulty slider and an encryption button, and a scene involving user input for a game.

LINK TO A VIDEO THAT SHOWS HOW THE GAME IS SET-UP: https://www.youtube.com/watch?v=WJ_RDMj_eek

Code Components

1. Home Screen (Class: HomeScreen)

Purpose: The home screen serves as the initial interface presented to the user, providing options to start different game modes.
Components:

- Title: "Cryptris"
- Interactive Buttons: "Start game Press a" and "File Mode" (currently commented out)

Functionality:

- User initiates game mode selection by pressing the 'a'.
- key.Pygame event handling for button clicks.

2. Text Input and Button Classes

Purpose: These classes provide reusable components for handling user input and button interactions.
Components:

- TextInput class: Handles input for username, password, and messages.
- Button class: Represents clickable buttons with associated actions.

Functionality:

- Allows users to input username, password, and messages.
- Buttons trigger actions based on events (e.g., key presses).

3. Intro Scene (Class: IntroScene)

Purpose: Represents a scene where the user inputs data and triggers specific actions to proceed.
Components:

- TextInput fields: Username, Password, and Message.
- Buttons: "Press Shift ctrl" and "Please fill all the fields" (conditionally displayed).

Functionality:

- Captures user input for username, password, and message.
- Transitions to the next scene on specific key presses.
- Displays messages based on user actions.

4. Slider and Encryption Scene (Classes: Slider, Button, EncryptionScene)

Purpose: Implements a difficulty slider and an encryption scene where users can interact with buttons to initiate actions.
Components:

- Slider class: Allows users to select difficulty levels.
- Button class: Represents clickable buttons with associated actions.
- EncryptionScene class: Manages the overall encryption scene.

Functionality:

- Difficulty slider enables users to choose game difficulty.
- Buttons trigger encryptionrelated actions and display messages.
- Scene responds to key presses (pygame.K_m and pygame.K_r) and updates the display accordingly.

Key Concepts and Techniques

The project extensively utilizes the Pygame library to create a graphical interface, handle events, and manage scenes. ObjectOriented Programming: Classes and objects are employed to encapsulate and organize code, promoting modularity and reusability. The application responds to user input, including key presses and mouse events, to drive the interactiveelements. The game State Manager is used to control the flow between different scenes, ensuring a smoothtransition between game states.

PACKAGES USED

The selection of appropriate packages is crucial in the development of a cryptographic Tetris game. Pygame was chosen as the primary game development library due to its ease of use, cross-platform compatibility, and extensive resources. pygame_gui complements Pygame by simplifying the creation of graphical user interfaces within the game.

time: The time module provides various time-related functions. It's commonly used for measuring and controlling time in Python programs.

pygame: is a set of Python modules designed for writing video games. It includes computer graphics and sound libraries and is often used for game development.

sys: The sys module provides access to some variables used or maintained by the Python interpreter and functions that interact strongly with the interpreter.

pathlib: It provides an object-oriented interface for file system paths. It is a more modern and convenient way to handle file paths than using traditional string-based methods.

tkinter: It is the standard GUI (Graphical User Interface) package that comes with Python. It provides tools for creating desktop applications with graphical interfaces.

messagebox (from tkinter): The messagebox module is part of the Tkinter library and is used for creating message boxes, including alert and confirmation dialogs.

cryptography: it was used for cryptographic operations. It includes modules for handling asymmetric and symmetric encryption, hashing, key derivation, etc.

hashlib: it provides a common interface to many secure hash and message digest algorithms, including SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512.

Functionality Achieved

- User inputs a message that is encrypted using a public key.

- The encrypted message is displayed within the game.

- The player controls Tetrominoes in a Tetris-like game environment.

- The player's score is tracked, and full lines are cleared for points.

- The player must clear the entire board to decrypt the message

- The user selects a difficulty level (1 to 3), each level have a different set of tetriminoes

cryptography:

The library is utilized for implementing cryptographic functionalities within the game.

Key features include:

1. Public Key Encryption: cryptography supports public key encryption, enabling the encryption ofuser-inputted messages using a public key.

2. Padding and Hashing: The library provides padding and hashing algorithms necessary for secureencryption and decryption processes.

3. Backend Support: It offers a backend system that facilitates the use of specific cryptographic Implementations.

Tests Conducted

The test conducted for the cryptographic Tetris game primarily focused on ensuring the game's functionality, educational aspects, and entertainment value. Here's an explanation of the test.

process and its results

Functionality Test:

- Checked if the game runs smoothly without crashes or errors.

- Verified that the dynamic scoring system generates a required score that is always divisible by 100.

- Tested the countdown timer to ensure it starts at the expected duration (30 seconds) and triggers the "game over" state if time runs out.

- Verified that the cryptographic elements, including encryption and decryption, function correctly.

Educational Aspect Test:

- Evaluated the game's ability to introduce players to the concept of encryption through the encrypted message.

- Assessed how well the game educates players about achieving a specific score within a time limit to decrypt the message.

Entertainment Test:

- Evaluated the overall gaming experience to ensure it is engaging and fun.

- Assessed the level of challenge presented by the dynamic scoring and countdown timer.

- Checked if players found the game entertaining and if it encouraged them to replay for higher scores.
-

Encryption/Decryption Workflow

1. Key Generation:

- A key is generated using a username provided by the player.

- The key is used for both encrypting and decrypting the secret message.

2. Message Encryption:

- The secret message is encrypted using the public key generated from the password.

- The encrypted message is displayed to the player.

3. Message Decryption:

- The player can trigger the decryption process by clearing all the blocks in the board.

- The private key, kept secret, is used to decrypt the message.

4. Display Decrypted Message:

- Once the decryption is successful, the decrypted message is displayed to the player.

Gameplay

1. Game Initialization:

- The player initiates the game by pressing the "a" key.

- A username and a secret message are entered by the player.

2. Encryption Process:

- Upon entering the username and message, the player triggers the encryption process.

- The message is encrypted using a dynamically generated key.

3. Game Window Entry:

- To enter the game window, the player uses the key combination Ctrl + Shift.

4. Game Interface:

- The game window displays a grid with 10 columns and 20 rows.

- Encrypted message and a score counter are shown on the right side.

5. Tetrimino Introduction:

- To increase difficulty, 5 random Tetriminos drop onto the grid at the start.

- The player is challenged to clear these Tetriminos to win the game.

6. Gameplay Mechanics:

- Tetriminos fall from the top, and the player can move them left or right.

- Rotation is possible to fit Tetriminos into the grid.

- Clearing lines earns the player points.

7. Scoring System:

- Completing a line removes it from the grid and adds 100 points to the score.

- The player's objective is to achieve a high score by clearing lines.

8. Winning Condition:

- The player wins by clearing all Tetriminos from the grid.

- Winning triggers the decryption process.

9. Decryption Process:

- Upon winning, the encrypted message is decrypted using the generated key.

- The decrypted message is revealed to the player.

10. Game Over:

- The game ends if a Tetrimino reaches the top of the grid.

- Alternatively, the player can win by achieving the required score.

11. Dynamic Difficulty:

- The game dynamically adjusts difficulty by introducing 5 random Tetriminos.

- Increasing levels may bring additional challenges or speed up Tetrimino drops.

12. Score Tracking:

- The score counter keeps track of the player's progress.

- The required score for winning increases with each level.

13. Player Interaction:

- Player interaction involves precise Tetrimino placement to complete lines.

- Encryption and decryption elements engage the player in a unique gaming experience.

14. Unique Game Elements:

- Encryption adds a layer of strategy to the gameplay, linking success with message decryption.

- The game combines classic Tetris mechanics with cryptographic challenges.

15. Overall Objective:

- The player's goal is to clear the grid, achieve a high score, and decrypt the hidden message.