Import pygame library
Import math library
Import random library

Initialise pygame

Get monitor resolution
Store width and height resolution in WIDTH variable and HEIGHT variable respectively
Set “win” variable and display pygame application in fullscreen
Set the application caption to “Educational Hangman Game!”

Set radius and gap of buttons
Set start width and height positions of buttons
Create list of letters

Set ASCII value for character A
Set x and y position of each letter of alphabet from left to right at the bottom of the screen in 13 columns and 2 rows

Set colour variables for WHITE, BLACK, GRAY, GREEN, YELLOW and RED

Set different pygame fonts for LETTERS, WORDS, TITLE, SUBTITLES and BUTTONS

Create image list for all hangman assets
Load correct hangman images depending on value from 1 to 10
Scale all hangman images corresponding to the player’s screen size

Set game variables for HANGMAN_STATUS, WORDS LIST, WORD, GUESSED, DIFFICULTY and CURRENT_STATE

Set stat variables for WIN_AMOUNT and LOSE_AMOUNT

Define “draw_title” function
    Render title “H A N G M A N G A M E” in font for TITLE and in BLACK
    Set text x and y positions to be in the top middle of the player’s screen
    Draw a line underneath text

    Detect if player is in menu
        If true, draw subtitle underneath title line saying "By: Kaleb Vong 11SENG1" in font for SUBTITLE and in BLACK

    If false:
        Render chosen difficulty in either GREEN for EASY, YELLOW for MEDIUM, or RED for HARD

Define “choose_word” function:
    Import word variable from game variable list

    Detect if player has chosen EASY mode
        Open “easy words” list file
        Set “words” list variable to all words in opened list file

    Else detect if the player has chosen MEDIUM mode
        Open “medium words” list file
        Set “words” list variable to all words in opened list file

    Else detect if the player has chosen HARD mode
        Open “hard words” list file
        Set “words” list variable to all words in opened list file

    Set word variable to random word inside words list variable

Define “draw” function
    Import word variable from game variable list

    Fill screen to white
    Draw title by calling “draw_title” function

    Call stats function

    Set display_word variable as an empty string
        If the letter is inside the chosen word and also in the guessed list, store the letter inside display_word variable.
        Else store each letter as “_ “ inside display_word variable
    Render display_word variable on screen

    For each letter of the alphabet
        If unclicked and still visible
            Draw a circle around each letter’s x and y positions
            Draw the letter in its corresponding circle

    Display corresponding hangman image on screen with hangman_status variable

    Draw back button in top left corner of player’s screen
    Scale back button with player’s screen

    Update pygame display

Define “display_message” function
    Delay sequence by 1000 ms
    Fill screen with WHITE
    Render either “win” or “lose” message depending on if the player has won or lost the game
    Update pygame display
    Delay sequence by 3000 ms

Define “menu” function
    Fill screen with WHITE
    Call draw_title function
    Call stats function

    Draw three long gray rectangles on top of each other on the player’s screen

    Label the top-most rectangle with “P L A Y” in WHITE
    Label middle rectangle with “DIFFICULTY: “ in WHITE
    Label bottom rectangle with “Q U I T” in white

    Set default difficulty to EASY mode

    Update pygame display

Define “diff_name” function
    Clear text display previously selected difficulty by hiding it with a grey rectangle
    Render new difficulty selected in corresponding difficulty colour (GREEN = EASY, YELLOW = MEDIUM, RED = HARD)
    Update pygame display

Define “stats” function
    Import “win_amount” and “lose_amount” variables

    Render “W I N S: “ at the bottom of player’s screen in BLACK and in SUBTITLE font
    Render “ L O S S E S: “ at the bottom of player’s screen in BLACK and in SUBTITLE font

    Render value stored in win_amount variable next to “W I N S: “ in GREEN and in SUBTITLE font
    Render value stored in lose_amount variable next to “L O S S E S: “ in RED and in SUBTITLE font

Define “main” function
    Import “hangman_status”, “difficulty”, “word”, “guessed” and “current_state” variables

Import “win_amount” and “lose_amount” variables

Set game to run at 60 FPS
Set clock variable to track amount of time game is running
Set “run” variable to true
Set “shown_warning” variable to 0

Draw menu by calling menu function
Set current_state variable to “menu”

Draw detection boxes for all menu buttons to detect collisions with mouse inputs

Draw detection box for in-game back button

Set up main game loop while game is running (run = True)
    Set clock variable to the FPS the game is running at

    Get all pygame events coded by default into pygame library
        If “X” button on application is pressed, quit the game by setting “run = False”

        If any mouse button is pressed (such as left-click, right-click, middle-click)
        Store position of where mouse was clicked in m_x and m_y variables
            If “current_state” variable is “menu”
                Quit game if mouse click detected inside detection box of quit button

                If mouse click detected inside detection box of difficulty button
                    Increase “difficulty” variable by 1
                    If difficulty variable > 2 then reset back to 0

                    If difficulty variable matches 0
                    Call diff_name function to draw EASY mode
                    Else if difficulty variable matches 1
                    Call diff_name function to draw MEDIUM mode
                    Else if difficulty variable matches 2
                    Call diff_name function to draw HARD mode

                If mouse click detected inside detection box of play button
                    Change “current_state” variable to “game”
                    Call choose_word function to randomly select a word depending on selected difficulty
                    Call draw function to replace menu with game

If “current_state” variable is “game”
If mouse click detected inside radius of circular letter button and it is visible
Set letter variable to false (thus making button invisible and showing in display_word variable)
If letter clicked was not inside word
Increase hangman_status by 1
Call draw function to update screen

If mouse click detected inside detection box of back button
If shown_warning variable equal to 1
Increase lose_amount by 1
Reset guessed letter list back to empty
Reset difficulty and hangman_status back to 0
Set letter variable back to True
Set “current_state” variable to “menu”
Draw menu by calling menu function

Set shown_warning back to 0

If shown_warning variable equal to 0 and current_state is “game”
Render warning text
Update pygame display
Increase shown_warning variable by 1

If mouse click was not detected inside detection box of back button and shown_warning = 1
Reset shown_warning variable back to 0

Set won variable to True
If a letter in display_word remains unguessed and not in guessed variable list
Set won variable to False
Break game loop

If won
Display message of “You WON!” by calling the display_message function
Increase win_amount variable by 1
Reset guessed letter list back to empty
Reset difficulty and hangman_status back to 0
Set letter variable back to True
Set “current_state” variable to “menu”

Draw menu by calling menu function

If hangman_status = 9
Display message of “You LOST!” by calling the display_message function
Increase lose_amount variable by 1
Reset guessed letter list back to empty
Reset difficulty and hangman_status back to 0
Set letter variable back to True
Set “current_state” variable to “menu”
Draw menu by calling menu function

Call main function
Terminate pygame library when no more game loop active