Code Representation: Algorithm

Constructor and Main Function

1. Declare preprocessor directives
2. Declare class
    a. Inside class declare constructor function and all necessary game functions as public.
    b. Inside class declare all necessary variables as private.
3. Declare the object for the class globally for use in all functions
4. Start main function
    a. Set up infinite while loop to keep game running indefinitely
        i. Run Slot_Centers
        ii. Run Menu_Function
        iii. Run GameLoop
        iv. Run CreditDisplay
        v. Run BoardClear
5. Initialize constructor function
    a. Declare necessary variables
    b. Set all necessary object variables to their default value

Slot_Centers

1. Declare necessary variables for looping
2. Use for loop to initialize the pixel locations for each slot center to give ability to place tokens
3. Store x and y locations in respective array

Menu_Function

1. Run Display_Menu
2. Declare necessary variables
3. Use do while loop to loop to check which menu button is pressed and the appropriate action to take
    a. Run Check_Touch
    b. Run Menu_Button_Check
    c. Use switch case for variable menu_choice to determine what value was entered for menu choice
        i. Case 1
            a. Run BlankBoard
            b. Set menu_loop to 0 to end do while loop
            c. Break
        ii. Case 2
            a. Run Instructions_Button
            b. Set menu_loop to 0 to end do while loop

c. Break

iii. Case 3

a. Run DisplayStats
b. Set menu_loop to 1 to go back to menu
c. Break

iv. Default

a. Set menu_loop to 1 to continue looping
b. Break

d. Loop while menu_loop is equal to 1

Display_Menu

1. Set background color to Spring Green
2. Clear existing screen, leaving only background color
3. Write the words Quad in red and Connection in Blue across the top of the screen
4. Write the words start and instructions in red and blue respectively and place each word inside of an appropriate colored rectangle, centered on screen
5. Write credit statements at bottom of screen in black

Check_Touch

1. Declare necessary variables for coordinates
2. Use while loop to make sure nothing happens while screen isn't being tapped
3. Use while loop to register coordinates and store in tapx and tapy variables when a tap is sensed

Menu_Button_Check

1. Use if statement to check if start was pressed. If tapy is between 95 and 125 pixels and tapx is between 115 and 195 pixels
   a. Set menu_choice equal to 1
2. Use else if statement to check if instructions was pressed. If tapy is between 135 and 160 pixels and tapx is between 95 and 245 pixels
   a. Set menu_choice equal to 2
3. Use else if statement to check if stats button was pressed. If tapy is between 215 and 240 and tapx is between 95 and 245 pixels
   a. Set menu_choice equal to 3
4. Else
   a. Set menu_choice equal to 0 to jump back to check touch

Instructions_Button

1. Run Display_Instructions
2. Run Check_Touch
3. If touch is sensed
    a. Run BlankBoard

Display_Instructions

1. Clear screen and set background color to Spring Green
2. Write to the screen "How to play: Each player takes a turn placing tokens in the slots. The tokens stack on top of each other. The first player to get 4 in a row wins. Four in a row can be vertical, horizontal, or diagonal. Tap anywhere to continue"

BlankBoard

1. Set looping variable n to 0
2. Clear the screen to a Spring Green background
3. Set the font color to gray
4. Fill a rectangle on the screen
5. Set the font color to scarlet
6. Draw out horizontal and vertical lines to create a 5x5 grid on the gray rectangle

GameLoop

1. Use while loop to run until loop variable n is changed from 0
    a. Run PlayerBlue
    b. Run WinState
    c. Use if statement to determine if n is still equal to 0, if so
        i. Run PlayerRed
        ii. Run Winstate

PlayerBlue

1. Set font color to black
2. Write "Blue Turn" at the top of the screen
3. Set token_type to 1
4. Run Check_Touch
5. Run Column_Check
6. Run Find_Slot
7. Set font color to blue
8. Run Fill_Slot

PlayerRed

1. Set font color to black
2. Write "Red Turn" at the top of the screen
3. Set token_type to 2
4. Run Check_Touch
5. Run Column_Check
6. Run Find_Slot
7. Set font color to red
8. Run Fill_Slot

Column_Check

1. Use if statement to determine column of tap
    a. If tapx is between 0 and 140 pixels
        i. Set column_index to 0
    b. Else if tapx is between 140 and 180 pixels
        i. Set column_index to 1
    c. Else if tapx is between 180 and 220 pixels
        i. Set column_index to 2
    d. Else if tapx is between 220 and 260 pixels
        i. Set column_index to 3
    e. Else if tapx is between 260 and 319 pixels
        i. Set column_index to 4
    f. Else
        i. Set column_index to 0

WinState

1. Declare all necessary counting/looping variables
2. Initialize all necessary counting variables
3. Set object variable n equal to 0
4. Use for loop with inner if statements to loop to find win states for player 1 horizontal and vertical lines
    a. If win state is found, set n equal to 1
5. Use for loop with inner if statements to loop to find win states for player 2 horizontal and vertical lines
    a. If win state is found, set n equal to 2
6. Use for loop with inner if statements to loop to find win states for player 1 main diagonal lines
    a. If win state is found, set n equal to 1
7. Use for loop with inner if statements to loop to find win states for player 2 main diagonal lines
    a. If win state is found, set n equal to 2

8. Use for loop with inner if statements to loop to find win states for player 1 minor diagonal lines
    a. If win state is found, set n equal to 1
9. Use for loop with inner if statements to loop to find win states for player 2 minor diagonal lines
    a. If win state is found, set n equal to 2
10. Use for loop with inner if statements to loop to count number of blank values in board array
    a. If count is equal to zero, set n variable equal to 3
11. If n is equal to 1
    a. Write "Blue Winner" at the top of the screen and write "BLUE!!!" repeatedly down the side of the screen
    b. Sleep for 4 seconds
12. If n is equal to 2
    a. Write "Red Winner" at the top of the screen and write "RED!!!" repeatedly down the side of the screen
    b. Sleep for 4 seconds
13. If n is equal to 3
    a. Write "Tie Game" at the top of the screen and write "TIE!!!" repeatedly down the side of the screen
    b. Sleep for 4 seconds

Find_Slot

1. Declare necessary looping variables
2. Use for loop with inner if statement to loop from bottom row of array to top
    a. If board index with row i and column column_index is equal to 0, or unfilled
        i. Set row_index equal to i
        ii. Run Fill_Slot
        iii. Break

Fill_Slot

1. Fill in a circle at x pixel position, x_slot_center[column_index] and y pixel position, y_slot_center[row_index] with radius of 20 pixels
2. Set the array board index for the corresponding spot equal to the token_type, 1 for blue, 2 for red, 0 for blank

BoardClear

1. Declare necessary looping variables

2. Use for nested for loop to loop through each array index for board array
    a. For each index, set the value to 0

CreditDisplay

1. Clear the board and set background to Spring Green
2. Set the font color to black
3. Write to screen “Thank you for playing! We hope you play again, -Michael and Jake”
4. Sleep for 3 seconds

DisplayStats

1. Clear the board and set background to Spring Green
2. Set the font color to blue
3. Write to screen the number of times the blue player won
4. Set the font color to red
5. Write to screen the number of times the red player won
6. Set the font color to blue
7. Write to screen “Tap Anywhere to Return to Main Menu”