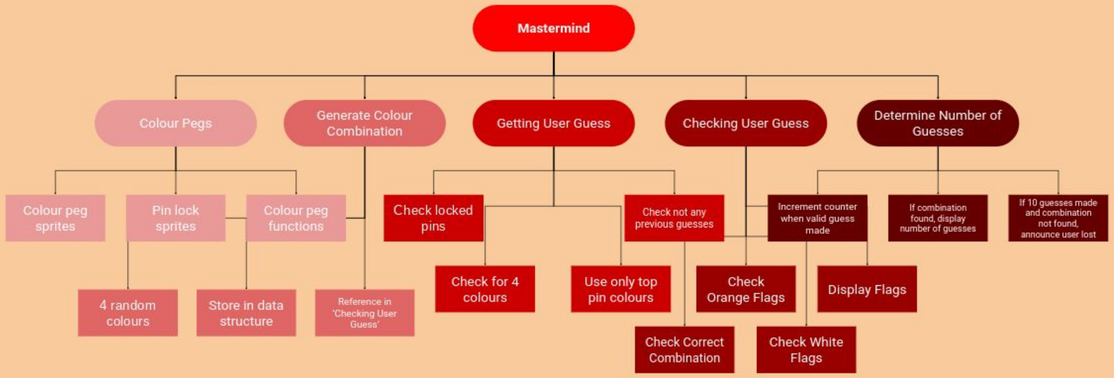
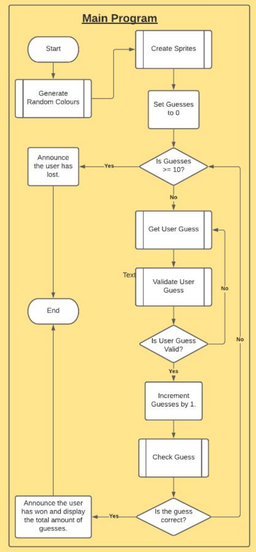
Analysis

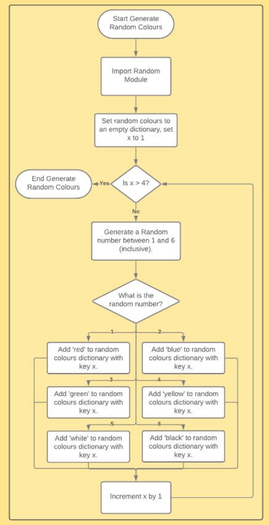


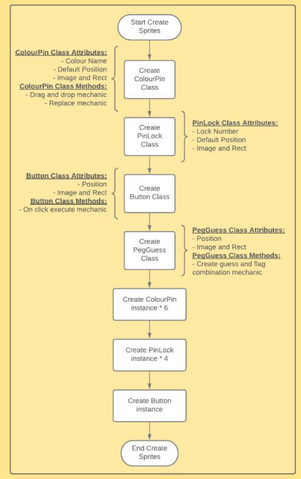
Program Criteria

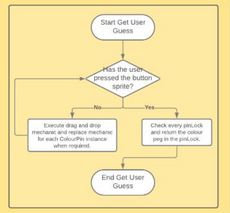
1. **My program must have interactive colour pegs that the user can move.**
   - 1.1. My program must have a collection of sprites representing the colour pegs the user can choose (to guess the colour combination with).
   - 1.2. My program needs to have an area where the user can lock colour pins in place for their guess.
   - 1.3. There must be a function to create a new colour pin of the same colour so the user can guess the same colour for more than one colour slot.
2. **My program must create a random colour combination that the user needs to guess to win the game.**
   - 2.1. There must be a random colour generated 4 times, to create a colour combination that the user has to guess to win.
   - 2.2. This colour combination must then be stored to be used later in the program.
   - 2.3. This colour combination must never change, and must be referenced when the user makes a guess.
3. **My program must have a function to determine the colours that the user guesses.**
   - 3.1. The colours that the user guesses must be determined by the program using the colour pins they have 'locked in'.
   - 3.2. The colours they have chosen must also be validated to make sure that they have guessed a total of four colours.
   - 3.3. The program must make sure that the determined colours are indeed the colours that the user have chosen, and not any of the previous colours that were locked in.
   - 3.4. The program should finally validate the guess by comparing the guess to previous guesses, and seeing if the user has guessed the exact colour combination before.
4. **My program should determine which guesses are correct and determine the number of orange/white flags to display.**
   - 4.1. The program should first check to see if the guess matches the generated combination. If this is true, the game should end.
   - 4.2. If the guessed combination does not match the generated combination, the program should start determining the number of flags (orange and white) to display.
   - 4.3. The program should first go through each colour in order, to see if the guessed colour matches the exact location of the colour in the generated combination. The number of orange flags must be incremented if this is true, and this colour should then be ignored for the next step.
   - 4.4. The program should then go through the rest of the colours that are not the correct colour in the correct place, and determine which colours are in the combination to determine the number of white flags to display.
5. **My program should determine the number of guesses after the combination has been guessed.**
   - 5.1. The program should be keeping track of the number of guesses made by the user throughout the game, incrementing the number of guesses only when a validated guess has been made.
   - 5.2. After the colour combination has been guessed, the program should display the number of guesses made by the user.
   - 5.3. If the user guesses 10 times and the combination has not been found, the program should conclude the game and say that the user has lost.

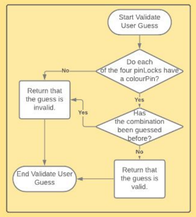
Design

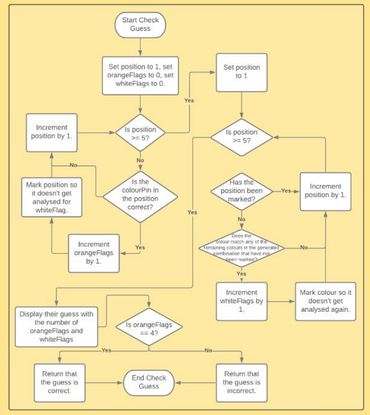


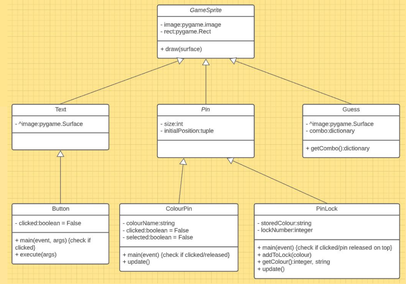












Implementation

Testing

- Mastermind
  - Colour Pegs
    - eg s
    - Pin lock sprites
    - Colour peg functions
  - Generate Colour Combination
    - 4 random colours
    - Store in data structure
    - Reference in ‘Checking User Guess’
  - Getting User Guess
    - Check locked pins
    - Check not any previous guesses
    - Check for 4 colours
    - Use only top pin colours
  - Checking User Guess
    - Check Orange Flags
    - Display Flags
    - Check Correct Combination
    - Check White Flags
  - Determine Number of Guesses
    - Increment counter when valid guess made
    - If combination found, display number of guesses
    - If 10 an ann