Computer Science Final Project - Written Portion

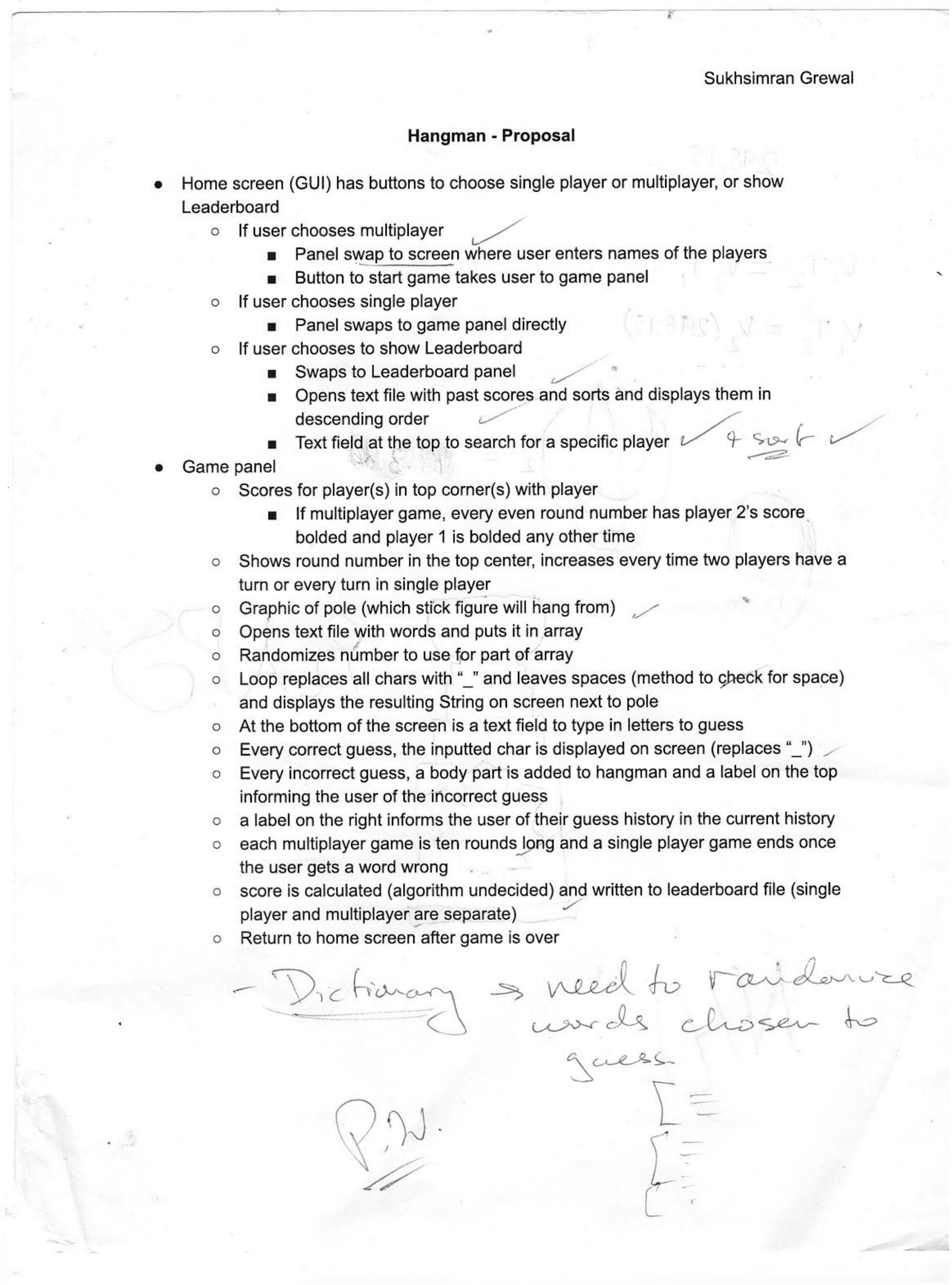
Sukhsimran S. Grewal

Heart Lake Secondary School

ICS3UO

Software Design Process

Proposal - V1 (originally signed copy)



Sukhsimran Grewal

Hangman - Proposal

- Home screen (GUI) has buttons to choose single player or multiplayer, or show Leaderboard
  - If user chooses multiplayer ✓
    - Panel swap to screen where user enters names of the players
    - Button to start game takes user to game panel
  - If user chooses single player
    - Panel swaps to game panel directly
  - If user chooses to show Leaderboard
    - Swaps to Leaderboard panel ✓
    - Opens text file with past scores and sorts and displays them in descending order ✓
    - Text field at the top to search for a specific player ✓ + sort ✓
- Game panel
  - Scores for player(s) in top corner(s) with player
    - If multiplayer game, every even round number has player 2's score bolded and player 1 is bolded any other time
  - Shows round number in the top center, increases every time two players have a turn or every turn in single player
  - Graphic of pole (which stick figure will hang from) ✓
  - Opens text file with words and puts it in array
  - Randomizes number to use for part of array
  - Loop replaces all chars with "_" and leaves spaces (method to check for space) and displays the resulting String on screen next to pole
  - At the bottom of the screen is a text field to type in letters to guess
  - Every correct guess, the inputted char is displayed on screen (replaces "_") ✓
  - Every incorrect guess, a body part is added to hangman and a label on the top informing the user of the incorrect guess
  - a label on the right informs the user of their guess history in the current history
  - each multiplayer game is ten rounds long and a single player game ends once the user gets a word wrong
  - score is calculated (algorithm undecided) and written to leaderboard file (single player and multiplayer are separate) ✓
  - Return to home screen after game is over

- Dictionary ⇒ need to randomize words chosen to guess.

P.W.

<u>Proposal - V2</u>

The problem in question is that a game needs to be developed that meets several requirements set by the teacher, including writing and reading to files, using loops, implementing procedures and functions, using if statements, and much more. The chosen game to meet the aforementioned requirements is Hangman (movie version). The game consists of three different classes in three different files. The first being the home screen, where the user will begin. There are two buttons on this frame; one button to access the game screen class and the other for the leaderboard class. If the user chooses to start the game, the home screen class will be disposed of and the game screen class will open. The game screen interface will consist of a label showing the score, a pole (image) for the hangman, a button to return to the home screen, a text field for the user to enter a letter into, a button to confirm the user's letter entry, the word that needs to be guessed in a label, a label for messages informing the user of their progress, and another label showing the user the labels they've already guessed. The word that needs to be guessed will be accessed from a text file (using array) and then the program will randomly choose the word. The user will have a maximum of six wrong entries into the text field per word. Loops will be used to go through the word and replace the guessed characters. If the user loses by running out of wrong guesses allowed, the body of the hangman will have fully formed, the information label will inform the user that they lost, a JOptionPane message will show the user their score, a JOptionPane input dialogue will ask for the user's name so that the program can store the name and score of the user into a file to be accessed in the leaderboard frame, and then they will be taken back to the home screen after the game screen frame has been disposed of. If all the letters of the "String" are guessed, the class will run again with the body of the hangman and the number of wrong guesses being reset except the score counter will go up using a "static int" variable for the score counter. This will continue until they lose. In terms of the leaderboard class for if the user chooses to view the leaderboards, the frame will comprise of a text field to search for a certain player and two text areas (one to see leaderboard and the other for the user's search results). The leaderboard information will be read into the program from a text file, sorted from highest to lowest score, and then finally displayed in the corresponding JTextArea.

Other specific requirements will be discussed in this section, beginning with procedures and functions. An example of a function in the game will be finding a word from the text file and then returning the word (String). An instance where a procedure will be used is displaying the leaderboard (using file reader, putting info into arrays, sorting, then displaying names and scores). Lastly, several built-in functions will be used such as setVisible to control the visibility of the already placed body parts depending on the number of guesses available and toUpperCase to properly compare the "hidden" word to the original word to see if the user won.

<u>IPO Charts</u>

Three IPO charts will be shown here. One for just the gameplay portion of the game, one for the player search, and one for the leaderboard display:

1.

Input	User's guessed letter (formed after String that needs to be guesses is randomly chosen from text file, encrypted, then shown on screen for the user to guess its characters)
Process	● Check if the original String contains user's entered letter (after accessing text field's first char) ● If true ○ Go through hidden string and replace "*" with original word's letter in that same position ○ Find message that suits situation ○ If the modified hidden string equals the original string, then restart class ● If false ○ Reduce number of wrong guesses available to user ○ Find message that suits situation ○ If number of wrong guesses = 0, ask for name and write name and score into text file (for leaderboards) and also return to the home screen
Output	● If correct letter… ○ Display found message ○ Show modified hidden string ○ If they got the word, display JOptionPane with congratulatory message ● If incorrect letter… ○ Display found message ○ Show body part of hangman depending on how many wrong guesses are available ○ If wrong guesses = 0, inform the user that the game is over

	along with a JOptionPane with their score

2.

Input	User enters name they want to search for
Process	● Access text file with names and scores ● Put names and scores into arrays (use bits to split each line as they are in the same line) ● Close text file ● Get text from text field and check if it equals any of the names in name array
Output	If name is found, display name with corresponding score in JTextArea If name is not found, display "NAME NOT FOUND" in JTextArea

3.

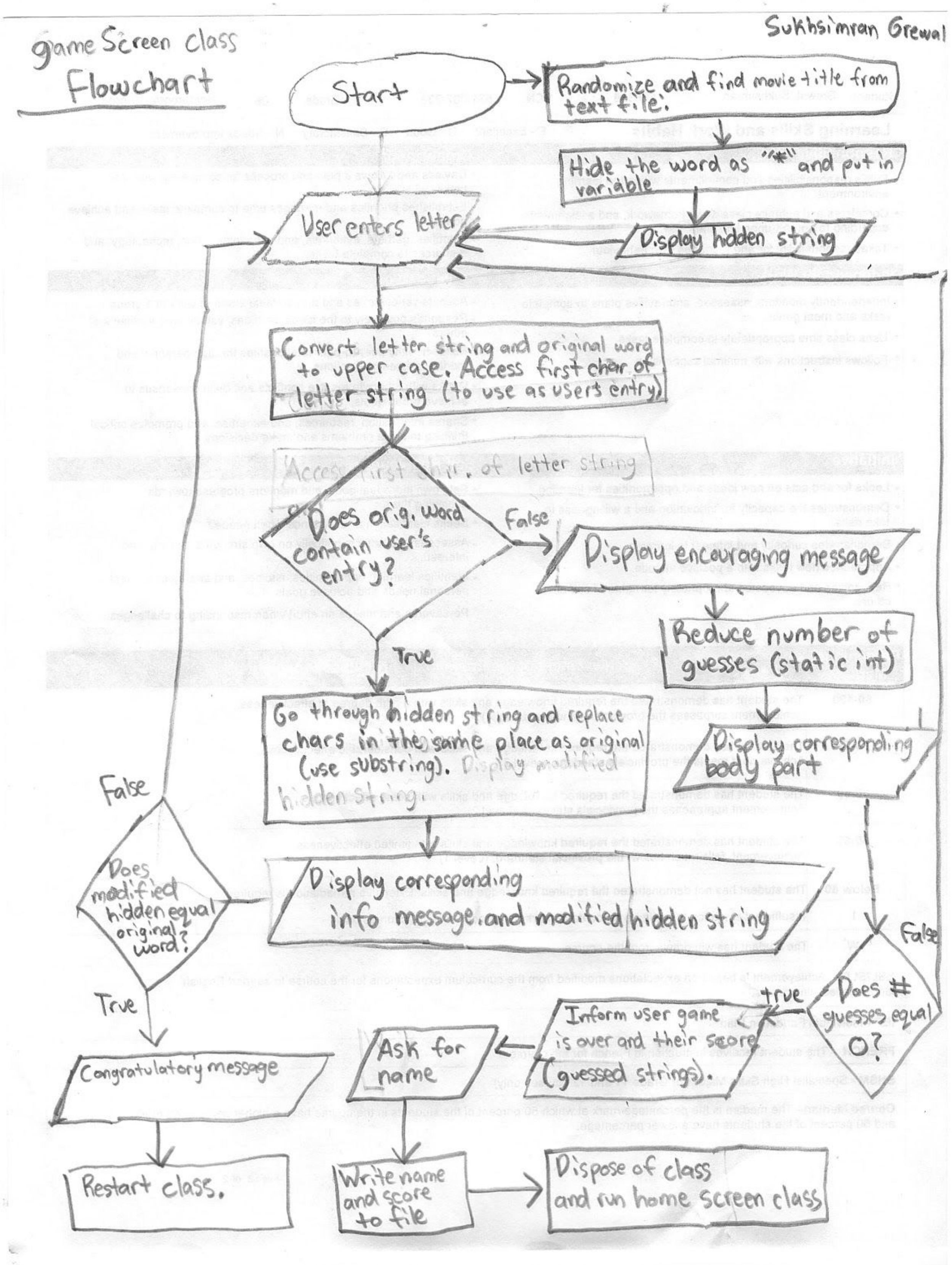
Input	User clicks button to show leaderboard
Process	● Access text file with names and scores ● Put names and scores into arrays (use bits to split each line as they are in the same line) ● Close text file ● Bubble sort the data based on descending order
Output	Output the sorted information on JTextArea

<u>Pseudocode - gameScreen class</u>

```
Start
Set static int scoreCount as 0
Set number of guesses as 6
Find random word and put into variable
        Access text file with movie names
```

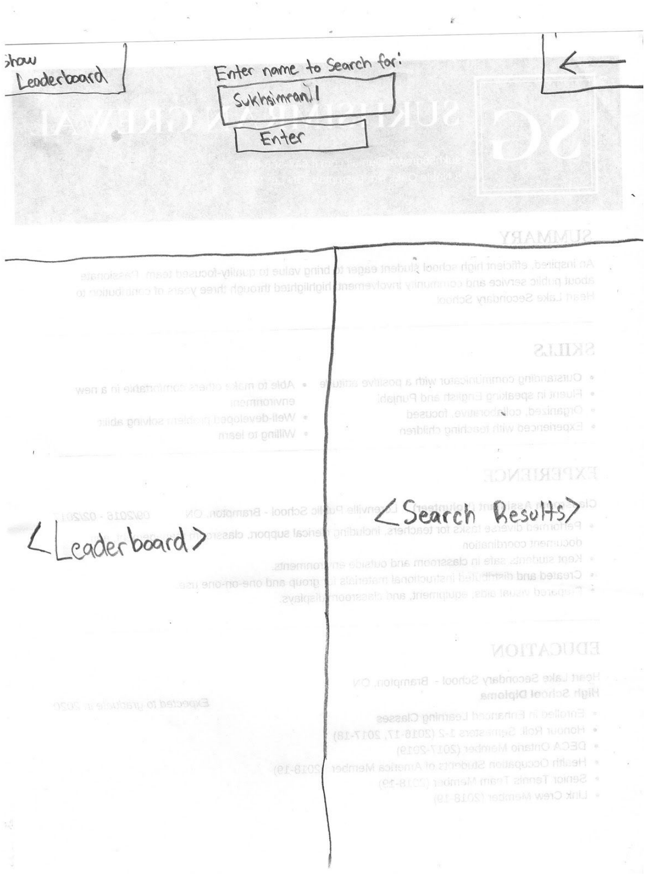
```
    Put names into array
    Randomize number
    Use number as array number
    Return String
Hide the movie title characters
    Declare empty String as hidden
    For loop for title’s length
        If ascii value is between 32 and 63, then add movie title’s char to initially empty String
        Anything else: add “*” to hidden string

Access what user entered into text field as letter after they press enter button
Convert user’s String to upper case along with original word
If original title contains letter, go through each char from original title using loop and when user’s letter is found, replace hidden String’s corresponding “*” with user’s letter (use substring)
    Set text label to now display modified hidden string
    Display appropriate message on another label
If original title does not contain letter, display appropriate message and subtract from guess variable
Any subtractions in guess variable will result in another body part for hangman appearing (image is set visible)
Once guess number hits zero, user is informed game is over through label, told what their score is, and asked for their name
    Name and score are written to file
    scoreCount is reset
    gameScreen class is disposed of
    homeScreen class opens
If modified hidden word equals original title (before guess hits zero)
    JOptionPane informs the user they got the word
    +1 to scoreCount
    gameScreen is disposed of
    Runs class again (scoreCount still displays as right score because it is static int variable, however, guess resets automatically as it is not)

Button in the top right to go back just disposes of class, resets scoreCount, and opens homeScreen class
```

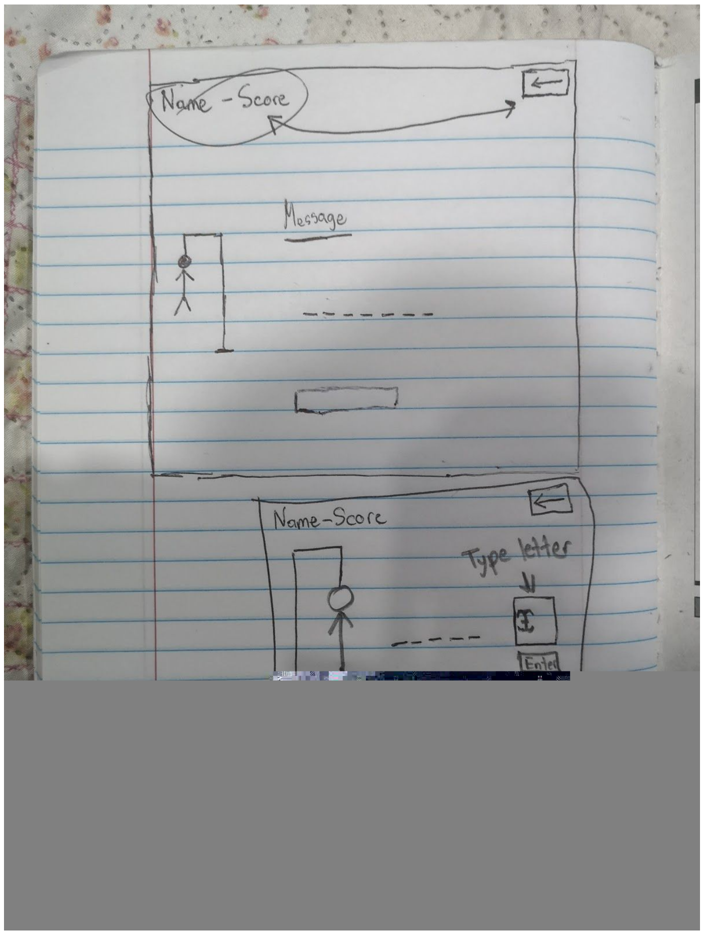
Flowchart

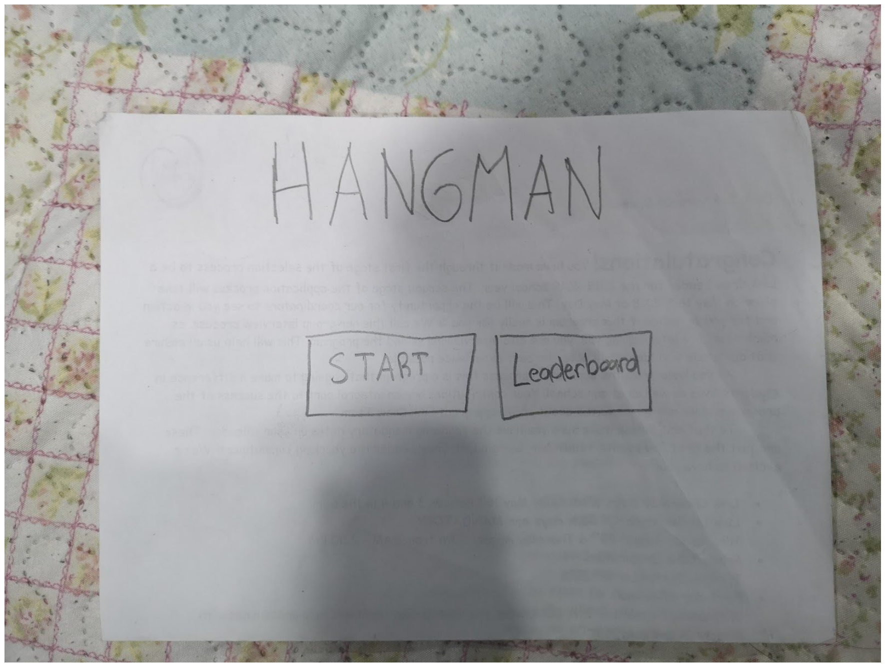

gameScreen class
Flowchart
Sukhsimran Grewal
Start
Randomize and find movie title from text file.
Hide the word as "*" and put in variable
Display hidden string
User enters letter
Convert letter string and original word to upper case. Access first char of letter string (to use as users entry)
~~Access first char. of letter string~~
Does orig. word contain user's entry?
False
Display encouraging message
Reduce number of guesses (static int)
Display corresponding body part
True
Go through hidden string and replace chars in the same place as original (use substring). ~~Display modified hidden string~~
Display corresponding info message. and modified hidden string
False
Does modified hidden equal original word?
True
Congratulatory message
Restart class.
False
Does # guesses equal 0?
true
Inform user game is over and their score (guessed strings).
Ask for name
Write name and score to file
Dispose of class and run home screen class

<u>User Interface Design</u>

These are my sketches for how I visualized my JFrames to originally look - Leaderboard, Game Screen, Home Screen (shown below in order):







User Instructions

To begin with, you need to decide on if they want to play the game or look at leaderboard by clicking on the corresponding button.

Game

1. When playing the game, the instructions are very elementary: the user enters a letter they believe could be found in the hidden movie title.
2. Once you type it in, you click the button to enter it into the program.
3. If you guessed a correct letter, it appears everywhere found in the title and a label gives you a message informing you that you were correct.
4. If you guessed an incorrect letter, a body part of the man being "hung" appears and an encouraging message that also informs you of being wrong.
5. You continue to guess until you guess all the letters of the movie title or you run out of wrong guesses (body is complete at 6 incorrect guesses).
    a. If you lost, the game ends and shows you your score (determined by how many titles you guessed right). You type in your name for leaderboard.
    b. If you guessed the title right, you move onto a different title and play until you lose.
6. You will be returned to the home screen once you eventually lose or decide to quit when you press the "back" button on the top right of the screen.

Leaderboard

1. Click "Show Leaderboard" to view scores of past users in descending order.
2. Type in a specific user and press the corresponding enter button below the text field at the top to see their score(s) to the right of the leaderboard.
3. Press "back" button in the top right to return to the home screen.