**Feasibility Study**

**Step 1: Problem Definition**

The application that I am planning on making is a Hangman Game. This is a game where the user must try to guess a hidden word by choosing letters of the alphabet in order to uncover the letters of the hidden word. Once the user has uncovered all of the letters of the word, and has thus revealed the hidden word itself, they will have won. However, the user has a limited number of incorrect guesses. Every time the users guess an incorrect letter, a body part is added to the “Hangman” image. Once the full body of the man is revealed, the user will have lost. Usually the user will be able to guess 6 incorrect letters before losing.

I have decided to create a Hangman game application because Hangman is a very popular pen and paper game, but there are not many well developed versions of it online. The lack of online Hangman games may make it harder for people on the internet to find a game that they enjoy playing, when Hangman may be the right game for them. So, I believed that this would be a great opportunity to create a project that solves this issue.

The users of this program will usually consist of internet users, young students, and other computer programmers. Everyday internet users may use this application to have access to a fun, enjoyable game to simply pass time. Young students may use this application in order to expand their vocabulary and challenge their minds by using their problem solving skills. Beginner computer programmers may want this application so that they have access to a well developed software algorithm that they can use for reference when creating their own projects, and to have a software application that they can use as a base to create a more sophisticated project.

**Step 2: Problem Analysis**

Constraints/problems expected to be encountered, and their solutions:

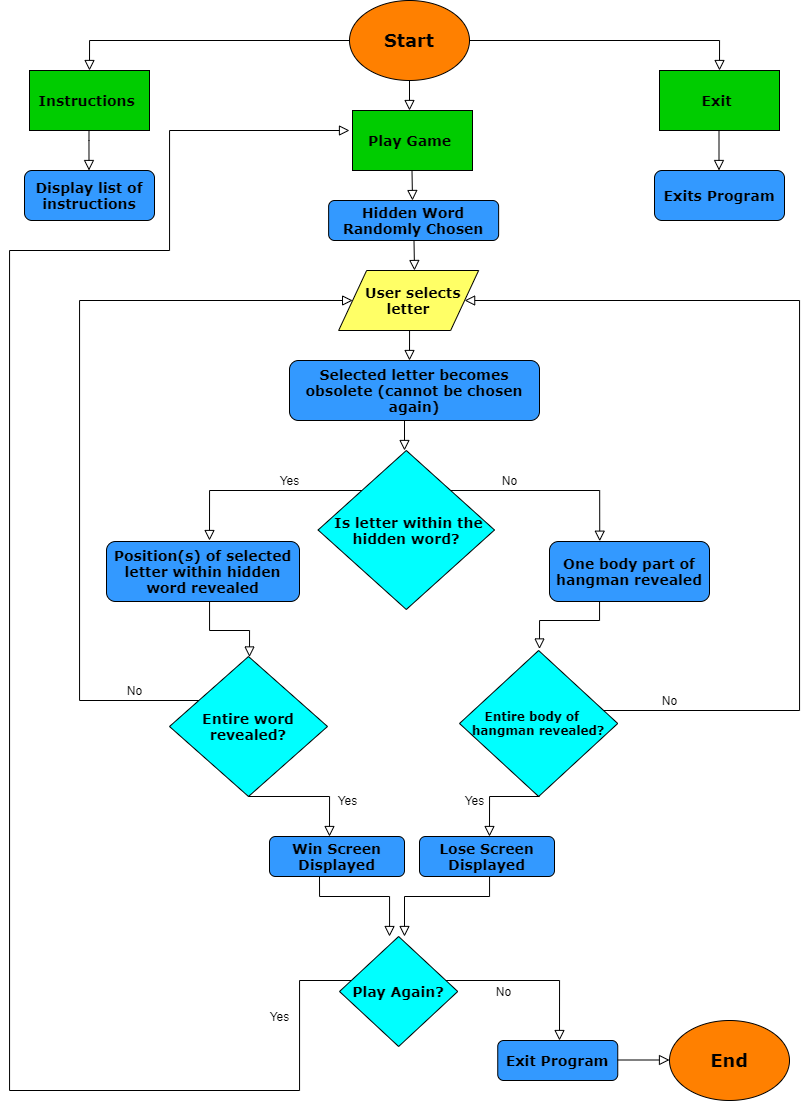
* Learning how to implement sound effects
  + I am planning on adding sound effects to my Hangman Game, but I do not know how to use them yet. So, I must do research on how to properly and efficiently implement them into my program.
* Learning how to add different screens
  + I am planning on adding different screens to my application. I would like to add an introduction screen: which allows the users to either begin playing, go over the rules, or exit, a main game screen: where the users will be able to play the Hangman Game, and Win/Lose screens: which indicate if the users have won or lost. However, I do not know how to add different screens into a Java program, so I will have to conduct research in order to figure out how to do so.
* Compatibility:
  + The Java application that I am creating may not be compatible for some users. Java applications only work with recent/up-to-date versions of certain operating systems, such as Windows, MacOS, Linux and Solaris OS. To solve this issue, I can put a disclaimer that tells the users which Operating Systems this program will work with.
  + I can also put a disclaimer that tells the users to download a JVM (Java Virtual Machine). This will allow for the users with a compatible OS to be able to run any Java application, including the one that I will be creating.
* Hardware Required:
  + The users of this application will require an up to date, modern computer. This computer must be able to run Java applications, including this one, by having the required memory capacity to install and run this application. Users with obsolete/old computers will not be able to use this application. This problem can be solved by having a disclaimer for the users that states the minimum hardware requirements to run this application.
* Recommended age of users:
  + The recommended age for the users of this application is anyone who is 6 years old or older. This is because those below this age may not fully understand the rules of the game, as well as how to use the application itself. I will put another disclaimer, which states the minimum age to use this application.
* Time users will need to learn how to use this program:
  + It will not take a very long time for the users to learn how to use this program, because in the introduction screen, I will provide a full set of instructions, which will explain to the users exactly how to use this application.

**Step 3: Software Project Plan**

Statement Of Work (SOW)

The features that will be included in this application include:

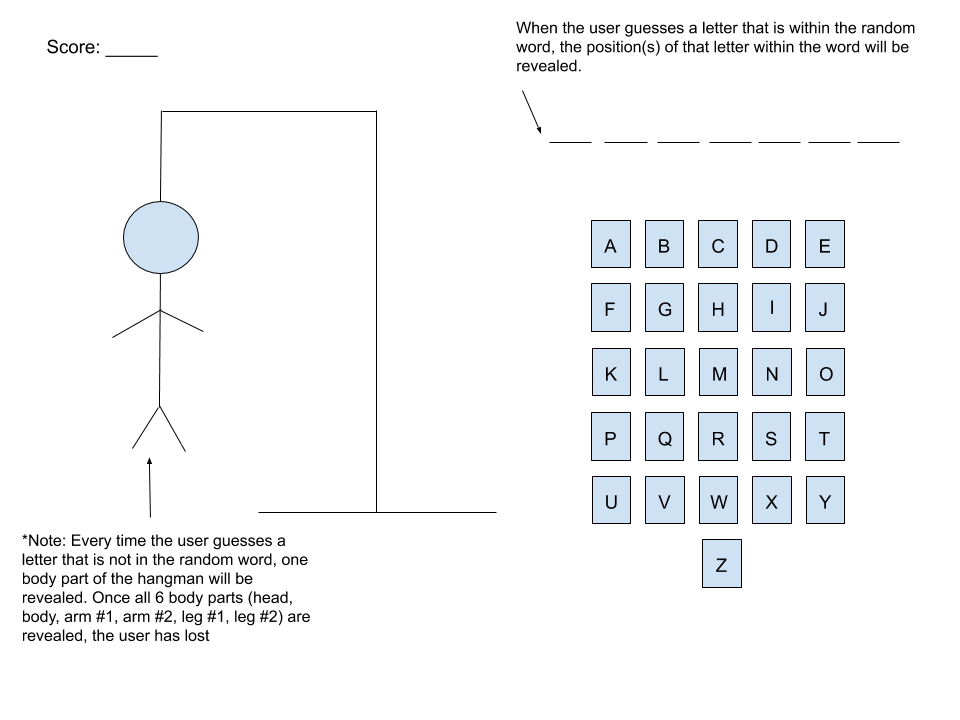
* Introduction, Game & Win/Lose screens, Word Selection, Letter Selection, Hangman GUI, Word Update, Score Count and Sound Effects.

Flowchart:

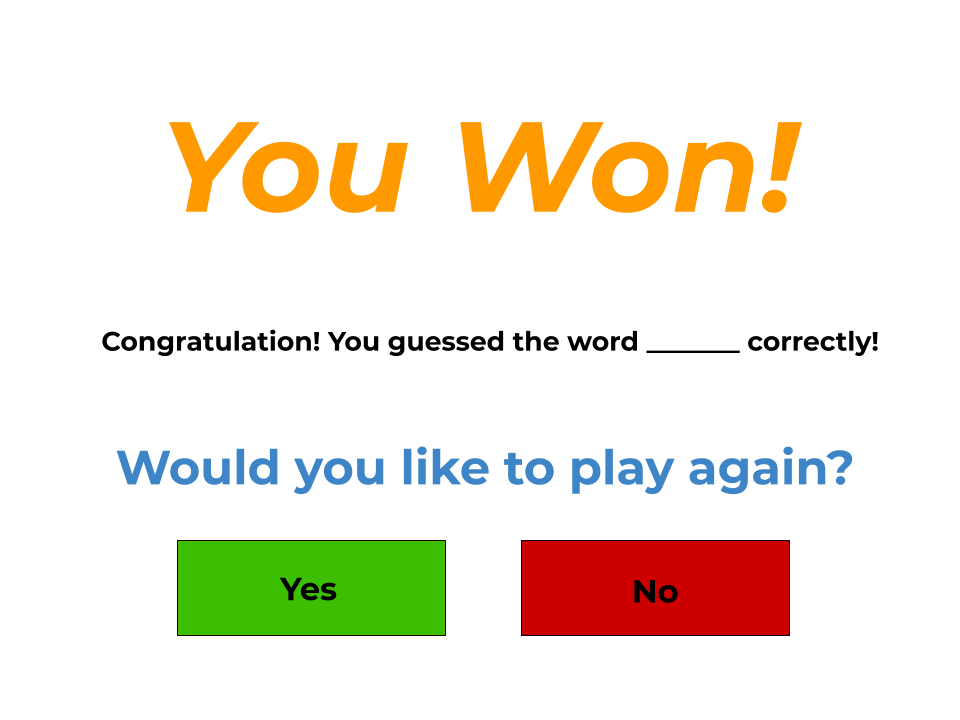
Introduction Screen:



Main Game Screen:



Win Screen:



Lose Screen:



Resource List:

* A modern PC that has Java Netbeans and a Java Virtual Machine installed on it
* Mouse, Keyboard, Monitor and Headphones
* Access to electricity
* Access to a reliable source of wifi
* Quiet room to be able to work in (mainly my bedroom, which has a full computer setup)
* USB Stick to save data regularly
* My teacher, who is available to assist me whenever I need help
* My father, who is always available to check my work and provide feedback

Work Breakdown

|  |  |
| --- | --- |
| **Feature of Java application** | **Time Estimate** |
| Introduction Screen | 1 hour |
| Main Game Screen | 1 hour |
| Win/Lose Screens | 1 hour |
| Word Selection | 1 hour |
| Letter Selection | 3 hours |
| Hangman GUI | 2 hours |
| Word Update | 4 hours |
| Score Count | 2 hours |
| Sound Effects | 3 hours |
| Total Time Estimate | 18 hours |

Project Schedule (From Gantt Chart) \*\* One Work Day = 3 Hours Of Work \*\*

|  |  |  |  |
| --- | --- | --- | --- |
| **Sub Task** | **Start Time (Day #)** | **End Time (Day #)** | **Length of Task (In "Work Days")** |
| Scope Document | 1 | 3 | 2 |
| Feasibility Study | 3 | 5 | 2 |
| Introduction Screen | 5 | 5 & 1/3 | 1/3 |
| Win/Lose Screens | 5 & 1/3 | 5 & 2/3 | 1/3 |
| Main Game Screen | 5 & 2/3 | 6 | 1/3 |
| Game | 6 | 11 | 5 |
| Encapsulation | 11 | 12 | 1 |
| User Guide | 12 | 14 | 2 |
| Testing | 14 | 15 | 1 |
| Reflection Report | 15 | 17 | 2 |
| Total Time | 1 | 17 | 16 work days (or 48 hours) |

Risk Plan:

Potential risks and how they will be solved:

* There could be a risk in estimating the time schedule, as it may take longer than expected to finish the project.
  + This can be dealt with by “overestimating” on the time schedule. This would mean that for each sub-task of this project, I can set the expected time period for its completion to an amount of time that exceeds the actual amount of time that I believe I will take to complete it. This way, if each sub-task takes longer than expected to complete, it will not be as big of a deal, as the inflated time estimates for each sub-task will be ready to deal with that.
* The program may not be compatible with all system, which would make it unavailable to some users who may be interested in this application, but have, say, an incompatible Operating System
  + This can be solved by telling the users beforehand that they must download a Java Virtual Machine (JVM) in order to access and use this Java application. This application can only be run on a computer that has the JVM installed on it.
  + I can also tell the users which Operating Systems are compatible with this Java application, as only specific Operating Systems and their versions can run Java applications effectively.
* If the program is not appropriately backed up somewhere, then there is an increased risk of the program being lost, and all of its data being deleted.
  + This can be resolved by ensuring that my work is saved in a safe location, which for me, will be on a USB Drive. I will also save this application on the Hard Drive of my computer, so that if my USB fails to save my application, I will always have an additional copy of it available, meaning that my application will be “backed up”.
* There may be various logical errors within the application.
  + These can be fixed by going through, and tracking each and every line of code to ensure that the desired result is achieved every time the application is executed. Also, documenting the application will make it easy to see what each line of code is doing, and if any adjustments/fixes have to be made.
* There may be a security vulnerability for this application.
  + This can be solved by installing a software security system (such as McAfee) on my computer (which I already have), in order to prevent the unauthorized access of my information, including the application that I am creating. I can also advise the users of the program to do so as well, in order to ensure the security of their computers and information.