Documentation Template

|  |
| --- |
| ***You MUST provide evidence showing how the problem has been decomposed, how the components have been developed and trialled, and of how they have been assembled and tested to create a final, working outcome.*** |

### **Outline / Decomposition**

*Please write down your task decomposition here (a numbered list is a good idea)*

1. Ask the user if they have played the game before, if not then show the instructions.
2. Ask user to pay an initial amount for how many rounds they want to play, $1 per round, maximum amount of money the user can spend is $10 per session
3. Generate a random token that is either a zebra, horse, donkey or unicorn. If the token is a unicorn, the user wins $5, if it is a horse or donkey, they win $0.50 and if it is a donkey, they don’t win anything.
4. Work out how much money the user won / lost and how much they have left each round.

### **Flowchart**

*Please show a developed flowchart of your program below (you may use draw.io to create your flowchart)*

*Diagram, engineering drawing

Description automatically generated*

### **Version Log**

*Your version log should go here. Annotated screenshots are a good idea at this point*

*Text

Description automatically generated*

**Trial 1 of the yes/no checker for the instructions:**

This piece of code worked for both UPPER and lower-case characters but was inefficient. It has five if / elif / else statements and quite a bit of repeated code.

*Text

Description automatically generated*

**Trial 2 of the yes/no checker for instructions:**

I used the ‘or’ command to combine my ‘yes’ / ‘y’ and ‘no’ / ‘n’ statements and this code is more efficient compared with trial #1.

*Text

Description automatically generated*

**Trial 3 of the yes/no checker (02\_yes\_no\_v1.py):**

I made the code from trial 3 into one function which makes it easier to ask more than one yes / no question in a program. I will use this function in my Lucky Unicorn outcome.

### **Component Testing**

*Show that you have tested each component here. You should have a test plan and then screenshot proof for each component. You should also include notes that justify the major decisions you made.*

**Yes / No Checker Test Plan:**

Graphical user interface, text, application

Description automatically generated

**Yes / No Checker Expected Values Test Plan:**

Table

Description automatically generated

**Yes / No Checker Testing:**

Text

Description automatically generated

All cases worked as expected (code has been looped to make testing easier)

**Instructions (and Yes / No checker) Testing:**

Text

Description automatically generated

Test Case 1 (maybe then ‘yes’) – output is as expected

Text

Description automatically generated

Test Case 2 (no) – output is as expected

The code has been continually refined as at the start it was case sensitive by accepting only lowercase answers and being inefficient. Then the code was enhanced to accept lowercase/UPPERCASE and became efficient through the use of functions.

### Assembled Outcome Testing

*Please show testing for your assembled outcome below. This should include a test plan followed by screenshot proof*

### Usability Testing

*Write a list of things improvements which need to be made based on your usability testing. Then write down what you changed.*

### Post Usability Test…

*Show that your post usability testing program works correctly*

### Social and End User Considerations…

**How did you ensure that your task was suitable for your chosen audience?**

*Answer here*

**How have you honoured copyright?**

*Answer here*

**How did you make your quiz easy to use?**

*Answer here*