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.*** |

Getting Lucky

**Game Summary**

Choose number of rounds 10-30, objective is to reach the specified money goal based off the rounds within the number of rounds.

The money goal = rounds \* 10

Each round user gets to choose 4 options, Loot, Steal, Job, Shop. Choosing any options except shop will pass a round. E.G. Round 1: User chooses Loot, game now moves on to round 2.

**Loot:** gives you a chance to get a medium amount of money but also a chance to not get any money that round.

(Generate random, 80% chance to gain $8 and 20% chance to get nothing that round.)

**Steal:** gives you a chance to get a money but also a chance to lose money that round.

(50% chance to lose 30% of current balance or 50% chance to gain 30% of current balance)

**Job:** Nothing to lose but have to answer a question like 2+2 to get a small amount of money.

($6 if user gets question right and if wrong user gains nothing during that round.)

**Shop:** User can access a shop which displays a menu of upgrades with costs.

(Shop sells upgrades:

Increase reward from loot by $3 for $10

Increase percentage to get money from steal by 10% for $15

Increase reward to $10 from jobs for $20)

When game ends user gets asked if they want to play again.

### Outline / Decomposition

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

1. Base Component
2. Welcome User
3. Yes and No checker
4. Instructions
5. Ask user how many rounds to play (A whole number more than or equal to 10 and less than or equal to 20)
6. Game checks for loot, steal, job, shop and XXX each round.
7. Generate random for loot
8. Workout percentage of balance for steal and 50/50 chance to lose or gain money
9. Generate random question for job
10. Work out balance after each round
11. When user inputs shop display menu of upgrades with cost
12. When user buys from shop minus from balance and apply the upgrade
13. Looping the game / ask user to play again

### Flowchart

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

*Diagram

Description automatically generated*

### Version Log

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

### Component Testing

**Yes / No checker Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| Have you played before? User input: Yes | Game Starts |
| Have you played before? User input: No | Displays instructions and starts game |
| Have you played before? User input: Maybe  Have you played before?  User input: Nope  Have you played before?  User input: 1saf | <error> Please choose y / n    <error> Please choose y / n    <error> Please choose y / n |

**Rounds checker Test Plan (ask user to input number of rounds between 10 – 30)**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| Choose number of rounds:  User input: 10 | Game continues with 10 rounds |
| Choose number of rounds:  User input: 30 | Game continues with 30 rounds |
| Choose number of rounds:  User input: 15 | Game continues with 30 rounds |
| Choose number of rounds:  User input: 7  Choose number of rounds:  User input: 12.5  Choose number of rounds:  User input: 34  Choose number of rounds:  User input: asd | Displays error and asks questions again    Displays error and asks questions again    Displays error and asks questions again    Displays error and asks questions again |

**Choice checker Test Plan (Checks for the inputs <loot>, <steal>, <job>, <shop>**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| User Input: loot  User Input: steal  User Input:  job  User Input:  Shop  User Input:  xxx | You chose <user choice> |
| User Input:  LOOT  User Input:  STEAL  User Input:  JOB  User Input:  SHOP  User Input:  XXX | You chose <user choice> |
| User Input:  123  User Input:  Question  User Input:  ???  User Input:  exit | ERROR, please choose Loot, Steal, Job, Shop or XXX to quit  Ask question again |

**Loot Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| Choose loot 10 times | Loot should be chosen 10 times with a 20% chance to get no money and if they do get money, they gain $8 + current balance |

**Steal Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| Choose steal 10 times | Steal is looped 10 times with a 50/50 chance to lose 30% of current balance or gain 50% of money. (Balance is currently $100 for testing purposes). |

**Steal Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| Steal ran 10 times. |  |

**Job Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| User Input: Correct Answer  User Input: Wrong Answer  User Input: k (Invalid input)  User Input: ‘:’ (Invalid input)  User Input: !@# (Invalid input)  User Input: Correct Answer |  |

**Shop Test Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| User Input: food  User input: 1  User input: loot / steal / job (with sufficient money)  User input: loot / steal / job (with insufficient money)  User input: exit |  |

**End Game Test**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| User input: xxx  If user didn’t meet money goal within the rounds played  User input: invalid |  |

**Play Again Plan**

|  |  |
| --- | --- |
| **Test Case** | **Expected Values** |
| User input: no  User input: yes  User input: invalid |  |

### Assembled Outcome Testing

### 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*

Documentation Templatee

***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.***

Brief

*Describe your game/quiz.*

Outline / Decomposition

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

Flowchart

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

Version Log

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

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.*

Assembled Outcome Testing

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

**Usability Testing (REPEATABLE)**

*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 (REPEATABLE)**

*Show that your post usability testing program works correctly*

Social and End User Considerations…

**Who are your target audience / users?**

*Answer here*

**Why did you need to ensure that your task was suitable for your chosen audience?**

*Answer here*

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

*Answer here*

**Why do we honour copyright?**

*Answer here*

**How have you honoured copyright?**

*Answer here*

**Why do you need to make your quiz easy to use?**

*Answer here*

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

*Answer here*