Project 1: The Highway Code

*Jai mistry*

2019

Contents

[Introduction 2](#_Toc5804609)

[Analysis 2](#_Toc5804610)

[Research 2](#_Toc5804611)

[Success Criteria 3](#_Toc5804612)

[Design 4](#_Toc5804613)

[The Questions 4](#_Toc5804614)

[Testing 5](#_Toc5804615)

[Development 6](#_Toc5804616)

[Evaluation 7](#_Toc5804617)

[Test Results 7](#_Toc5804618)

[Evaluation 7](#_Toc5804619)

# Introduction

In this project, a simple quiz will be created using Python and the easyGUI module. Users will be asked to identify the correct road sign for a stated meaning. At the end of the program, the user should be given a score, and a list of signs to revise.

# Analysis

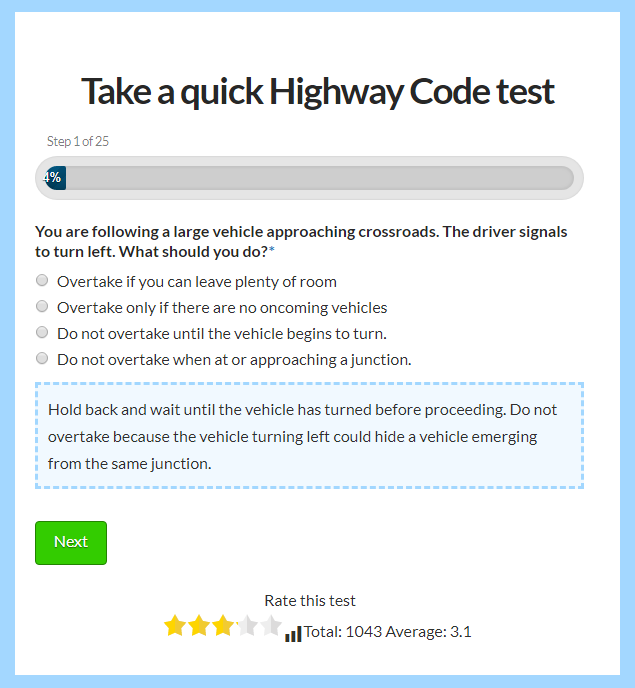
There are several key parts of the problem that make it solvable with Python and easyGUI:

* It needs to show a menu to launch the game.
* Each question needs to be presented visually, with buttons to press.
* Each answer needs to be checked, and a score should be kept.

easyGUI will provide the menus, while Python can check the answers.

## Research

The program should mirror other popular solutions like the highway code test[[1]](#footnote-1):



## Success Criteria

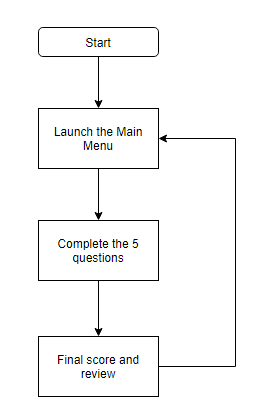
1. A menu should display to let the user begin the quiz, or quit.
2. 5 questions should display, in order.
   1. The user should be presented with a road instruction, and three signs. One of the signs should match the instruction, the other two should not.
   2. If the user selects the correct sign, they should be told that they are correct, and their score should increase by 1.
   3. If the user selects an incorrect sign, the correct sign should be highlighted. The score should not increase.
3. At the end of the quiz, the user should be presented with their final score, and images of any signs they were not able to identify correctly.

# Design

The project can be decomposed into the following parts:

* The main menu
* Questions and scoring
* The final score and incorrect answers.

The program should flow in this manner:



## The Questions

The program should use the following questions:

|  |  |  |  |
| --- | --- | --- | --- |
| Question number | Question | Icons displayed | Correct answer |
| 1 | Which of these warns of falling rocks? | Ahead only, falling rocks, steep hill | Falling rocks |
| 2 | Which of these means ‘No stopping’? | No stopping, no overtaking, ahead only | No stopping |
| 3 | Which of these means there are strong side winds? | Sidewinds, turn left, no entry | Sidewinds |
| 4 | Which of these means no motor vehicles allowed? | No motor vehicles, cycle route, no vehicles over this height. | No motor vehicles |
| 5 | Which of these means no through road? | No through road, opening bridge ahead, uneven roads. | No through road |

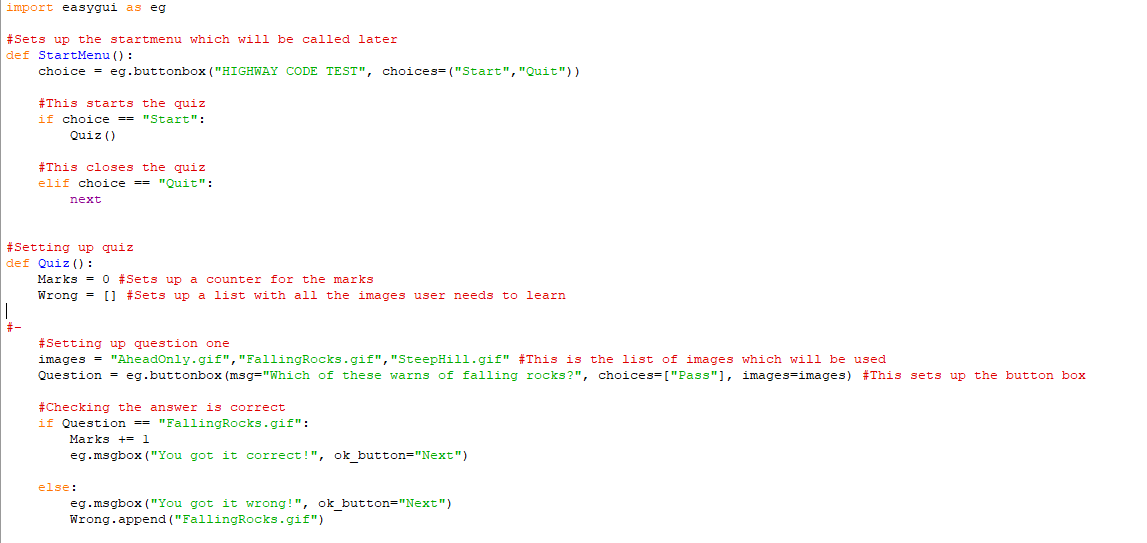
## Testing

The solution should be tested using the following processes:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test series 1** | | | |
| **Test no.** | **Purpose** | **Data** | **Expected result** |
| 1 | Test question 1 with incorrect answer. | Ahead only button clicked | Marked incorrect, score doesn’t go up. |
| 2 | Test correct answer for question 1 | Falling rocks | Market as correct. Score goes up. |
| 3 | Check counting of total when questions correct. | 1. Falling rocks 2. No stopping 3. Side winds 4. No motor vehicles 5. No through road | Final score result is 5. |
| 4 | Check counting of total when some questions incorrect. | 1. Falling rocks 2. Ahead only 3. Side winds 4. Cycle route 5. No through road | Final score result is 3. |

# Development

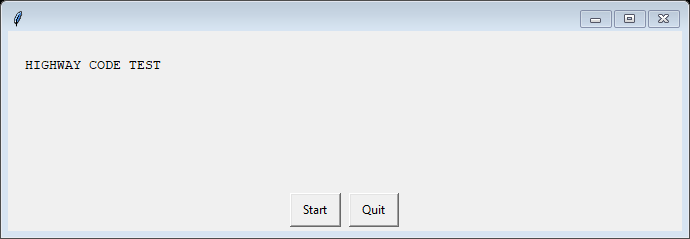
*[Paste screenshots of your code here. Explain how key parts work.]*

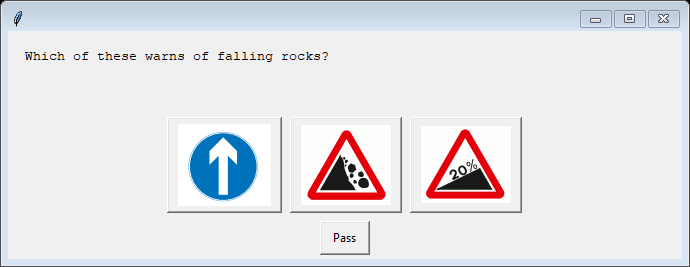
**

*Note:*

*Setting up question one is the same as setting up the rest of the questions*

*Also I didn’t finish up the final screen*





# Evaluation

## Test Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Test series 1** | | | |
| **Test no.** | **Purpose** | **Expected result** | **Actual result** |
| 1 | Test question 1 with incorrect answer. | Marked incorrect, score doesn’t go up. | Marked Incorrect  Score doesn’t go up |
| 2 | Test correct answer for question 1 | Market as correct. Score goes up. | Marked correct  Score goes up |
| 3 | Check counting of total when questions correct. | Final score result is 5. | Scored didn’t print correctly |
| 4 | Check counting of total when some questions incorrect. | Final score result is 3. | Score didn’t print correctly |
| **Test series 2** | | | |
| **Test no.** | **Purpose** | **Expected result** | **Actual result** |
| 1 | Check counting of total when questions correct. | Final score result is 5. | Score printed correctly |
| 2 | Check counting of total when some questions incorrect. | Final score result is 3. | Score printed correctly |

## Evaluation

*[Review each of the criteria below. Did you meet them? If so, how? If not, why not?]*

1. *A menu should display to let the user begin the quiz, or quit.*
2. *5 questions should display, in order.*
   1. *The user should be presented with a road instruction, and three signs. One of the signs should match the instruction, the other two should not.*
   2. *If the user selects the correct sign, they should be told that they are correct, and their score should increase by 1.*
   3. *If the user selects an incorrect sign, the correct sign should be highlighted. The score should not increase.*
3. *At the end of the quiz, the user should be presented with their final score, and images of any signs they were not able to identify correctly.*

The menu function worked as intended

The texted isn’t centred but the buttons themselves all worked as intended.

I also managed to get all the questions done. I didn’t finish creating the end screen but I managed to do basic white box testing to check all the variables are being processed as intended. Marks went up for every correct question and all the questions that the user would be getting incorrect would be staying in a list which would have been put on the end screen.

1. <https://highwaycodetest.co.uk/> [↑](#footnote-ref-1)