**Saint Augustine’s College, Sydney**

**Software Engineering Year 11: Programming Fundamentals**

Planning

## Task Definition

I have been assigned the task of developing a quiz. A Python application for making a simple but effective quiz. Mainly used for pub trivia as it ensures to have a general knowledge and sport section and for bonus category there is ‘other’ it is very difficult and could be used for bonus points. I aimed for a bug-free application with an engaging print for the terminal and relative questions per topic.

The functional requirements I will need to implement include:

Randomly select questions from a dictionary when each topic is chosen.

Provide an engaging effect for the terminal.

Get players guesses for answers and determine if they are right or wrong

Validate guesses check if they are proper ones or just blank.

Get score for correct questions and use it for a leaderboard.

Allow hints for 1 section and relative hints for the question.

## Structure Chart

As I will be taking a functional approach to the development of my quiz, it is appropriate to create a structure chart that will decompose the game logic into a mainline and the individual functions within, and help visualise the data/parameters that will be passed around.

The following flowchart maps out the functions within my program. It was made using draw.io

A diagram of a diagram

AI-generated content may be incorrect.

## Flowchart

A screenshot of a computer

AI-generated content may be incorrect.This algorithm's logic can be effectively illustrated through a flowchart to visually augment comprehension. While the detailed operations of the subfunctions are simplified, this overview should adequately convey the workings of the quiz.

## Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Data Type | Format for display | Description | Example | Validation |
| questions | List[Dictionary] | List of dictionaries | All of the questions, answers and hints for the quiz | Questions = [  {“Question”: “What is…”  ] | - |
| score | integer | Numeric | Tracks the player score during the quiz | score = 7 | Must be positive |
| hints | integer | Numeric | Tracks the number of hints during the ‘Other’ category of the quiz | hints = 1 | Can’t be below 0 |
| category | string | Text | Stores the category selected by the player, used for the leaderboard | category = “sport” | Must be a category that was avaible |
| hint\_loop | bool | boolean | Is a loop to check if the player used a hint, then allows them to answer quesiton | hint\_loop = False | Must be true or false |
| valid\_answer | bool | boolean | Is a loop to check if the player answered the question and not left it blank. | valid\_answer = False | Must be true or false |

# Development/GitHub Repository

My GitHub Assessment Page - <https://github.com/ollienunn/Quiz-Assessment-/tree/main>

# Project Reflection

The planning phase of the quiz was left last as I was naive to properly plan out the quiz and thoroughly think about the functions and what I would want and need to make this a fully functional quiz. Since planning was left to the end, I struggled with making both the flowchart and structure chart, but managed. I insisted on coding the quiz first as I thought it would make doing the charts easier (it didn’t). I did regret this decision and for future projects will not do this and plan before hand to get an idea and make the charts easier to make and understand.

Initially, I encountered difficulties with the topic selection and getting the different questions per topic which is why my code is not neat however, to me it seems neat and works as I want and intended just struggled with the logic. I also struggled with implementing the leaderboard function, I had a brief understanding of the concept and was happy with writing in the new file, yet when retrieving the scores I had to reley on co-pilot to make the code. I somewhat understand it but it does use code which doesn’t make sense to me despite co-pilot explaining it.

The GitHub repository management proved to be very useful as when I made a big change that resulted in lots of errors I was able to go back and abandon the idea will still having a working project. It also helped with making sure the code I added was actually useful and not just filler since every little change needed to be synced and made into a repository.

Overall I reckon I improved with making code that relates to a specific topic and was able to use tools that otherwise I wouldn’t touch like the repository which was a watchful eye and the charts which was a pain but made the code simple to understand and helps breakdown the schematics of my quiz and help me understand what I made. This planner was also very helpful and I was able to use the proper structure on how to document and how to describe each section of the documentation, which without I wouldn’t have been this neat or structurally proper.