Knowledge required

in addition the the knowledge of previous classes, the following is required:

- `class`es
- JSON files and parsing
- some understanding of `pydantic` for the extra features
- some understanding of exceptions
- maybe requests for the extra features

packages required

Most of the packages used for this exercise are build it, and not all of those have to be used.

- required packages:
  - `argparse`
  - `random`
  - `json`
- extra packages:
  - `pydantic`

Things to go over in the lecture

- classes and objects
- classes vs data structures (records)
- exceptions
- sets
- JSON files
- extra
  - `pydantic` for data validation and parsing - extra

The exercise

It is recommended to first Implement the core functionality of the game, then do the full requirements and then add the extra features.

Core functionality

The basic idea of the is to have a trivia questions game. If we have 2 players playing then the game would go like this:

1. The game selects a question that is unknown to both of the players, and then prints the question with all the possible answers (each answer with a number).
2. the players take turns answering the question.
   - If the answer is correct, the player gets a point.
   - If the answer is wrong, the game prints a message saying that the answer is wrong.
3. in either case the turn goes to the next player. If the previous player answered correctly, the next player gets a new question, if not, he gets the same question, with the same answers.

So, if a question was not answered correctly, the next player gets the same question, with the same answers, and has a greater chance of answering correctly.

Full requirements

- The game should be activated from the command line (using `argparse`, or something similar)
    - The parameters should be path to the file with the word list
    - number of players
- The computer selects a question from the file, and the players take turns answering the question as described in the Core functionality section.
    - If the current player answers correctly he gets a point, and the next player gets a new question.
    - if the current player answers incorrectly, the next player gets the same question, with the same answers.
- When there are no more questions, the game is finished, and should print the winner and the full scores.
- If I play the game twice with the same file
    - The order of the questions should be different each time
    - The order if the answers in each question should be different each time
        - (each game, not each time the qestion is asked)

Extra features

- allow categories and difficulty levels (and allow to choose)
    - So each question has a category (you decide them) and a difficulty level (easy, medium, hard)
    - On each turn, before a new question is asked, the player can choose a category and a difficulty level from the ones sill available
    - The game should then select a question from the selected category and difficulty level (if there are more than one question in that category and difficulty level, the program chooses one of them at random.)
    - The computer should not allow to choose a category and difficulty level that are not available.
- Get the question from a web API instead of a file. You can use