Main individual Assignment: The Question Bank Project

Assignment brief

Handed out	Monday 11th March 2024
Hand-in	Wednesday 8th May 2024, 1 pm via Blackboard upload *(Assessment and Feedback > Main individual assignment > The Question Bank Project submission point)*
Feedback	Friday 31st May 2024
Percentage of overall module marks:	**50%**

Problem statement

A local school requires a prototype quiz program to help its students evaluate their understanding and revise for exams. This prototype must be text based with a textual menu to:

1. enable teachers to build a bank of questions with answers for a specific module,

2. enable students to take a quiz associated with a module, where a selection of questions are presented in a random order.

Your Task

Question bank functional requirements

A question bank is associated with just one module, but one module can have many question banks.

A teacher manages question banks as follows:

FR1[1]: A teacher can create a new empty question bank and associate it with one module identifier. Each question bank has a **question bank unique identifier** of the form ***module-identifier:bank-identifier*** where bank identifiers must be unique for a specific module. The format of the identifiers (both module and question bank) is up to the teacher, that is, it can be any text such as a number or a word or even a phrase. However, the maximum length of module and bank identifiers should be 7 and 15.

FR2: A teacher can add new questions to an existing question bank (also see **Further information on FR2** below):

> **FR2a**: Single choice: select one answer from ***N*** options. Here, ***N*** is between 1 and 10. For the single-choice question type, the teacher:
> - enters the question as text,
> - provides each question answer as text, up to ***N*** answers,

[1] FR stands for functional requirement.

- indicates which of the answers is correct; there must only be one right answer.

FR2b: Fill the blanks. For the fill-the-blanks question type, the teacher:

- enters a single text statement, but within that text indicates where the blank words are missing. Note that the blanks can represent one or more words; it is up to you to decide how this is done,
- enters the correct words for the blanks; again, it is up to you to decide how this is done.

FR3: A teacher can list and then remove questions from an existing bank.

FR4: A teacher can list all the question banks for a specific module.

FR5: A teacher can delete a question bank, but only when it is empty.

Quiz functional requirements

FR6: Take a quiz. A student can select quiz. When they take the quiz, they can:

FR6a: List the question banks for a specific module identifier: if the module identifier is unknown then an error message should be displayed.

FR6b: Take a specific question bank quiz using its question bank unique identifier (see **FR1**). This will ask the student for the number of questions to display (call this number ***Q***). If this is greater than the number of questions in the bank, then it will be set to the question bank size.

A running question bank has the following requirements:

FR7: A student may end the question session at any time, at which point some statistics are displayed:

- A score, representing the number or percentage of correct answers out of the total questions in the question bank.
- The number of questions not answered.
- The elapsed time between the start of the quiz and the end.

FR8: The quiz displays ***Q*** questions (see **FR6b**) in a random order. It indicates the number of questions to be displayed. Only one question at a time is displayed and the question number is clearly indicated.

FR9: The student decides when they want to move to the next question or go back to a previous question.

FR10: A single choice question is displayed to the student in an appropriate textual format allowing the student to choose an answer depending on the question type. For a fill-the-blanks question the student adds the missing words or phrases for the blanks. The form this takes is up to you.

Main individual Assignment: The Question Bank Project

Non-functional requirements

NFR1[2]: The question bank must be persistent and stored in a text-based database: i.e. text files. You may decide on the format that you wish to use. You must not use a relational database nor any binary format, i.e. it must be possible to view the files with a text editor.

NFR2: A text-based, menu-driven user interface is required.

NFR3: It must be possible to plug in new kinds of question in the future, hence the need for inheritance.

** For this prototype there are no security requirements.*

Steps

1. **Analyse the problem**:
    a. Read the above **Problem statement** carefully, identifying and listing potential classes and their objects. These are often found by looking for nouns (or noun-based phrases) in the problem statement (although some of these may be attributes of classes).
    b. Identify functional requirements, e.g. the task points above. Based on these functional requirements **draw a use case diagram**.

2. **Draw a class diagram**. This must be drawn with software tool of your choice. **It must not be auto-generated from existing code**. Identify cases where inheritance will help reduce code duplication (hint: there will be things in common to the different question types).
3. Create outline classes in IntelliJ (or some IDE of your choice) and incrementally add code. You will probably want to start with implementing the use case diagram as a command line menu (or handling command line arguments), perhaps within an `Application` class. Every now and then look for code duplication. If you find duplication, then try to extract this duplicated code into a private method that is called from several places.
4. **Write a test table** (see the template example provided on Blackboard with this assignment). Test as you proceed. Use the visual debugger. You will probably add further tests as you implement but start with a set of tests based on the functional requirements that you have identified. Note that even if you fail to implement a requirement, I still expect you to attempt to write a test for that requirement in the test table.

Hand-in instructions

Upload a single zip file (NOT rar or anything else: a 2% penalty will be applied if the wrong format is used) to Blackboard that contains:

1. Your IntelliJ project (everything in the project).
    a. If you are not using IntelliJ then package up the source code files, class files and any data files.

[2] NFR stands for non-functional requirement.

b. I will run your program from IntelliJ.

2. A pdf (a 2% penalty will be applied if the wrong format is used) 2000-word (+/- 10%) document that contains:
   a. A **front cover page** that includes your name (or student reference if you prefer), email, document date, and the title of this assignment.
   b. **Page 2 (two) must provide a contents page** (section titles and their associated page numbers). See textbooks for examples. Use Word or a similar tool to generate this for you.
   c. **Page 3 (three) must start with an introduction** that says what is to follow. This **must** also add a sentence at the end saying how far you got with the assignment. This **must** also contain a **UML use-case diagram** that presents the functional requirements.
   d. Have a section called ***Design*** that has a **class diagram**, and a **textual description and rationale for each class and its relationships**. I will be looking for well-designed classes, method names and attribute names. **Include a pseudo-code example of the most complex algorithm.** Justify your choice.
   e. Have a section called ***Testing.*** This must have a test table as described in class (see example provided with this assignment). This section must also contain screenshots of your program running; a screenshot for every requirement implemented with caption text saying what each of them shows. It must also have a discussion of test data used and the results of running the tests: e.g. why a particular data input value was used, such as letters where numbers were expected. **Show your program running from the command-line: I want to see a command similar to the following example:**

      java -cp out/production/project RoladexApplication
      (see Lecture 9 on Blackboard).

   f. Have a section called ***Evaluation***. Describe how you went about solving the assignment. What flair, if any, was attempted? What was difficult, what remains to be done, what did you learn, and what mark do you think you should be awarded and why. Also, include a brief description of any creativity and innovations you have done in your implementation. **Have a subsection for each of the above.**

IMPORTANT: You should name the submitted zip file as **YourEmailID-individual-main-assignment-2023-24.zip** (example: **far8-indiviual-main-assignment-2023-24.zip**). Otherwise, 2% penalty will be applied if the wrong name of the zip file is provided.

Note: this is an "individual" assignment and must be completed as a one-person effort by the student submitting the work. This assignment is **NOT** marked anonymously.

PLEASE NOTE: By submitting your work for this worksheet using Blackboard, you are making the following declaration:
"This submission is my own work, except where clearly indicated.

Main individual Assignment: The Question Bank Project

I understand that there are severe penalties for plagiarism and other unfair practice, which can lead to loss of marks or even the withholding of a degree. I have read the sections on unfair practice in the Students' Examinations Handbook and the relevant sections of the current Student Handbook of the Department of Computer Science. I understand and agree to abide by the University's regulations governing these issues."

See the guidelines in the Student Handbook regarding Unacceptable Academic Practice. [1]

Marking scheme

Assessment will be based on the assessment criteria described in Appendix AA of the Student Handbook [2].

However, the following table gives you some indication of the weights associated with individual parts of the assignment. This will help you judge how much time to spend on each part. Note that I will award full marks for each category below if you fully complete the requested feature and to a high standard. However, the *Creativity & innovation* marks represent work that goes beyond the requested features.

Quality of the documentation: design as described, diagrams, grammar, spelling, etc.	Does the documentation follow instructions in points 2(*a)* to *2(d)* above?	20%
Implementation: Question bank management	This covers FR1, FR3, FR4, FR5	5%
Implementation: Creating new questions	This covers FR2: FR2a, FR2b	5%
Implementation: Running the quiz	This covers FR6, FR7, FR8, FR9, FR10	5%
Implementation: Persistence and UI	This covers NFR1, NFR2, NFR3	5%
General implementation and design	Other functions required are implemented correctly. Is the code of high quality, e.g. object oriented and following principles such as high cohesion and low coupling, good identifier names, indentation, small methods, commenting, error checking and exceptions, reading and writing a file, use of ArrayLists (or similar), packages?	20%
Implementation: use of inheritance	Has inheritance been used effectively?	10%
Testing	Does the documentation follow the instructions in point 2(*e)* above?	15%
Evaluation	Does the documentation follow instructions in point 2(*f)* above?	5%

Creativity & innovation	I am looking for applications that show flair, creativity or innovation. Note that it is important that the basic requirements are met before attempting any flair. This addresses the Appendix AA 80% to 100% assessment criteria that state: *"and will more than completely fulfil the functional requirements."* A good possibility here is to provide a graphical user interface for controlling the program: user controls. Other possibilities include: to add further question kinds; storing student results to monitor progress. Ask me if you are not sure.	10%

References

[1] Computer Science Department Student Handbook Undergraduate Programmes 2021/22 (Online) https://impacs-inter.dcs.aber.ac.uk/en/cs-undergraduate/official-information/student-handbooks (Accessed 6th March 2024)

[2] Appendix AA. Assessment Criteria for Development (Online) https://impacs-inter.dcs.aber.ac.uk/images/editor-content/Documentation/Handbooks/Appendices/AppendixAA.pdf (Accessed 6th March 2024)