Perfect Number [55 points]

A perfect number is a positive integer that is equal to the sum of its positive divisors, excluding the number itself. In other words, a perfect number is one where the sum of its divisors (excluding itself) equals the number itself.

For example: the number 6 has divisors 1, 2, and 3 (excluding itself), and (1 + 2 + 3 = 6), making it a perfect number. The next perfect number is 28, since (1 + 2 + 4 + 7 + 14 = 28).

Create a GUI that shall take two numbers, one is the num and the other is the number of threads cnt. Your goal is to show all the perfect numbers from 1 to num using cnt threads. You should also have a way to handle errors and showing the appropriate error message in the GUI.

Next is you should create Threads by unequally dividing the tasks to the threads. Divide them unequally to highlight the differences in progress of individual threads. In order to visualize the progress of the threads, dynamically create the same number of ProgressIndicators as the number of threads in your GUI and use their setProgress() method that will indicate how close each threads are to finishing the task. The setProgress() method will take a double from 0 to 1, where 1 means 100% done.

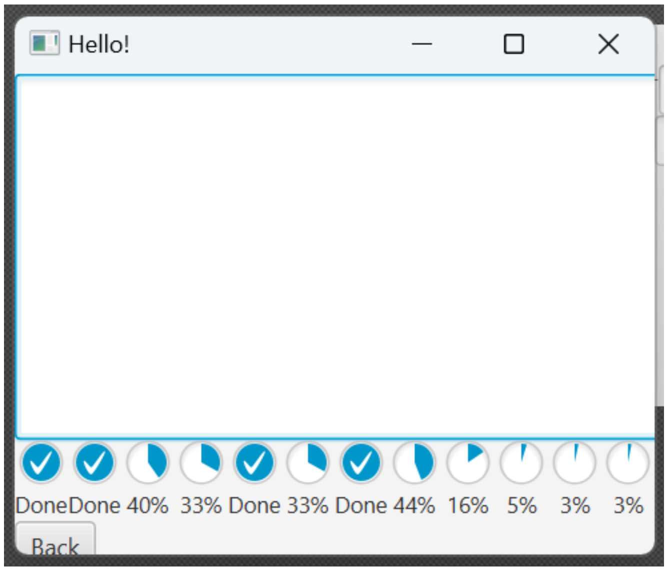
When all threads are done, aside from showing the perfect numbers (preferably in GUI), also output how many are less than perfect (i.e., the sum of factors is less than itself) and more than perfect (i.e. the sum of factors is greater than itself).

In creating GUI, you can use either FXML or coding.

Note: You will still get 25 points if you only implemented the threads, without the GUI.

Note: This may be prone to errors as JavaFX and other threads may not work well together.

Sample Output (you may add more elements):

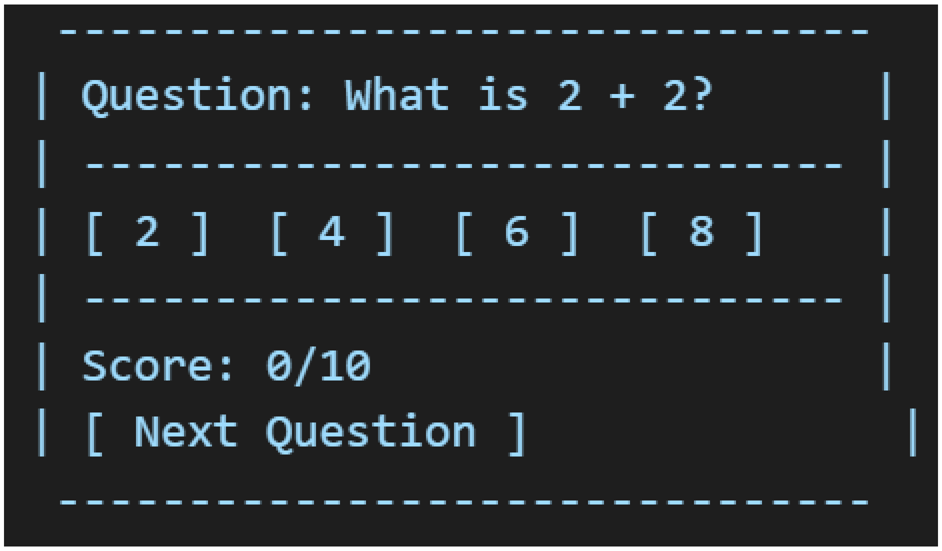


Quiz App [55 points]

Create a JavaFX-based application that allows users to take a multiple-choice quiz, track their scores, and view results.

Requirements:

1. **GUI Layout:**
   - A label to display the current question.
   - Four buttons for multiple-choice answer options.
   - A "Next" button to move to the next question.
   - A label to display the current score.
   - A result screen at the end of the quiz showing the total score and percentage.
2. **Functionality:**
   - The application should present questions one at a time with four answer choices.
   - These questions and choices may be hard-coded. Make sure to have at least 5.
   - Clicking an answer should highlight the selected choice.
   - Clicking “Submit” updates the scores and disables submit for this question.
   - Clicking "Next" should move to the next question.
   - The application should track the number of correct answers.
   - At the end of the quiz, the user should see their final score and percentage.
   - Proper error handling for cases such as no answer selected.
3. Example UI Layout:



```
 --------------------------------
| Question: What is 2 + 2?      |
| ------------------------------ |
| [ 2 ]  [ 4 ]  [ 6 ]  [ 8 ]    |
| ------------------------------ |
| Score: 0/10                   |
| [ Next Question ]              |
 --------------------------------
```

Bonus Tasks:

- Dark/Light Mode
- Ability to add questions
- More!

Smarter BruteForce [70 points]

Given a password, use threads to brute-force your way into guessing the password.

Perform this by having n*5 threads, where n is the length of the password.

This technique involves the use of vowels wherein each thread will focus on a vowel and a position. For example, the password "gwapo", having 5 as its length, it shall create 5*5=25 threads to perform the following:

Thread 01: _ _ _ _ a

Thread 02: _ _ _ a _

Thread 03: _ _ a _ _

Thread 04: _ a _ _ _

Thread 05: a _ _ _ _

Thread 06: _ _ _ _ e

Thread 07: _ _ _ e _

...

Thread 10: e _ _ _ _

...

Thread 24: _ u _ _ _

Thread 25: u _ _ _ _

Hence, there will be two threads that can possibly find "gwapo" -- the __a__ and ____o. In your output as you print your attempts, print it similarly as the one above, having the thread number before the attempt. In a similar way as the laboratory activity, stop the other threads if it has been found by one thread.

With this technique, we may not be able to have an attempt for the words without vowels -- to which this problem can live with.