Assignment 5:

1. Learn about Ruby.
   Sources: http://rubylearning.com/satishtalim/tutorial.html
   http://ruby-for-beginners.rubymonstas.org/index.html

2. Complete these problems after learning about ruby:

1. Let's introduce ourselves

- Print out "Welcome to the Ruby Arithmetic Tutor

Hint: Look up the #print, #puts and #p statements. See which is the most appropriate here

2. Let the user introduce himself

- Ask "What is your name?"
- Print out "Hello, <name>"

e.g.

What is your name?

John

Hello, John

Hint: Look up #gets and string interpolation

3. Breathing space

- "Press enter to begin the quiz"

Hint: #gets

4. Let's ask a question

- "What is 2 + 2?"
- Wait for the answer
- If correct, print "Correct!"
- If wrong, print "Wrong - the answer was 4"

Hint: #puts, #gets, #if, #==

5. Five questions

- Repeat, asking four more questions

Hint: copy-and-paste

6. Keeping score

- Add a score: "You got <n> questions right out of 5"
- Display the score as a percentage

e.g.
You got 3 questions right out of 5.
Your score was 60%

Hint: variables, arithmetic

7. Extract a function

- Write a function, ask, that takes two string arguments, a question and an answer
- ask(question, answer) should
    - print out question
    - get an answer from the user
    - compare it to answer
    - if correct, print "Correct!"
    - if wrong, print "Wrong! The answer was $answer"
    - return true or false, depending on whether the answer was right or wrong

Hint: #def, #return

8. Use the function

- Rewrite your program from Q6 to use the function from Q7. Use the return value of the function to keep score

9. Numeric loop

- Rewrite to use a loop:
    - Put 5 questions in an array, questions[]
    - Put the corresponding 5 answers in an array, answers[]
    - Call ask(questions[i], answers[i]) for each i from 0 to 4
    - Print out the final score

Hint: arrays, #times, blocks

10. Change the data structure

- Rather than questions = [q0, q1, q2, q3, q4] and answers=[a0, a1, a2, a3, a4], use a single array, questions = [[q0, a0], [q1, a1]...]

Hint: questions[i] is now an array, [qi, ai]

11. Internal iteration

- Use #each rather than #times to iterate over the array of questions

Hint: #each

12. Classes

- Write a Question class, with member variables @question and @answer
- Have a constructor, Question.new(question, answer)
- Have accessors to read the question and the answer:
  - a = Question.new("What is 2 + 2?", "4")
  - a.question #=> "What is 2 + 2?"
  - a.answer #=> "4"

Hint: #class, #initialize, #attr_reader

13. Rewrite the ‘ask’ function from forQ7 to take a single Question object as a parameter

14. Rewrite the quiz to construct and then iterate over an array of 5 Question object

15. Make ask a method of the Question class

- rather than ask(question), we want question.ask

Hint: methods

16. Write a Question.make_addition class method that takes three numbers as arguments and returns an addition question

- Question.make_addition(3, 4, 7) should do the same thing asQuestion.new("What is 3 + 4?", "7")

Hint: String interpolation, #to_s

17. Make ruby calculate the answer for us

- Revise your make_addition method from Q16 to take only two arguments, and have it calculate the answer itself
- Question.make_addition(3, 4) should do the same thing as Question.new(“What is 3 + 4?”, “7”)

18. Validate the inputs to make_addition

- Check that both inputs are integers
- Raise an exception if they are not

Hint: #raise, exceptions

19. Add a make_random_addition method

- Same as make_addition, but takes no arguments, generating two random numbers instead.

Hint: #rand, have make_random_addition call make_addition rather than duplicating its code

20. Rewrite the quiz to initialize an array with five random questions instead

21. Rewrite the quiz to do away with the array altogether, generating questions as needed

22. Add a Question.make_subtraction method

- Question.make_subtraction(10, 4) = Question.new("What is 10 - 4", "6")
- Validate that the both inputs are positive integers
- Validate that the answer is not negative

23. Add a Question.make_random_subtraction method

- Call make_subtraction with two random integers
- Rescue the exception if the answer is negative
- Retry with the arguments in the other order

Hint: #begin/#rescue/#end

24. Modify your quiz to take two command line arguments, the first being either “add” or “subtract” and the second being how many questions to ask

e.g. to ask ten addition questions, run

```
$ ruby quiz.rb add 10
```

Hint: ARGV, #to_i

25. Validate the command line arguments. If they are missing or wrong, print out a usage example and exit.

e.g.

```
$ ruby quiz.rb 4

Usage: ruby quiz.rb <add|subtract> <number of questions>
```

3. After completing above problems, start learning Ruby on Rails.

Source: http://guides.rubyonrails.org/getting_started.html
https://www.railstutorial.org/book/beginning

4. After learning about Rails, create a new application that has following functionalities. This application will have two models: Users, Books where a user can have many books.

A. Functionality for user registration and login. (Hint: Use devise gem.)
B. Create a page for displaying all the books.
C. Functionality for adding, deleting and editing books, only when user is logged in.
D. User can upload a PDF file for the books and give functionality to download books. (Hint: Use paperclip, carrierwave gem)
E. Functionality to filter the books by the user who uploaded them and their upload date. Also, do this by using AJAX.
F. Functionality for sending emails when a book is added to the user who added it. (Hint: Use action mailer)

After completing this task, please add all your code on Github.