1. Variables, Arrays, and Loops
    a. Create a program that asks the user for their favorite 5 foods. Then display those foods as an array, using the “p” keyword.
    b. Now, instead of printing out the array, output 5 separate lines of each food, with the prefix, “I love”. For example

```
I love cobb salad
I love spaghetti
I love pizza
I love Swedish meatballs
I love mac and cheese
```

    c. Finally, change your program so that when the list is printed in the terminal, each food is prefaced with a number, beginning with 1 and going up to 5, for example:

```
1. cobb salad
2. spaghetti
3. pizza
4. Swedish meatballs
5. mac and cheese
```

2. Operators and Conditionals
    a. Create and define a variable ‘count = 0’. Using a loop and the ‘+=’ operator, output the following:

```
0
1
2
3
4
5
6
7
8
9
10
```

    b. In plain English, describe to your group (Just talk it out):

```
if answer != 5
```

    c. Imagine in your code, you have this line:

```
if result != true
```

    and you replace it with this line:

```
unless result == true
```

    will the code run the same as before? Yes or No?
    d. Look at the conditional below. What value(s) can the variable ‘result’ be for it to output “HELLO!” What values will it not output? Experiment in Sublime:

```
unless result
  puts "HELLO!"
end
```

    e. Translate the following into ruby code. Run the program to make sure it works:

If Sam can cook more than 10 recipes and Sally speaks more than 5 languages, they should date. If Sam can make crepes or Sally can speak French, they should marry.

3. Hashes
    a. Create a banking program that asks the user 5 times to enter a first name, last name, and email. This information should be stored as an array of hashes.
    b. Each person should automatically be given an account number which is a randomized ten digit number.
    c. After the user is finished, the program should print out all the users, for example:

```
FIRST NAME: Joan
LAST NAME: Kelson
EMAIL: jkelson@juno.com
ACCT #: 2372041038
```

    and so on for all the users.

4. Final Review
    a. Create a program that puts your class into groups! Have the program request the user to enter each student's name. Assume the classroom has an even number of students, so there are only groups of two. For example, you can have the program output groups like so:

```
Group: Hermione Seamus
Group: Lucius Cho
Group: Sirius Luna
Group: Severus Draco
```

    b. Refactor the program to take an odd or even number of students. If odd, one group should have three students.

```
Group: Hermione Seamus
Group: Lucius Cho
Group: Sirius Luna
Group: Severus Draco Harry
```

5. ****Bonus*****
    a. Continue with Exercise 3: Expand the program so a user gets all the account info by entering an account number i.e. prompt the user to enter an account number, and output the first_name, last_name, etc.
    b. Also continue with exercise 3: reject a user entering an account if the email address doesn't have an "@" and doesn't end in ".com" Prompt the user to try again if this happens.