Instructions

Write one method per task (all of these methods should go in the same class). **If a name for a method is specified, make sure it has that name!** Otherwise, it won't work when you try to call it later.

Before writing a method, it is recommended to FIRST write an algorithm or make a flowchart depicting the high-level operations being performed. **For methods with arguments, make sure to test out your methods on a variety of inputs to ensure they are working correctly!**

Style

These style guidelines should be followed at all times:

- Source code file names and class names should start with an uppercase (capital) letter.
- Source code file names and class names should be relevant to the function of the code.
- Variable names should start with a lowercase letter (unless they are program-wide static final constants, in which case they should be written in all caps).
- Variable names should reflect their purpose, for readability.
- Every time you open a block of code with a {, all of the code inside the block (i.e., everything up until the corresponding }) should be indented. The amount to indent your code by is up to personal preference (I like 2-4 spaces), but you should keep it consistent throughout a program.
- Use line breaks (newlines, as in what you get when you hit the 'Enter'/'Return' key on your keyboard) appropriately to make your code more readable.
- Write a block comment (that's the one that starts with /* and ends with */) at the start of your program containing a brief description of what the program does, along with your name or hacker name.
- Write comments throughout your code whenever you do anything complicated or particularly interesting. A good rule of thumb is to comment anything you don't think you'd be able to immediately understand, if you looked at your code 2 months from now.

Part A: Static, No Return Type, No Arguments

1. Write a method ‘public static void printGreeting()’ that prints out a friendly greeting.
2. Write a method ‘public static void singSong()’ that uses a loop to print out the lyrics to your favorite song chorus 5 times (the chorus is the part that gets repeated in a song several times, usually the catchy part).
3. Write a method ‘public static void spamWithGreetings()’ that uses a loop to call your printGreeting() method 10 times.
4. Write a method ‘public static void getAndCommentOnFavoriteColor()’ which initializes a Scanner, asks the user what their favorite color is, reads a line of text from the user, then makes a comment on the line of text based on what color it contains. Your method should have responses for at least 3 different colors. If the input doesn’t contain one of the colors you’re looking for, then print something like “sorry, I don’t know that color.”.

Part B: Static, With Arguments

1. Write a method ‘public static void printTwice(int x)’ that takes in an integer x as an argument, then prints out the number twice.
2. Write a method ‘public static void printNGreetings(int n)’ that takes in an integer n as an argument, then uses a loop to call your printGreeting() method n times. For example, if n is 12, then printGreeting() should be called 12 times.
3. Write a method ‘public static void bottles(int n, String beverage)’ that takes in an integer n and a String as an argument, then prints out the appropriate verse of the “99 bottles of <beverage>” song. For example, if the arguments are 5 and “soda”, then the method should print ONLY:

   ```
   5 bottles of soda on the wall, 5 bottles of soda.
   Knock one down, pass it around, 4 bottles of soda on the wall.
   ```

   (You may assume the entered number is 3 or more. For a bonus, you can add extra logic to handle 2, 1, and lower numbers.)

Part C: Static, Return Type

1. Write a method ‘public static int add3(int x)’ that takes in an integer x and returns that number plus 3.
2. Write a method ‘public static int superSubtraction(int x, int y)’ that takes in two integers, x and y, and returns x + x - y.
3. Write a method ‘public static boolean saidYes(String answer)’ that takes in a String, answer, and returns true or false. The method should return true if answer matches exactly `"yes"`, `"yeah"`, or `"affirmative"` with any mix of capitalization, and false otherwise.
4. Write a method ‘public static String substituteTN(String oldString)’ that takes in a String, oldString, and returns a copy of that String where all the ‘t’ characters have been replaced with ‘n’ characters.