CODE FOR 1,3,4,5,6, & 7 MAY BE FOUND ON OUR COURSE GITHUB.
1. Acme Payroll System

Acme Parts runs a small factory and employs workers who are paid one of three hourly rates depending on their shift: first shift, $17 per hour; second shift, $18.50 per hour; third shift, $22 per hour. Each factory worker might work any number of hours per week; any hours greater than 40 are paid at one and one-half times the usual rate. In addition, second- and third-shift workers can elect to participate in the retirement plan for which 3% of the worker's gross pay is deducted from the paychecks. Write a program that prompts the user for hours worked and shift, and, if the shift is 2 or 3, whether the worker elects the retirement. Display: (1) the hours worked, (2) the shift, (3) the hourly pay rate, (4) the regular pay, (5) overtime pay, (6) the total of regular and overtime pay, and (7) the retirement deduction, if any, and (8) the net pay. Save the file as **AcmePay.java**

Starter Code is provided in the GitHub repo.

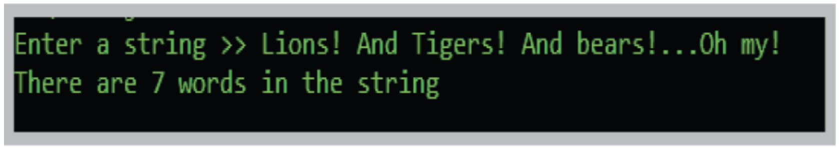
YOU MUST USE THE VARIABLE NAMES PROVIDED – DO NOT CHANGE THEM

You must complete the following:

- Ask for hours worked
- Ask for participation in retirement plan if the second or third shift are selected
- Fill in the details for the following methods/functions:
    - payRate
    - hoursBreakdown
    - grossPay
    - retirementPay

2. Word Count

Write an application that counts the words in a String entered by a user.
Words are separated by any combination of spaces, periods, commas, semicolons, question marks, exclamation points, or dashes. Save the file as **CountWords.java. Your output should look as follows:**



```
Enter a string >> Lions! And Tigers! And bears!...Oh my!
There are 7 words in the string
```

No starter code provided.

3. Rock.java

Program **Rock.java** contains a skeleton for the game Rock, Paper, Scissors. Add statements to the program as indicated by the comments so that the program asks the user to enter a play, generates a random play for the computer, compares them and announces the winner (and why). For example, one run of your program might look like this:

```
Enter your play: R, P, or S
r
Computer play is S
Rock crushes scissors, you win!
```

Note that the user should be able to enter either upper or lower case r, p, and s.
The user's play is stored as a string to make it easy to convert whatever is entered to upper case. Use a switch statement to convert the randomly generated integer for the computer's play to a string (the source code provided currently is using if statements)

Make the following improvements to the game:

- Allow the user to enter a string (rock, paper, or scissors).

- To allow for player misspellings, accept the player's entry as long as the first two letters are correct. (In other words, if a player types scixxrs, you will accept it as scissors because at least the first two letters are correct.)

- When the player does not type at least the first two letters of the choice correctly, reprompt the player and continue to do so until the player's entry contains at least the first two letters of one of the options.

- Allow 10 complete rounds of the game. At the end, display counts of the number of times the player won, the number of times the computer won, and the number of tie games.

4. Pizza POS

In this lab, we will be editing a pizza ordering program. It creates a Pizza order to the specifications that the user desires. It walks the user through ordering, giving the user choices, which the program then uses to decide how to make the pizza and how much the cost of the pizza will be. The user will also receive a $2.00 owner discount if his/her name is Jack or Diane.

Task #1

1. Open the file PizzaOrder.java contained in the Chapter 3 Lab zip file.

2. Compile and run PizzaOrder.java. You will be able to make selections, but at this point, you will always get a Hand-tossed pizza at a base cost of $12.99 no matter what you select, but you will be able to choose only pepperoni topping, and it should add into the price correctly. You will also notice that the output does not look like money. So we need to edit PizzaOrder.java to complete the program so that it works correctly.

3. Construct a simple if statement. The condition will compare the String input by the user as his/her first name with the first names of the owners, **Jack and Diane**. Be sure that the comparison is not case sensitive.

4. If the user has either first name, set the discount flag to true. This will not affect the price at this point yet.

Task #2

1. Write an if-else-if statement that lets the computer choose which statements to execute by the user input size (10, 12, 14, or 16). For each option, the cost needs to be set to the appropriate amount.

2. The default else of the above if-else-if statement should print a statement that the user input was not one of the choices, so a 12 inch pizza will be made. It should also set the size to 12 and the cost to 12.99.

3. Compile, debug, and run. You should now be able to get correct output for size and price (it will still have Hand-tossed crust, the output won't look like money, and no discount will be applied yet). Run your program multiple times ordering a 10, 12, 14, 16, and 17 inch pizza.

Task #3

1. Write an if statement that uses the flag as the condition. Remember that the flag is a Boolean variable, therefore is true or false. It does not have to be compared to anything.

2. The body of the if statement should contain two statements:

    a. A statement that prints a message indicating that the user is eligible for a $2.00 discount.
    b. A statement that reduces the variable cost by 2.

3. Compile, debug, and run. Test your program using the owners' names (both capitalized and not) as well as a different name. The discount should be correctly applied at this time.

Task #4

1. The NumberFormat class is used to format information so that it looks right when printed or displayed. Read up on how to use it so that the output is correctly displayed.

5. GUI Grades

A Graphical User Interface, GUI (GOOO-EEEE), is an interface through which a user interacts with visual elements like icons or buttons (also called GUI objects) on the screen of an electronic device using a pointer, keyboard, or touch screen. Java has the capability of creating GUI programs by using Swing.

Swing in Java is **a Graphical User Interface (GUI) toolkit that includes the GUI components**. Swing provides a rich set of widgets and packages to make sophisticated GUI components for Java applications.

You will be creating a Grade class to convert grades entered by the user into other forms of the same grade. The results are shown in a GUI, as is the input.

GUIGrades.java is the main file and Grade.java is the object that we use in the GUIGrades.java. The majority of the code has been filled out. You must complete the following:

GUIGrades.java:

- Include string for origNumeric

Grades.java:

- Complete set methods for the other cases for each numeric grade
- Complete set methods for the other cases for each letter grade
- Complete the two get methods (Accessor method) for new numeric grade and letter grade

6. 3D Plots and Linear Regressions

Complete the Jupyter Notebook by creating 3D plots using the data set provided. All necessary dependencies are included. You must create a 3D scatter plot and then create the same 3D scatter plot with the line of best fit. If you need assistance either check in with me or review the following G4G resource:

https://www.geeksforgeeks.org/3d-scatter-plotting-in-python-using-matplotlib/

7. Taylor Polynomials

Complete the Jupyter Notebook by creating the Taylor Polynomial for cosine, exp(x) and ln(1+x). Pick four values for the degree to plot on the same graph as the analytic function.