1. Create a file sample.txt programmatically and write the following lines in that file programmatically.

   Line1:Programmatically created file.
   Line 2:These words were written programmatically.
   Line 3:All is Well. Be happy & enjoy the moment.

2. Write a Class to store the values in java.util.Properties (approx 5 Keys vs Values) object. Keys & Values can be provided from the runner class.After loading the properties with the provided keys & values, store the values in a text file (myprops.txt)

3. Write a Class to read the keys & values from the above created myprops.txt file(any file) & store those values back into the java.util.Properties object and print the properties.

4. Repeat tasks 1, 2, 3 by creating the files sample.txt, myprops.txt under /home/INC*/myDir. Please note that you should create the myDir directory programmatically.

5. Write a class with an additional constructor which accepts a “String” . Assign that String to an instance variable. Invoke this constructor from a separate runner class & print your object. This printing should print the object alone but the String variable should get printed automatically.

6. Write a POJO class with a String variable, integer variable. Have the necessary getter & setter methods for these variables with proper access specifiers & return types. Now define a constructor which accepts all these variables as an argument & set the incoming values to the appropriate variables. Now from a separate runner class, you should try to create an instance for your class using the above said constructor & print your object. You have to print the object alone in the Runner but it should print those String & Integer variables automatically.

7. Now from a separate runner class, you should try to create an instance for the above

POJO class using the default constructor.Using that ,call the appropriate setter methods to set the values for the String, integer. After that invoke the appropriate getter methods & print each value that is set previously.

8. Write a POJO class with a default constructor & an overloaded constructor (String & Integer args). It have a String variable & Integer variable, with proper getter & setter methods for these variables. Compile the class. Write a runner class with a different package than the POJO class. Hint:Refer reflection
    1. In the runner class, POJO class should not be imported explicitly or implicitly. i.e Runner class should get compiled without the POJO class.
    2. Invoke the default constructor of the POJO from the runner class
    3. Invoke the overloaded constructor of the POJO from the runner class
    4. Invoke any one getter method of the POJO from the runner class
    5. Invoke any one setter method of the POJO from the runner class

9. Define an Enum for the rainbow colors, with values(colorcode) ranging from 1 to 7. Print the color and its corresponding color code from the main method.. For eg: Color code of Violet is 1, Color code of Indigo is 2 etc. Invoke values() ordinal methods of the enum as well.

10. Write a Singleton class, so that there exists only one instance of the class in a JVM. Also read about how the Singleton pattern evolved in Java & what will be the best way to write a Singleton class.

11. Write a class & add methods to do the following. Refer: java.time package
    1. Return the currentTime with Date, seconds etc
    2. Return the currentTime in milliseconds. Using System class as well.
    3. Return the currentTime with Date,seconds in New york & London. Observer the difference.
    4. Return the week day for the currentTime in millis or any given time in millis
    5. Return the month(not numeric) of the currentTime in millis or any given time in millis
    6. Return the Year of the currentTime in millis or any given time in millis

12. Logger:

1. Replace all your System.out.println() in your previous tasks with appropriate Logger.log method. Make sure to do this in your future tasks as well.
2. Replace all your exp.getMessage() & exp.printStackTrace() with appropriate Logger.log method. Make sure to do this in your future tasks as well.