**Lab Assignment (3) on Exception Handling**

1. A special two-digit number is such that when the sum of its digits is added to the product of its digits, the result is equal to original number  
   For example: 59  
   5+9 = 14  
   5\*9 = 45  
   45+14 = 59 Therefore, 59 is a special two-digit number  
   Write a program in java that throws custom exception when number is not a special two-digit number.
2. Suppose selection of a particular exam include a fitness test which is conducted on ground. There will be a batch of 3 trainees, appearing for running test in track for 3 rounds. You need to record their oxygen level after every round. After trainee are finished with all rounds, calculate for each trainee his average oxygen level over the 3 rounds and select one with highest oxygen level as the most fit trainee. If more than one trainee attains the same highest average level, they all need to be selected. Display the most fit trainee (or trainees) and the highest average oxygen level. The oxygen value entered should not be accepted if it is not in the range between 1 and 100. If the calculated maximum average oxygen value of trainees is below 70 then throws custom exception that declare the trainees as unfit with meaningful message as “All trainees are unfit. Average Oxygen Values should be rounded.
3. Use inheritance to create an exception superclass and various exception subclasses. Write a program to demonstrate that the catch specifying the superclass catches subclass exceptions.
4. Create three new types of exceptions. Write a class with a method that throws all three. In main (), call the method but only use a single catch clause that will catch all three types of exceptions.
5. Modify Exercise 4 by adding a finally clause. Verify your finally clause is executed, even if a NullPointerException is thrown.
6. Create your own exception class using the extends keyword. Write a constructor for this class that takes a String argument and stores it inside the object with a String reference. Write a method that prints out the stored String. Create a try-catch clause to exercise your new exception.
7. Write a class with a method that throws an exception of the type created in Exercise 6. Try compiling it without an exception specification to see what the compiler says. Add the appropriate exception specification. Try out your class and its exception inside a try-catch clause.
8. Write a program to demonstrate the implementation of multiple catch blocks in Java.
9. Write a program to show the applicability of finally blocks in the following three cases.

Case 1: When an exception does not occur

Case 2: When an exception occurs but not handled by the catch block

Case 3: When an exception occurs and is handled by the catch block

1. Write a program to show the propagation of unchecked exception through the calling chain.
2. Write a program to show the propagation of checked exception through the calling chain.
3. Write a Java program that reads a list of integers from the user and throws an exception if any numbers are duplicates.
4. Write a Java program to create a method that takes a string as input and throws an exception if the string does not contain vowels.