Collection Framework Exercises:

1. **(Duplicate Elimination)** Write a program that reads in a series of first names and eliminates duplicates by storing them in a Set. Allow the user to search for a first name.
2. **(Color Chooser)** Use a HashMap to create a reusable class for choosing one of the 13 predefined colors in class Color. The names of the colors should be used as keys, and the predefined Color objects should be used as values. Place this class in a package that can be imported into any Java program. Use your new class in an application that allows the user to select a color and draw a shape in that color.
3. **(Counting Duplicate Words)** Write a program that determines and prints the number of duplicate words in a sentence. Treat uppercase and lowercase letters the same. Ignore punctuation.
4. **(Inserting Elements in a** ***LinkedList*** **in Sorted Order)** Write a program that inserts 25 random integers from 0 to 100 in order into a LinkedList object. The program should sort the elements, then calculate the sum of the elements and the floating-point average of the elements.
5. **(Copying and Reversing** ***LinkedList*s)** Write a program that creates a LinkedList object of 10 characters, then creates a second LinkedList object containing a copy of the first list, but in reverse order.
6. **(Prime Numbers and Prime Factors)** Write a program that takes a whole number input from a user and determines whether it’s prime. If the number is not prime, display its unique prime factors. Remember that a prime number’s factors are only 1 and the prime number itself. Every number that’s not prime has a unique prime factorization. For example, consider the number 54. The prime factors of 54 are 2, 3, 3 and 3. When the values are multiplied together, the result is 54. For the number 54, the prime factors output should be 2 and 3. Use Sets as part of your solution.
7. **(Sorting Words with a** ***TreeSet*)** Write a program that uses a String method split to tokenize a line of text input by the user and places each token in a TreeSet. Print the elements of the TreeSet. [Note: This should cause the elements to be printed in ascending sorted order.]