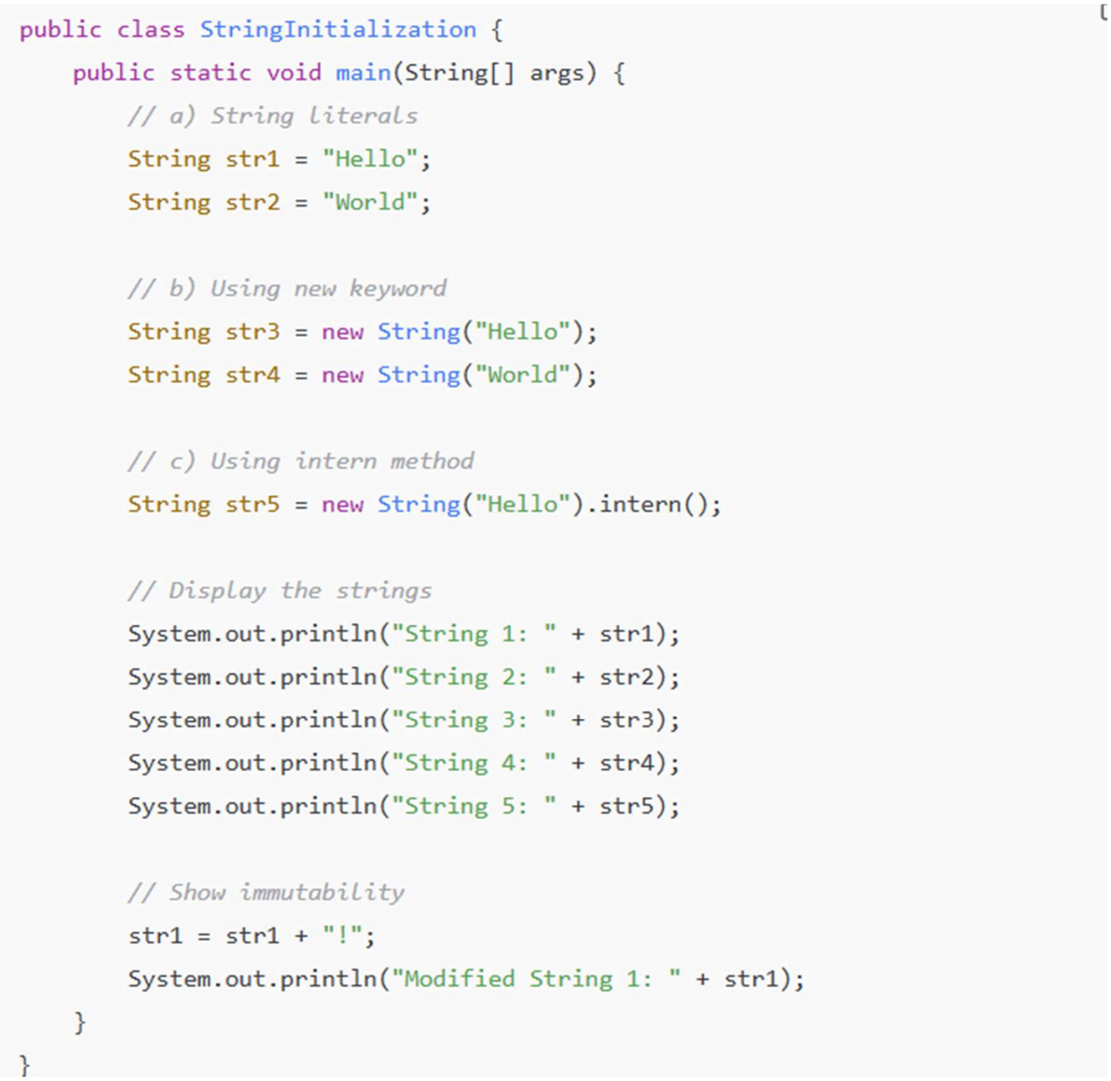
LAB # 01

INTRODUCTION TO STRING POOL, LITERALS, AND WRAPPER CLASSES

OBJECTIVE: To study the concepts of String Constant Pool, String literals, String immutability and Wrapper classes.

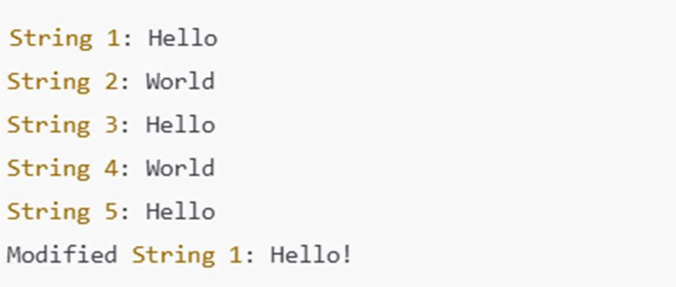
LAB TASKS

1. Write a program that initialize five different strings using all the above mentioned ways, i.e., a) string literals b) new keyword also use intern method and show string immutability.



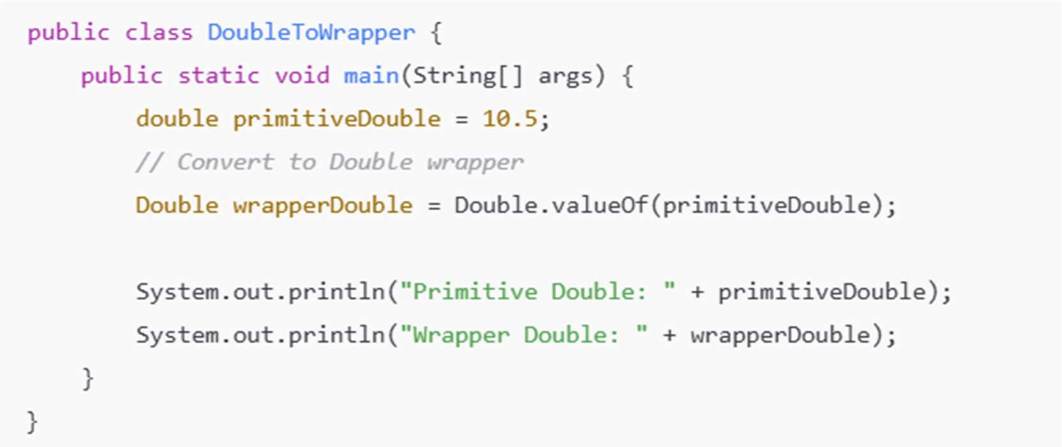
```
public class StringInitialization {
    public static void main(String[] args) {
        // a) String literals
        String str1 = "Hello";
        String str2 = "World";

        // b) Using new keyword
        String str3 = new String("Hello");
        String str4 = new String("World");

        // c) Using intern method
        String str5 = new String("Hello").intern();

        // Display the strings
        System.out.println("String 1: " + str1);
        System.out.println("String 2: " + str2);
        System.out.println("String 3: " + str3);
        System.out.println("String 4: " + str4);
        System.out.println("String 5: " + str5);

        // Show immutability
        str1 = str1 + "!";
        System.out.println("Modified String 1: " + str1);
    }
}
```

Output:



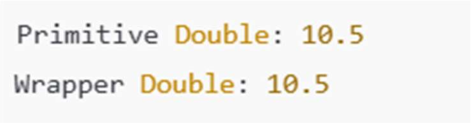
```
String 1: Hello
String 2: World
String 3: Hello
String 4: World
String 5: Hello
Modified String 1: Hello!
```

2. Write a program to convert primitive data type Double into its respective wrapper object.



```
public class DoubleToWrapper {
    public static void main(String[] args) {
        double primitiveDouble = 10.5;
        // Convert to Double wrapper
        Double wrapperDouble = Double.valueOf(primitiveDouble);

        System.out.println("Primitive Double: " + primitiveDouble);
        System.out.println("Wrapper Double: " + wrapperDouble);
    }
}
```

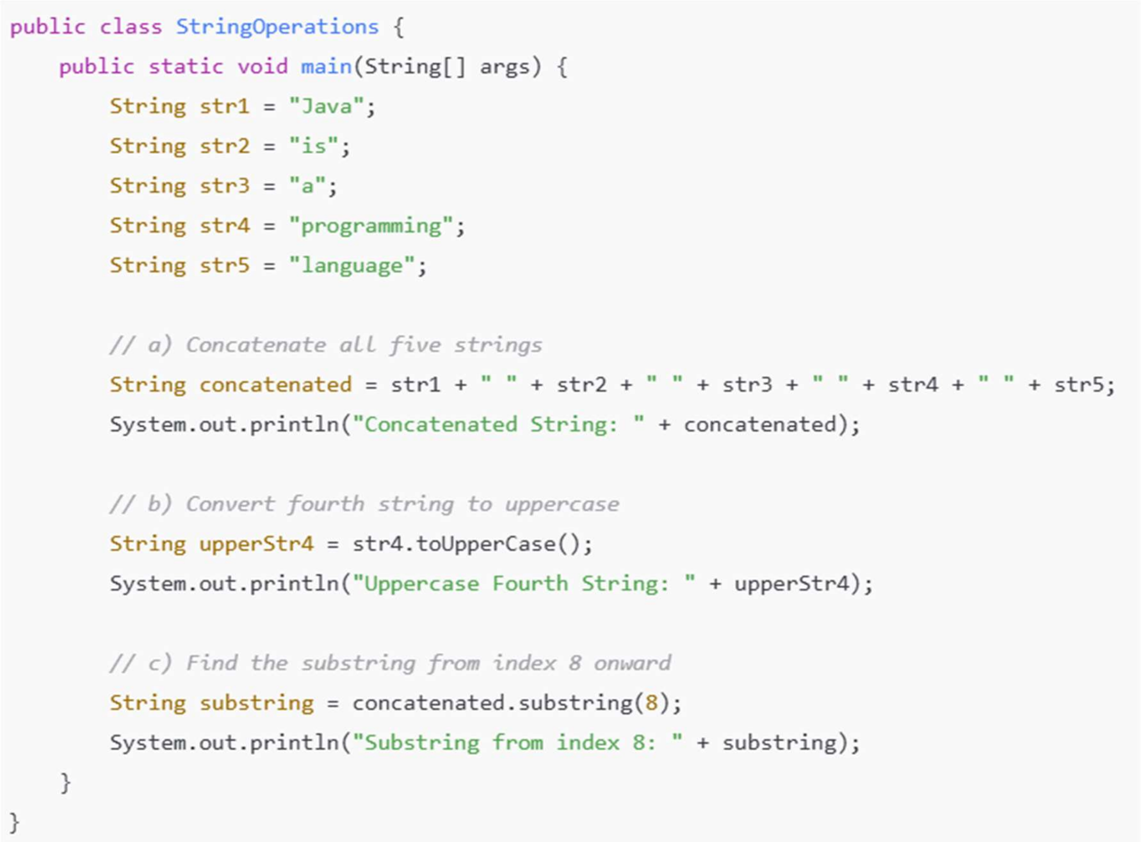
Output:



```
Primitive Double: 10.5
Wrapper Double: 10.5
```

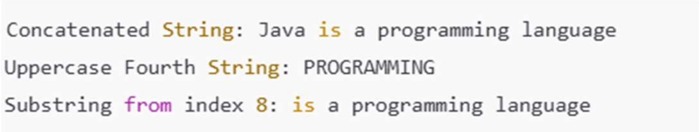
3 Write a program that initialize five different strings and perform the following operations.

a. Concatenate all five stings. b. Convert fourth string to uppercase. c. Find the substring from the concatenated string from 8 to onwards.



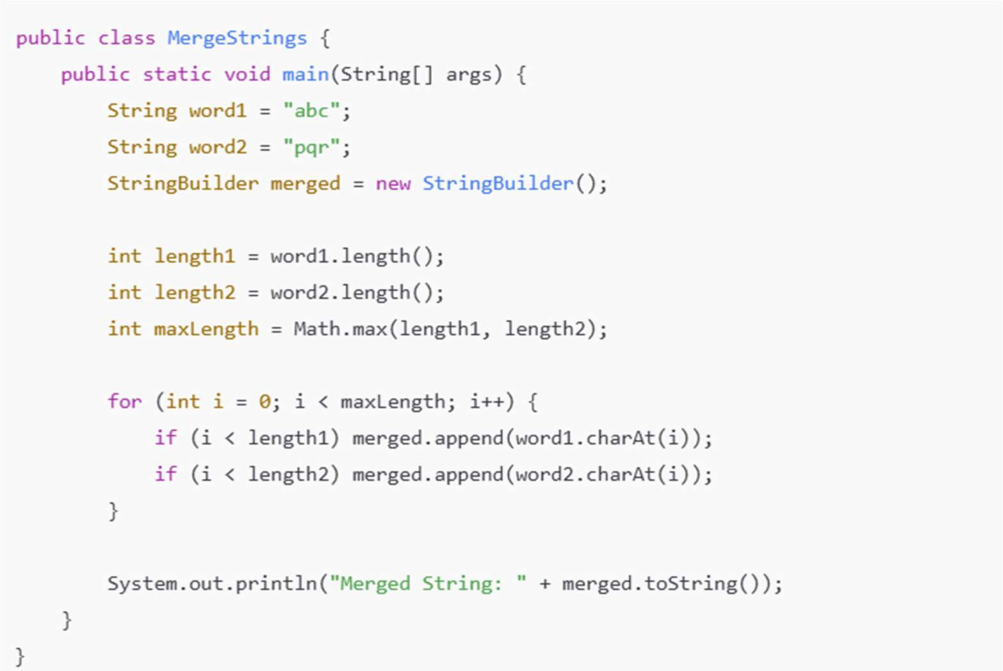
```java
public class StringOperations {
    public static void main(String[] args) {
        String str1 = "Java";
        String str2 = "is";
        String str3 = "a";
        String str4 = "programming";
        String str5 = "language";

        // a) Concatenate all five strings
        String concatenated = str1 + " " + str2 + " " + str3 + " " + str4 + " " + str5;
        System.out.println("Concatenated String: " + concatenated);

        // b) Convert fourth string to uppercase
        String upperStr4 = str4.toUpperCase();
        System.out.println("Uppercase Fourth String: " + upperStr4);

        // c) Find the substring from index 8 onward
        String substring = concatenated.substring(8);
        System.out.println("Substring from index 8: " + substring);
    }
}
```

Output:



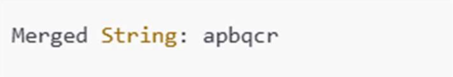
```
Concatenated String: Java is a programming language
Uppercase Fourth String: PROGRAMMING
Substring from index 8: is a programming language
```

4.You are given two strings word1 and word2. Merge the strings by adding letters in alternating order, starting with word1. If a string is longer than the other, append the additional letters onto the end of the merged string. Return the merged string.



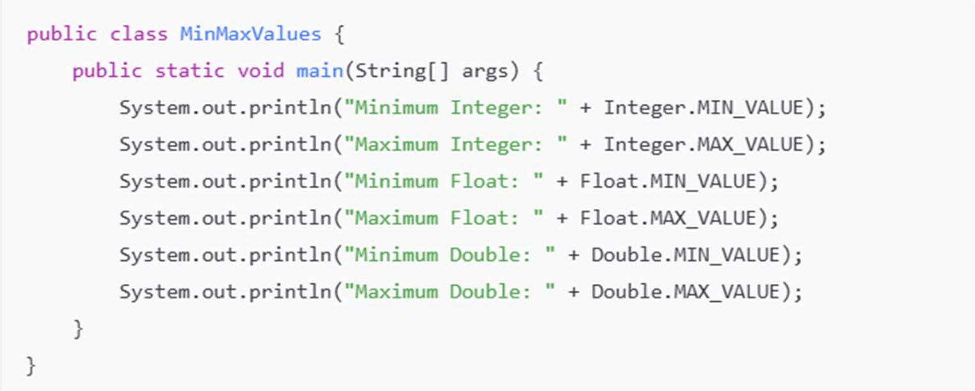
```
public class MergeStrings {
    public static void main(String[] args) {
        String word1 = "abc";
        String word2 = "pqr";
        StringBuilder merged = new StringBuilder();

        int length1 = word1.length();
        int length2 = word2.length();
        int maxLength = Math.max(length1, length2);

        for (int i = 0; i < maxLength; i++) {
            if (i < length1) merged.append(word1.charAt(i));
            if (i < length2) merged.append(word2.charAt(i));
        }

        System.out.println("Merged String: " + merged.toString());
    }
}
```

Output:



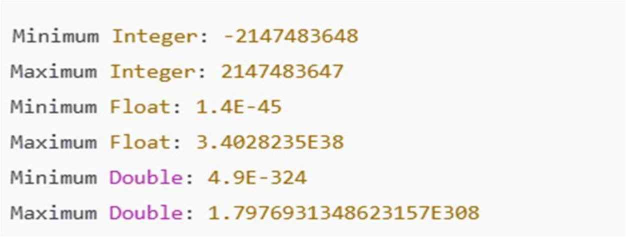
```
Merged String: apbqcr
```

5. Write a Java program to find the minimum and maximum values of Integer, Float, and Double using the respective wrapper class constants.



```
public class MinMaxValues {
    public static void main(String[] args) {
        System.out.println("Minimum Integer: " + Integer.MIN_VALUE);
        System.out.println("Maximum Integer: " + Integer.MAX_VALUE);
        System.out.println("Minimum Float: " + Float.MIN_VALUE);
        System.out.println("Maximum Float: " + Float.MAX_VALUE);
        System.out.println("Minimum Double: " + Double.MIN_VALUE);
        System.out.println("Maximum Double: " + Double.MAX_VALUE);
    }
}
```

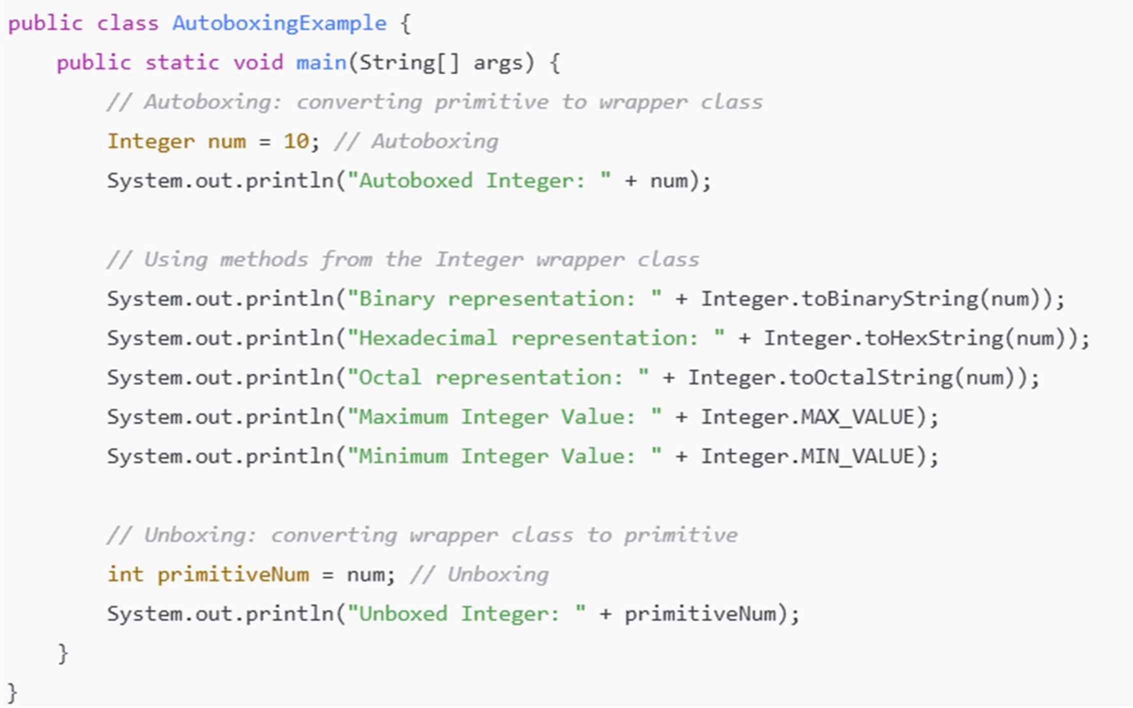
Output:



```
Minimum Integer: -2147483648
Maximum Integer: 2147483647
Minimum Float: 1.4E-45
Maximum Float: 3.4028235E38
Minimum Double: 4.9E-324
Maximum Double: 1.7976931348623157E308
```

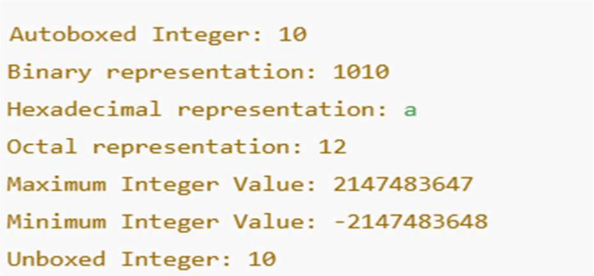
HOMETASK

1. Write a JAVA program to perform Autoboxing and also implement different methods of wrapper class.



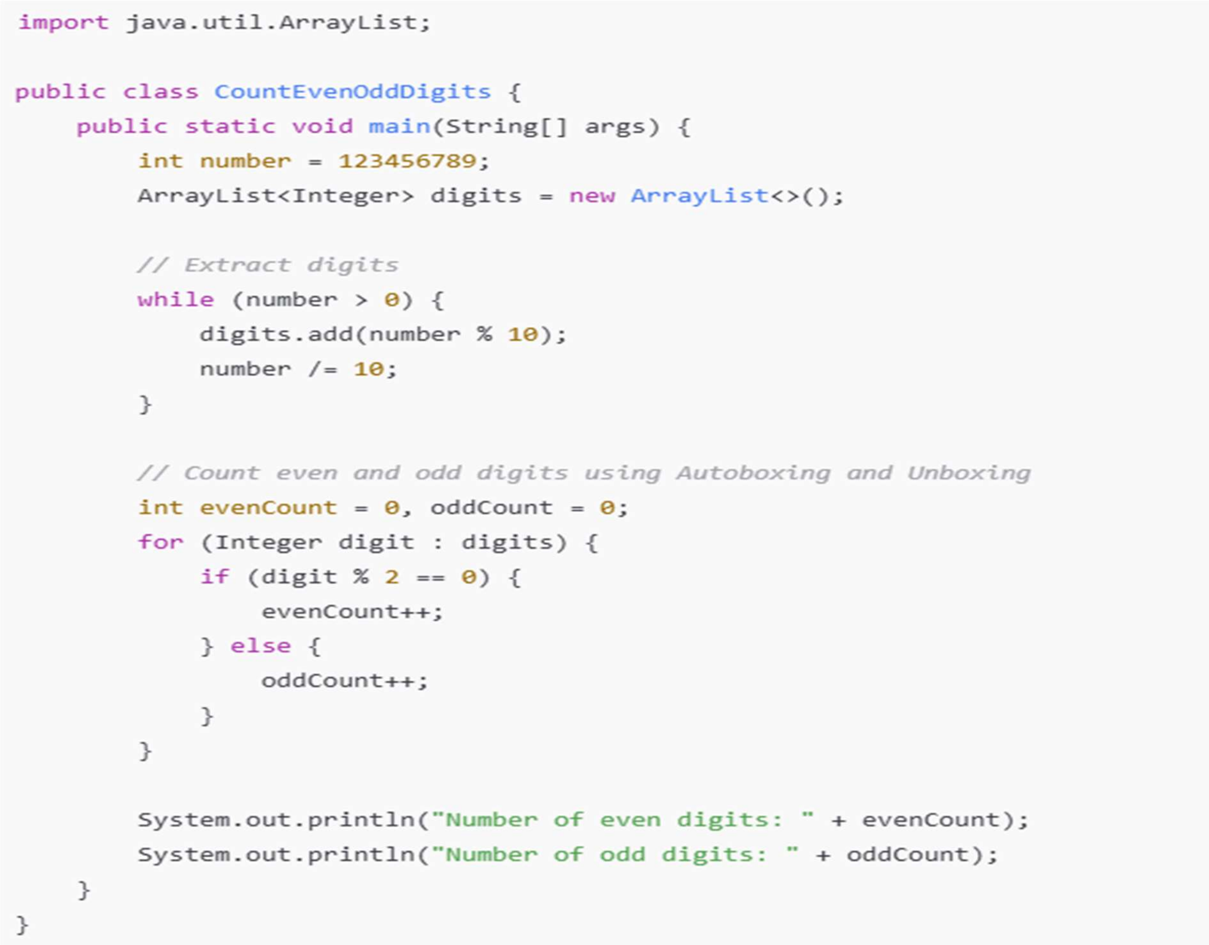
```java
public class AutoboxingExample {
    public static void main(String[] args) {
        // Autoboxing: converting primitive to wrapper class
        Integer num = 10; // Autoboxing
        System.out.println("Autoboxed Integer: " + num);

        // Using methods from the Integer wrapper class
        System.out.println("Binary representation: " + Integer.toBinaryString(num));
        System.out.println("Hexadecimal representation: " + Integer.toHexString(num));
        System.out.println("Octal representation: " + Integer.toOctalString(num));
        System.out.println("Maximum Integer Value: " + Integer.MAX_VALUE);
        System.out.println("Minimum Integer Value: " + Integer.MIN_VALUE);

        // Unboxing: converting wrapper class to primitive
        int primitiveNum = num; // Unboxing
        System.out.println("Unboxed Integer: " + primitiveNum);
    }
}
```

Output:



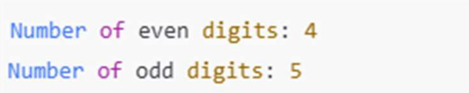
```
Autoboxed Integer: 10
Binary representation: 1010
Hexadecimal representation: a
Octal representation: 12
Maximum Integer Value: 2147483647
Minimum Integer Value: -2147483648
Unboxed Integer: 10
```

2. Write a Java program to count the number of even and odd digits in a given integer using Autoboxing and Unboxing.



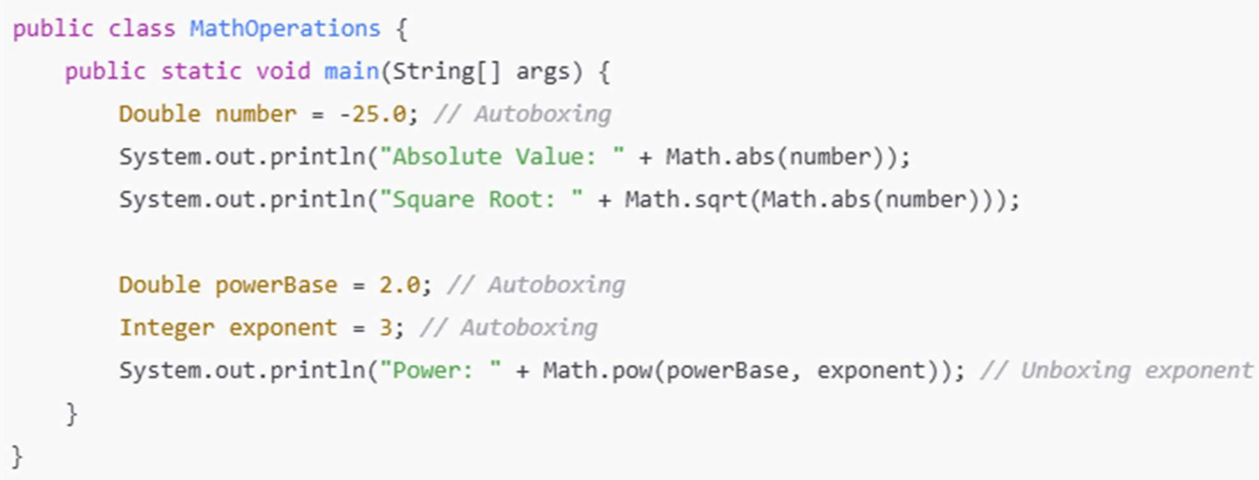
```
import java.util.ArrayList;

public class CountEvenOddDigits {
    public static void main(String[] args) {
        int number = 123456789;
        ArrayList<Integer> digits = new ArrayList<>();

        // Extract digits
        while (number > 0) {
            digits.add(number % 10);
            number /= 10;
        }

        // Count even and odd digits using Autoboxing and Unboxing
        int evenCount = 0, oddCount = 0;
        for (Integer digit : digits) {
            if (digit % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Number of even digits: " + evenCount);
        System.out.println("Number of odd digits: " + oddCount);
    }
}
```

Output:



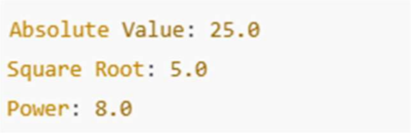
```
Number of even digits: 4
Number of odd digits: 5
```

3 Write a Java program to find the absolute value, square root, and power of a number using Math class methods, while utilizing Autoboxing and Wrapper classes.



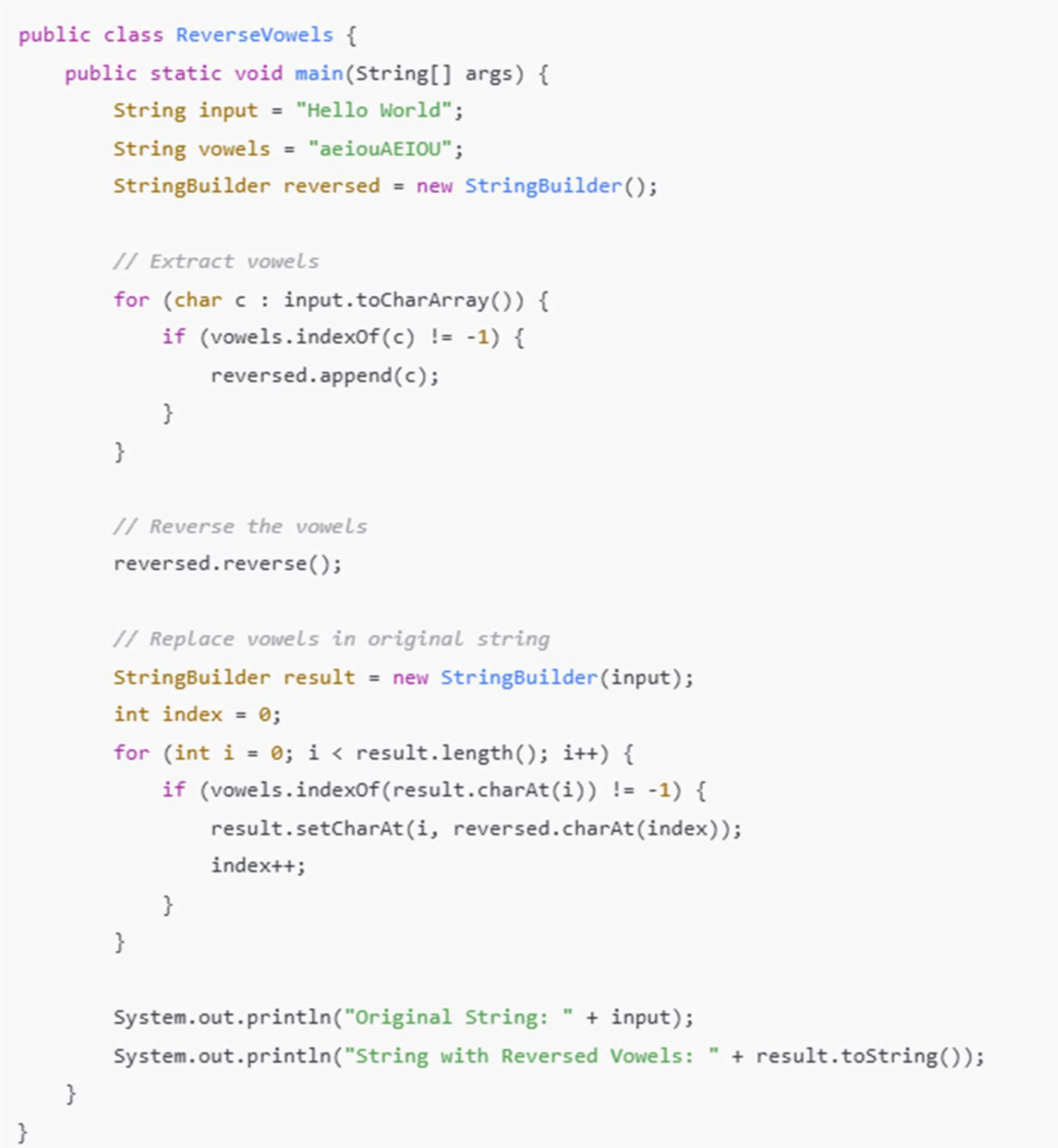
```
public class MathOperations {
    public static void main(String[] args) {
        Double number = -25.0; // Autoboxing
        System.out.println("Absolute Value: " + Math.abs(number));
        System.out.println("Square Root: " + Math.sqrt(Math.abs(number)));

        Double powerBase = 2.0; // Autoboxing
        Integer exponent = 3; // Autoboxing
        System.out.println("Power: " + Math.pow(powerBase, exponent)); // Unboxing exponent
    }
}
```

Output:



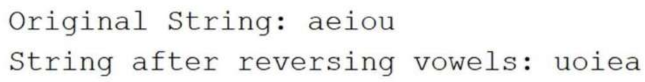
```
Absolute Value: 25.0
Square Root: 5.0
Power: 8.0
```

4. Write a Java program to reverse only the vowels in a string.



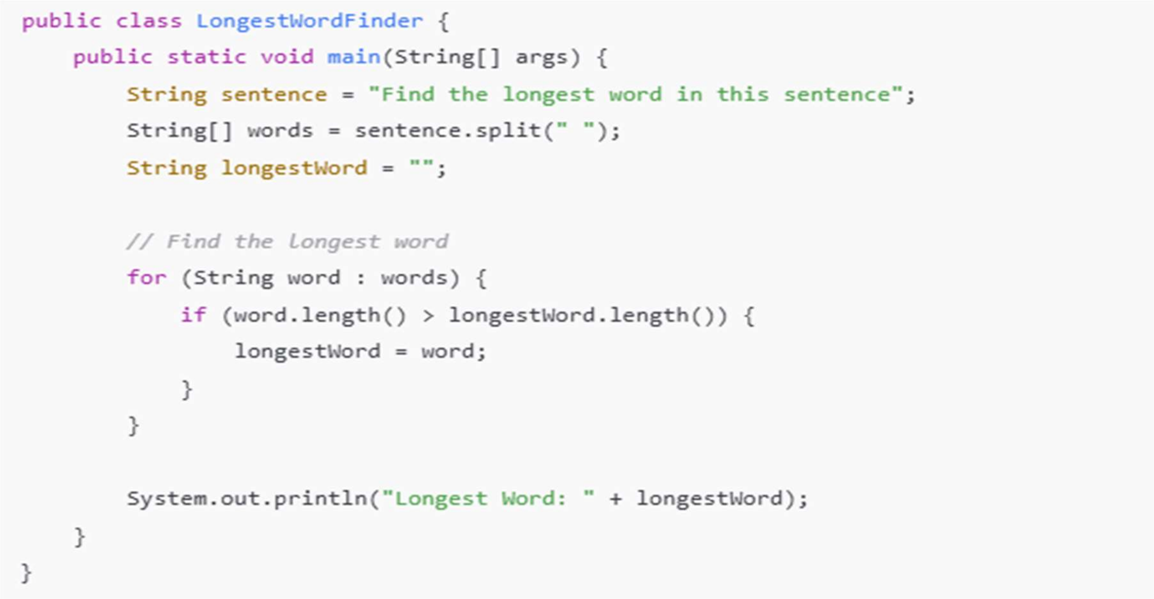
```java
public class ReverseVowels {
    public static void main(String[] args) {
        String input = "Hello World";
        String vowels = "aeiouAEIOU";
        StringBuilder reversed = new StringBuilder();

        // Extract vowels
        for (char c : input.toCharArray()) {
            if (vowels.indexOf(c) != -1) {
                reversed.append(c);
            }
        }

        // Reverse the vowels
        reversed.reverse();

        // Replace vowels in original string
        StringBuilder result = new StringBuilder(input);
        int index = 0;
        for (int i = 0; i < result.length(); i++) {
            if (vowels.indexOf(result.charAt(i)) != -1) {
                result.setCharAt(i, reversed.charAt(index));
                index++;
            }
        }

        System.out.println("Original String: " + input);
        System.out.println("String with Reversed Vowels: " + result.toString());
    }
}
```

Output:



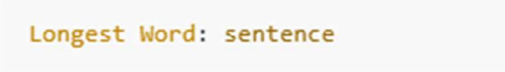
```
Original String: aeiou
String after reversing vowels: uoiea
```

5. Write a Java program to find the longest word in a sentence.



```
public class LongestWordFinder {
    public static void main(String[] args) {
        String sentence = "Find the longest word in this sentence";
        String[] words = sentence.split(" ");
        String longestWord = "";

        // Find the longest word
        for (String word : words) {
            if (word.length() > longestWord.length()) {
                longestWord = word;
            }
        }

        System.out.println("Longest Word: " + longestWord);
    }
}
```

Output:



```
Longest Word: sentence
```