Task 2

String Operations Write a method that takes two strings, concatenates them, reverses the result, and then extracts the middle substring of the given length. Ensure your method handles edge cases, such as an empty string or a substring length larger than the concatenated string.

ANS:

```
package Day16;
public class StringOperations {
  public static String processStrings(String str1, String str2, int
substringLength) {
      // Step 1: Concatenate the strings
      String concatenated = str1 + str2;
      // Step 2: Reverse the concatenated string
      String reversed = new
StringBuilder(concatenated).reverse().toString();
      // Step 3: Check if the desired substring length is greater than the
reversed string length
      if (substringLength > reversed.length()) {
          return "Substring length is larger than the concatenated string
length.";
      }
      // Step 4: Calculate the start index of the middle substring
      int start = (reversed.length() - substringLength) / 2;
      // Step 5: Extract the middle substring
      String middleSubstring = reversed.substring(start, start +
substringLength);
      return middleSubstring;
  }
  public static void main(String[] args) {
      // Test cases
      System.out.println(processStrings("hello", "world", 5));  // Expected:
"dlrow"
      System.out.println(processStrings("java", "program", 3));  //
Expected: "org"
      System.out.println(processStrings("", "", 2));  // Expected: "Substring
length is larger than the concatenated string length."
      System.out.println(processStrings("test", "", 2));  // Expected: "ts"
```

```
        System.out.println(processStrings("abc", "def", 10));  // Expected:
"Substring length is larger than the concatenated string length."
    }
}
```

OUTPUT:

```
rowol
orp
Substring length is larger than the concatenated string length.
se
Substring length is larger than the concatenated string length.
```