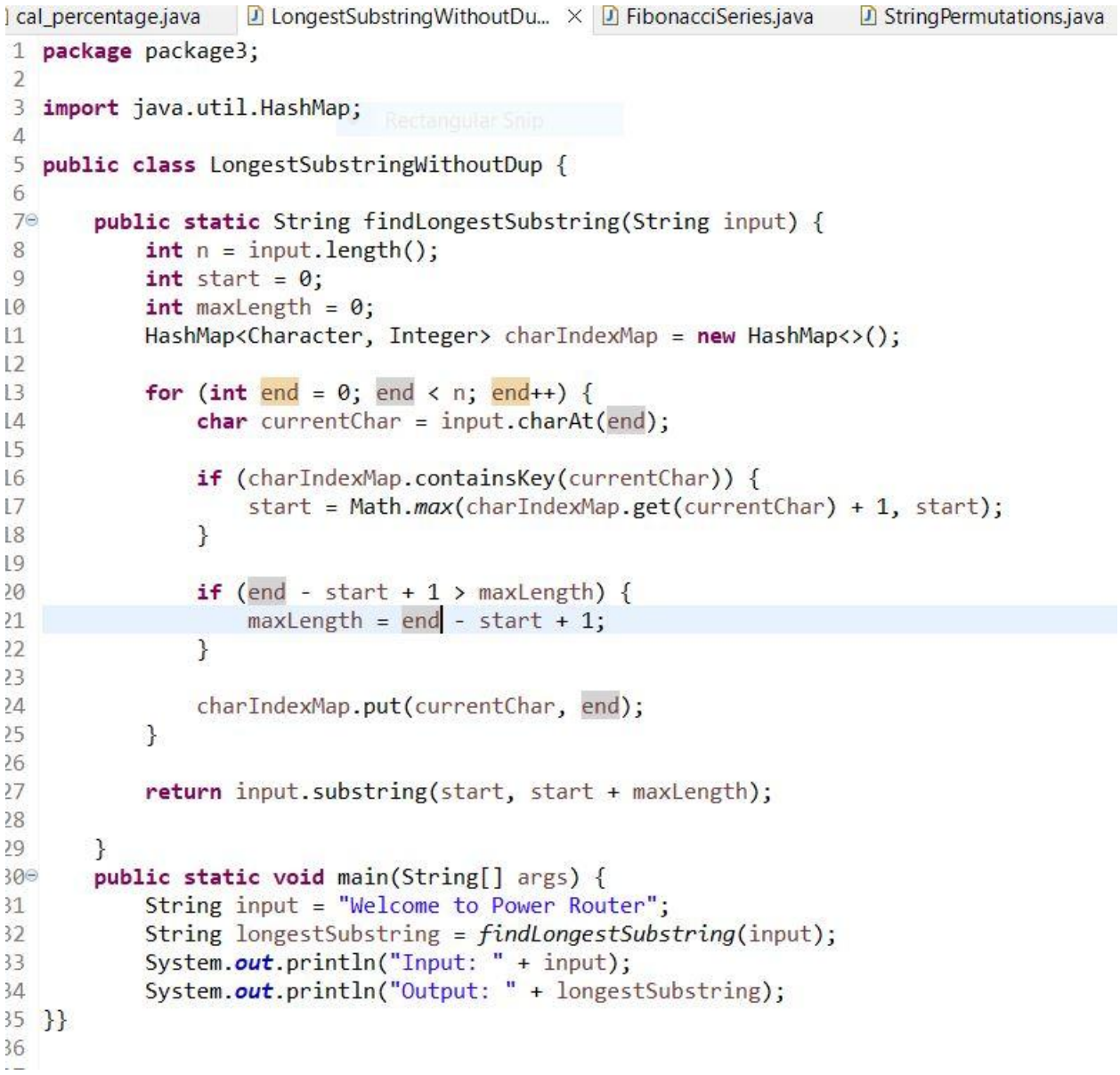
Q1 Write a Java program to find the longest substring from a given string that doesn't contain any duplicate characters.
Example:
○ Input : Welcome to PowerRouter.
○ Output : Welcome

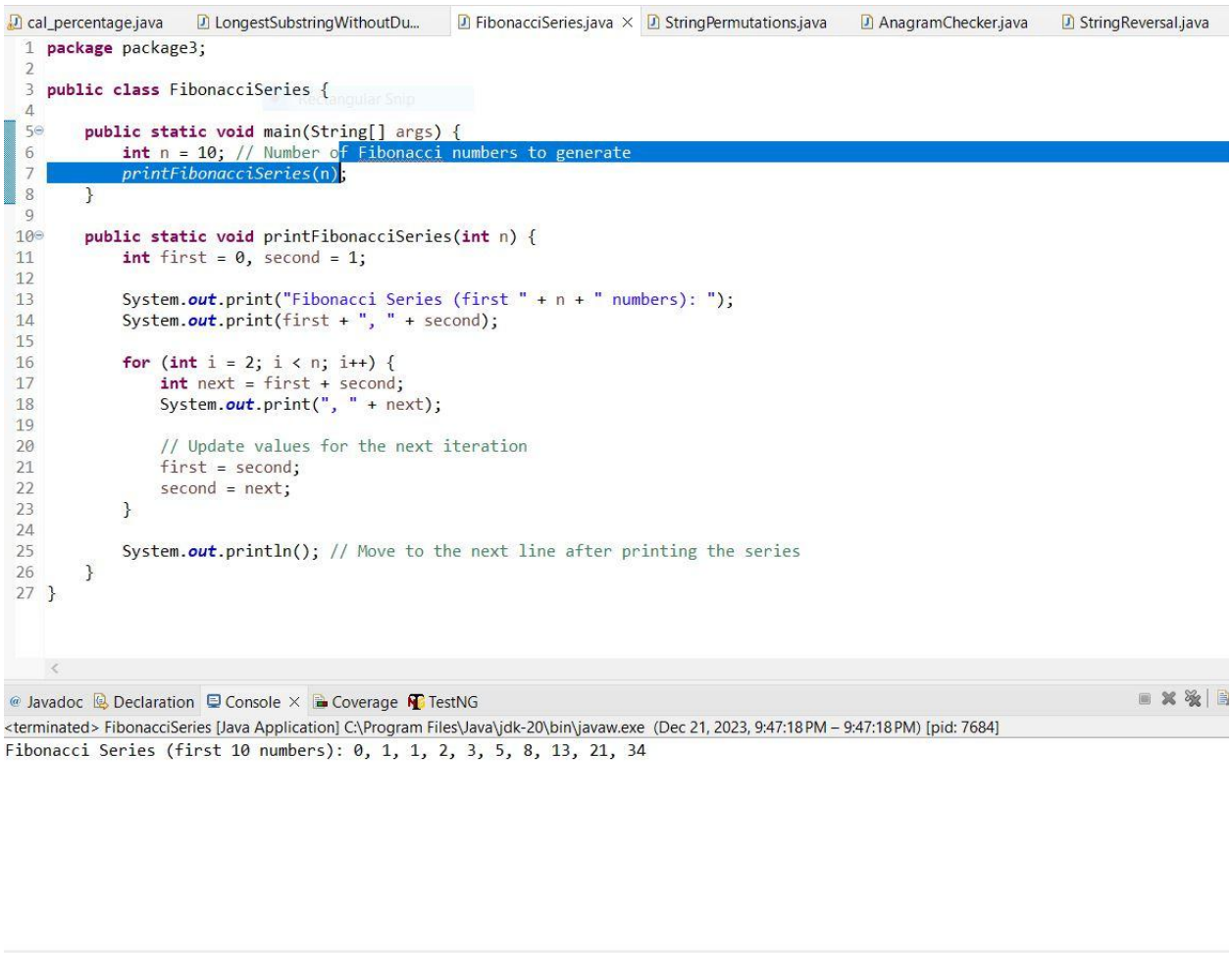
```
package package3;
import java.util.HashMap;
public class LongestSubstringWithoutDup {
        public static String findLongestSubstring(String input) {
                int n = input.length();
    int start = 0;
    int maxLength = 0;
    HashMap<Character, Integer> charIndexMap = new HashMap<>();
    for (int end = 0; end < n; end++) {
       char currentChar = input.charAt(end);
       if (charIndexMap.containsKey(currentChar)) {
          start = Math.max(charIndexMap.get(currentChar) + 1, start);
       }
       if (end - start + 1 > maxLength) {
          maxLength = end - start + 1;
       }
       charIndexMap.put(currentChar, end);
    }

                return input.substring(start, start + maxLength);

        }
        public static void main(String[] args) {
    String input = "Welcome to Power Router";
    String longestSubstring = findLongestSubstring(input);
    System.out.println("Input: " + input);
    System.out.println("Output: " + longestSubstring);
}}
```



Q2 Write a Program for the first 10 numbers of Fibonacci series.
Example:
○ Output : 0,1,1,2,3,5,8,13,21,34

```
package package3;
public class FibonacciSeries {
        public static void main(String[] args) {
    int n = 10; // Number of Fibonacci numbers to generate
    printFibonacciSeries(n);
  }
  public static void printFibonacciSeries(int n) {
    int first = 0, second = 1;
```

```
        System.out.print("Fibonacci Series (first " + n + " numbers): ");
        System.out.print(first + ", " + second);
        for (int i = 2; i < n; i++) {
            int next = first + second;
            System.out.print(", " + next);
            // Update values for the next iteration
            first = second;
            second = next;
        }
        System.out.println(); // Move to the next line after printing the series
    }
}
```



Q3 Write a program to print all permutations of a string.
Example:
- **Input : CAT**
- **Output : CAT , CTA , ACT , ATC , TAC , TCA**

```
package package3;
import java.util.Scanner;
public class StringPermutations {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter a string: ");
    String input = scanner.nextLine();
    System.out.println("Permutations of the string:");
    printPermutations(input);
    scanner.close();
  }
  public static void printPermutations(String str) {
    printPermutationsHelper(str, 0, str.length() - 1);
  }
  private static void printPermutationsHelper(String str, int left, int right) {
    if (left == right) {
      System.out.println(str);
    } else {
      for (int i = left; i <= right; i++) {
        str = swap(str, left, i);
        printPermutationsHelper(str, left + 1, right);
        str = swap(str, left, i); // backtrack
      }
    }
  }
  private static String swap(String str, int i, int j) {
    char[] charArray = str.toCharArray();
    char temp = charArray[i];
    charArray[i] = charArray[j];
    charArray[j] = temp;
    return new String(charArray);
  }
}
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java

```
11
12          System.out.println("Permutations of the string:");
13          printPermutations(input);
14
15          scanner.close();
16      }
17
18      public static void printPermutations(String str) {
19          printPermutationsHelper(str, 0, str.length() - 1);
20      }
21
22      private static void printPermutationsHelper(String str, int left, int right) {
23          if (left == right) {
24              System.out.println(str);
25          } else {
26              for (int i = left; i <= right; i++) {
27                  str = swap(str, left, i);
28                  printPermutationsHelper(str, left + 1, right);
29                  str = swap(str, left, i); // backtrack
30              }
31          }
32      }
33
34      private static String swap(String str, int i, int j) {
35          char[] charArray = str.toCharArray();
36          char temp = charArray[i];
37          charArray[i] = charArray[j];
38          charArray[j] = temp;
39          return new String(charArray);
40      }
41  }
```

Javadoc | Declaration | Console | Coverage | TestNG

<terminated> StringPermutations [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Dec 21, 2023, 9:50:06 PM

```
Enter a string: cat
Permutations of the string:
cat
cta
act
atc
tac
tca
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java

```java
package package3;

import java.util.Scanner;

public class StringPermutations {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        System.out.println("Permutations of the string:");
        printPermutations(input);

        scanner.close();
    }

    public static void printPermutations(String str) {
        printPermutationsHelper(str, 0, str.length() - 1);
    }

    private static void printPermutationsHelper(String str, int left, int right) {
        if (left == right) {
            System.out.println(str);
        } else {
            for (int i = left; i <= right; i++) {
                str = swap(str, left, i);
                printPermutationsHelper(str, left + 1, right);
                str = swap(str, left, i); // backtrack
            }
        }
```

Javadoc | Declaration | Console | Coverage | TestNG

<terminated> StringPermutations [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Dec 21, 2023, 9:50:06 PM -

```
Enter a string: cat
Permutations of the string:
cat
cta
act
atc
tac
tca
```

Q4: Write a program to check if two strings are Anagrams?
Example:
- **Input : Welcome**
- **Output : ceelmow**

```
package package3;
import java.util.Arrays;
import java.util.Scanner;
public class AnagramChecker {
  public static void main(String[] args) {
     Scanner scanner = new Scanner(System.in);
     System.out.print("Enter the first string: ");
     String str1 = scanner.nextLine().toLowerCase();
     System.out.print("Enter the second string: ");
     String str2 = scanner.nextLine().toLowerCase();
     if (areAnagrams(str1, str2)) {
        System.out.println("The strings are Anagrams.");
     } else {
        System.out.println("The strings are not Anagrams.");
     }
     scanner.close();
  }
  public static boolean areAnagrams(String str1, String str2) {
     // Remove spaces and convert to lowercase
     str1 = str1.replaceAll("\\s", "");
     str2 = str2.replaceAll("\\s", "");
     // Check if lengths are equal
     if (str1.length() != str2.length()) {
        return false;
     }
     // Convert strings to char arrays and sort them
     char[] charArray1 = str1.toCharArray();
     char[] charArray2 = str2.toCharArray();
     Arrays.sort(charArray1);
     Arrays.sort(charArray2);
     // Compare sorted char arrays
     return Arrays.equals(charArray1, charArray2);
  }
}
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java | AnagramChecker.java

```java
        System.out.print("Enter the first string: ");
        String str1 = scanner.nextLine().toLowerCase();

        System.out.print("Enter the second string: ");
        String str2 = scanner.nextLine().toLowerCase();

        if (areAnagrams(str1, str2)) {
            System.out.println("The strings are Anagrams.");
        } else {
            System.out.println("The strings are not Anagrams.");
        }

        scanner.close();
    }

    public static boolean areAnagrams(String str1, String str2) {
        // Remove spaces and convert to lowercase
        str1 = str1.replaceAll("\\s", "");
        str2 = str2.replaceAll("\\s", "");

        // Check if lengths are equal
        if (str1.length() != str2.length()) {
            return false;
        }

        // Convert strings to char arrays and sort them
        char[] charArray1 = str1.toCharArray();
        char[] charArray2 = str2.toCharArray();

        Arrays.sort(charArray1);
        Arrays.sort(charArray2);

        // Compare sorted char arrays
        return Arrays.equals(charArray1, charArray2);
    }
}
```

Javadoc | Declaration | Console | Coverage | TestNG

<terminated> AnagramChecker [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Dec 21, 2023, 9:51:23 PM – 9:51:48 PM) [pid: 15472]

```
Enter the first string: welcome
Enter the second string: elcomew
The strings are Anagrams.
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java | AnagramChecker.java

```java
package package3;

import java.util.Arrays;
import java.util.Scanner;

public class AnagramChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the first string: ");
        String str1 = scanner.nextLine().toLowerCase();

        System.out.print("Enter the second string: ");
        String str2 = scanner.nextLine().toLowerCase();

        if (areAnagrams(str1, str2)) {
            System.out.println("The strings are Anagrams.");
        } else {
            System.out.println("The strings are not Anagrams.");
        }

        scanner.close();
    }

    public static boolean areAnagrams(String str1, String str2) {
        // Remove spaces and convert to lowercase
        str1 = str1.replaceAll("\\s", "");
        str2 = str2.replaceAll("\\s", "");

        // Check if lengths are equal
        if (str1.length() != str2.length()) {
            return false;
        }

        // Convert strings to char arrays and sort them
        char[] charArray1 = str1.toCharArray();
        char[] charArray2 = str2.toCharArray();
```

Javadoc | Declaration | Console | Coverage | TestNG

<terminated> AnagramChecker [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Dec 21, 2023, 9:51:23 PM – 9:51:48 PM) [pid: 15472]

```
Enter the first string: welcome
Enter the second string: elcomew
The strings are Anagrams.
```

Q5: Write a program to reverse a string.
Example:
- **Input : Welcome**
- **Output : emoclew**

```
package package3;
import java.util.Scanner;
public class StringReversal {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter a string: ");
    String input = scanner.nextLine();
    String reversedString = reverseString(input);
    System.out.println("Reversed String: " + reversedString);
    scanner.close();
  }
  public static String reverseString(String str) {
    char[] charArray = str.toCharArray();
    int start = 0;
    int end = str.length() - 1;
    while (start < end) {
      // Swap characters at start and end indices
      char temp = charArray[start];
      charArray[start] = charArray[end];
      charArray[end] = temp;
      // Move indices towards the center
      start++;
      end--;
    }
    return new String(charArray);
  }
}
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java | AnagramChecker.java | StringReversal.java

```
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        String reversedString = reverseString(input);

        System.out.println("Reversed String: " + reversedString);

        scanner.close();
    }

    public static String reverseString(String str) {
        char[] charArray = str.toCharArray();

        int start = 0;
        int end = str.length() - 1;

        while (start < end) {
            // Swap characters at start and end indices
            char temp = charArray[start];
            charArray[start] = charArray[end];
            charArray[end] = temp;

            // Move indices towards the center
            start++;
            end--;
        }

        return new String(charArray);
    }
}
```

Javadoc | Declaration | Console | Coverage | TestNG

```
<terminated> StringReversal [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe  (Dec 21, 2023, 9:48:33 PM – 9:48:40 PM) [pid: 9296]
Enter a string: welcome
Reversed String: emoclew
```

cal_percentage.java | LongestSubstringWithoutDu... | FibonacciSeries.java | StringPermutations.java | AnagramChecker.java | StringReversal.java | StackUsingQueues.java

```
package package3;

import java.util.Scanner;

public class StringReversal {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        String reversedString = reverseString(input);

        System.out.println("Reversed String: " + reversedString);

        scanner.close();
    }

    public static String reverseString(String str) {
        char[] charArray = str.toCharArray();

        int start = 0;
        int end = str.length() - 1;

        while (start < end) {
            // Swap characters at start and end indices
            char temp = charArray[start];
            charArray[start] = charArray[end];
            charArray[end] = temp;
```

Javadoc | Declaration | Console | Coverage | TestNG

```
<terminated> StringReversal [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe  (Dec 21, 2023, 9:48:33 PM – 9:48:40 PM) [pid: 9296]
Enter a string: welcome
Reversed String: emoclew
```

Q6:Implement stack using queue without using an array or linked list.

- **Push**
- **Pop**
- **Top**

```
package package3;
import java.util.LinkedList;
import java.util.Queue;
class StackUsingQueues {
  private Queue<Integer> mainQueue;
  private Queue<Integer> tempQueue;
  public StackUsingQueues() {
     mainQueue = new LinkedList<>();
     tempQueue = new LinkedList<>();
  }
  // Push element onto the stack
  public void push(int x) {
     // Move all elements from mainQueue to tempQueue
     while (!mainQueue.isEmpty()) {
        tempQueue.offer(mainQueue.poll());
     }
     // Add the new element to mainQueue
     mainQueue.offer(x);
     // Move elements back from tempQueue to mainQueue
     while (!tempQueue.isEmpty()) {
        mainQueue.offer(tempQueue.poll());
     }
  }
  // Removes the element on the top of the stack
  public void pop() {
     if (!mainQueue.isEmpty()) {
        mainQueue.poll();
     } else {
        System.out.println("Stack is empty. Cannot perform pop operation.");
     }
  }
  // Returns the element on the top of the stack
  public int top() {
     if (!mainQueue.isEmpty()) {
        return mainQueue.peek();
     } else {
        System.out.println("Stack is empty. Cannot perform top operation.");
        return -1; // indicating an empty stack
     }
  }
  // Returns true if the stack is empty
  public boolean isEmpty() {
     return mainQueue.isEmpty();
  }
```

```
    public static void main(String[] args) {
        StackUsingQueues stack = new StackUsingQueues();
        stack.push(1);
        stack.push(2);
        stack.push(3);
        System.out.println("Top element: " + stack.top()); // Output: 3
        stack.pop();
        System.out.println("Top element after pop: " + stack.top()); // Output: 2
        stack.pop();
        stack.pop();
        System.out.println("Is the stack empty? " + stack.isEmpty()); // Output: true
    }
}
```