Practice Assignment

By SHREYASI REJA

Question 1

Given an input string and a dictionary of words, find out if the input string can be segmented into a space-separated sequence of dictionary words. See following examples for more details.

This is a famous Google interview question, also being asked by many other companies now a days.

Consider the following dictionary

{ i, like, sam, sung, samsung, mobile, ice,

cream, icecream, man, go, mango}

Input: ilike

Output: Yes

The string can be segmented as "i like".

Input: ilikesamsung

Output: Yes

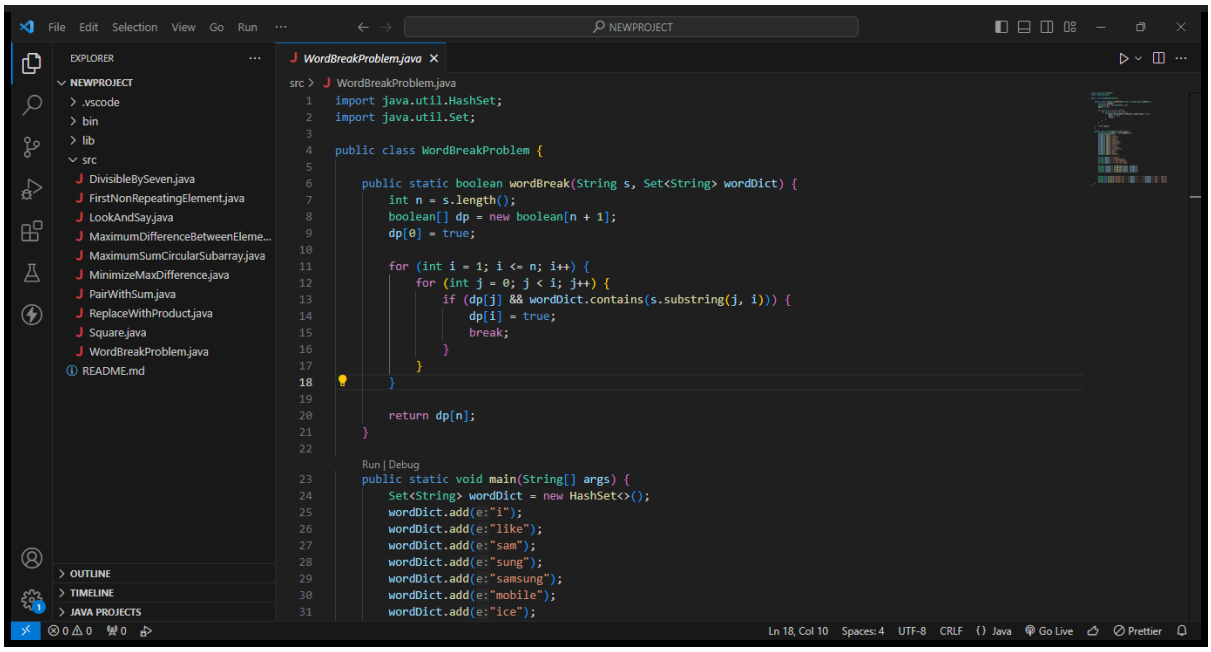
The string can be segmented as "i like samsung" or "i like sam sung".

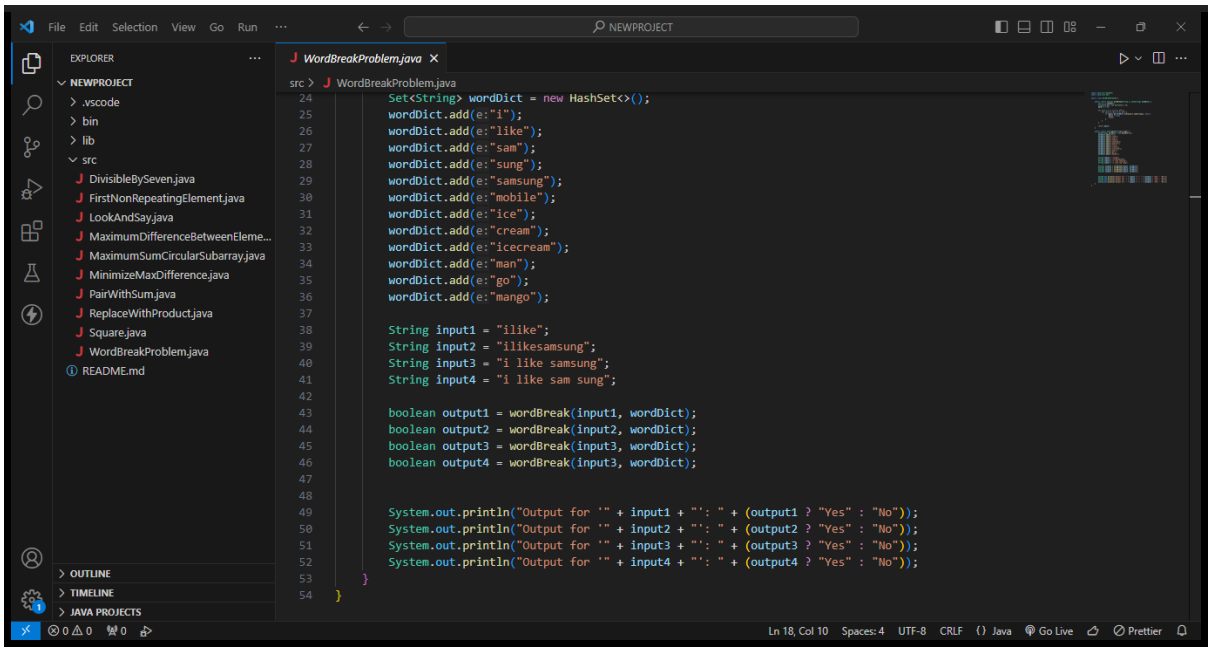
JAVA CODE:-

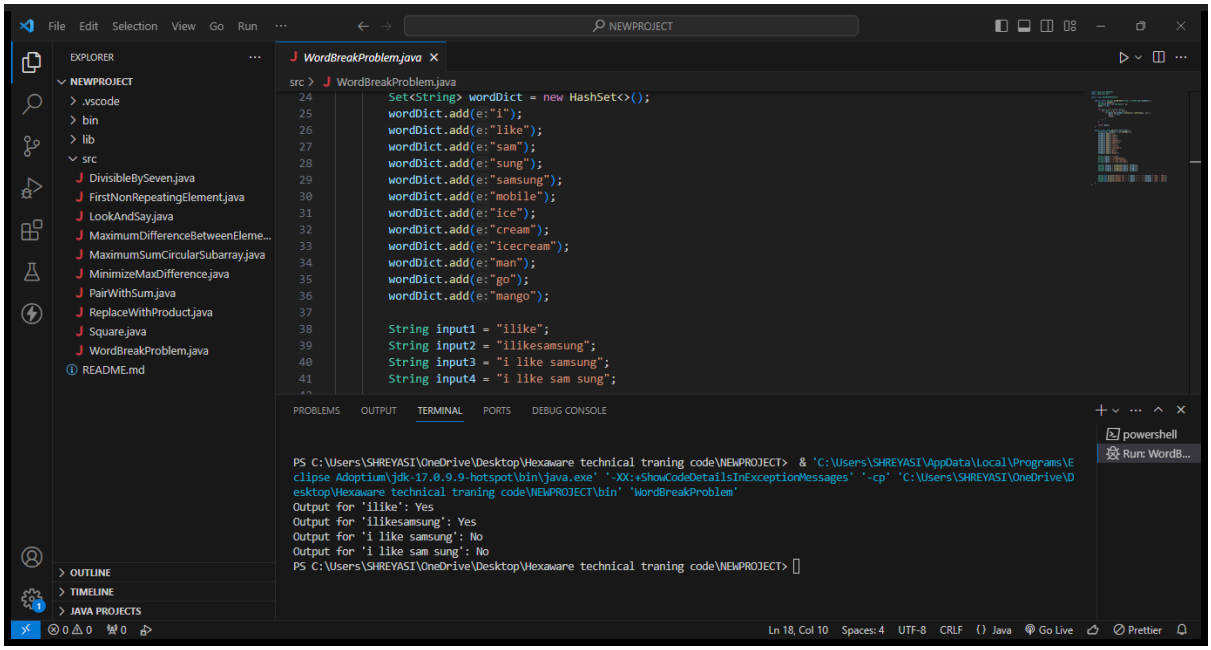
```
import java.util.HashSet;

import java.util.Set;


public class WordBreakProblem {


    public static boolean wordBreak(String s, Set<String> wordDict) {

        int n = s.length();

        boolean[] dp = new boolean[n + 1];

        dp[0] = true;


        for (int i = 1; i <= n; i++) {

            for (int j = 0; j < i; j++) {

                if (dp[j] && wordDict.contains(s.substring(j, i))) {

                    dp[i] = true;

                    break;

                }

            }

        }


        return dp[n];

    }


    public static void main(String[] args) {
```

```
Set<String> wordDict = new HashSet<>();
wordDict.add("i");
wordDict.add("like");
wordDict.add("sam");
wordDict.add("sung");
wordDict.add("samsung");
wordDict.add("mobile");
wordDict.add("ice");
wordDict.add("cream");
wordDict.add("icecream");
wordDict.add("man");
wordDict.add("go");
wordDict.add("mango");

String input1 = "ilike";
String input2 = "ilikesamsung";
String input3 = "i like samsung";
String input4 = "i like sam sung";

boolean output1 = wordBreak(input1, wordDict);
boolean output2 = wordBreak(input2, wordDict);
boolean output3 = wordBreak(input3, wordDict);
boolean output4 = wordBreak(input3, wordDict);


System.out.println("Output for '" + input1 + "': " + (output1 ? "Yes" : "No"));
System.out.println("Output for '" + input2 + "': " + (output2 ? "Yes" : "No"));
```

```
        System.out.println("Output for '" + input3 + "': " + (output3 ? "Yes" : "No"));

        System.out.println("Output for '" + input4 + "': " + (output4 ? "Yes" : "No"));

    }

}
```







Ques-2

A number can always be represented as a sum of squares of other numbers. Note that 1 is a square and we can always break a number as (1*1 + 1*1 + 1*1 + ...). Given a number n, find the minimum number of squares that sum to X.

Examples :

Input: n = 100

Output: 1

Explanation:

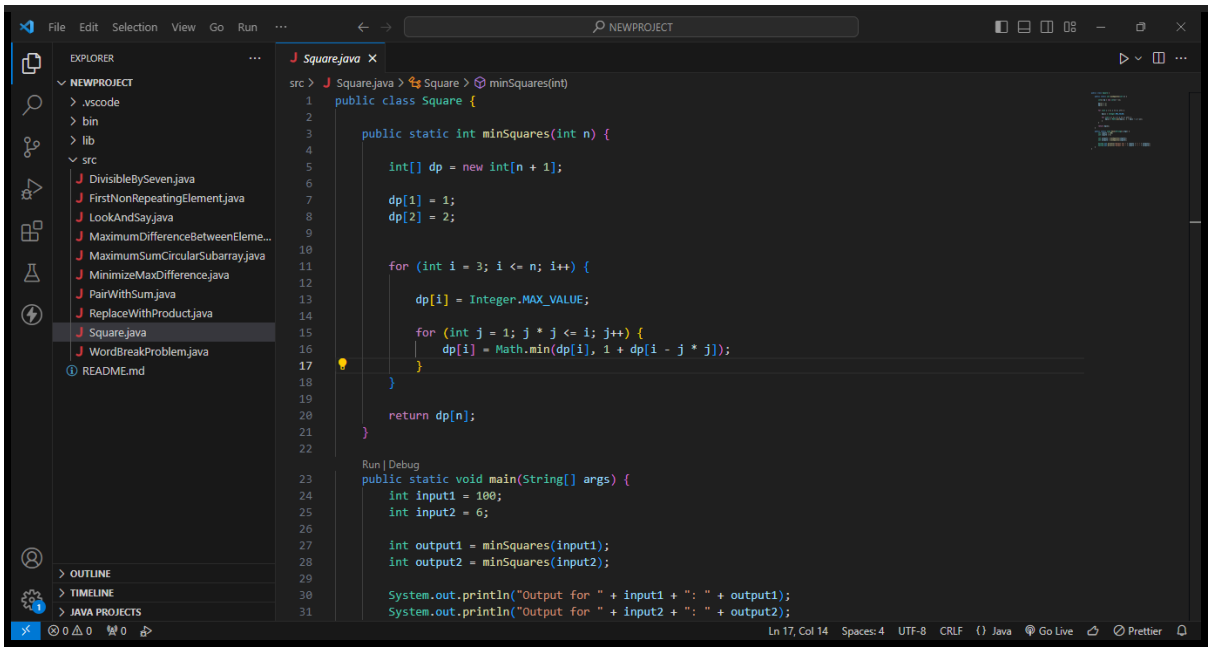
100 can be written as 102. Note that 100 can also be written as 52 + 52 + 52 + 52, but this representation requires 4 squares.

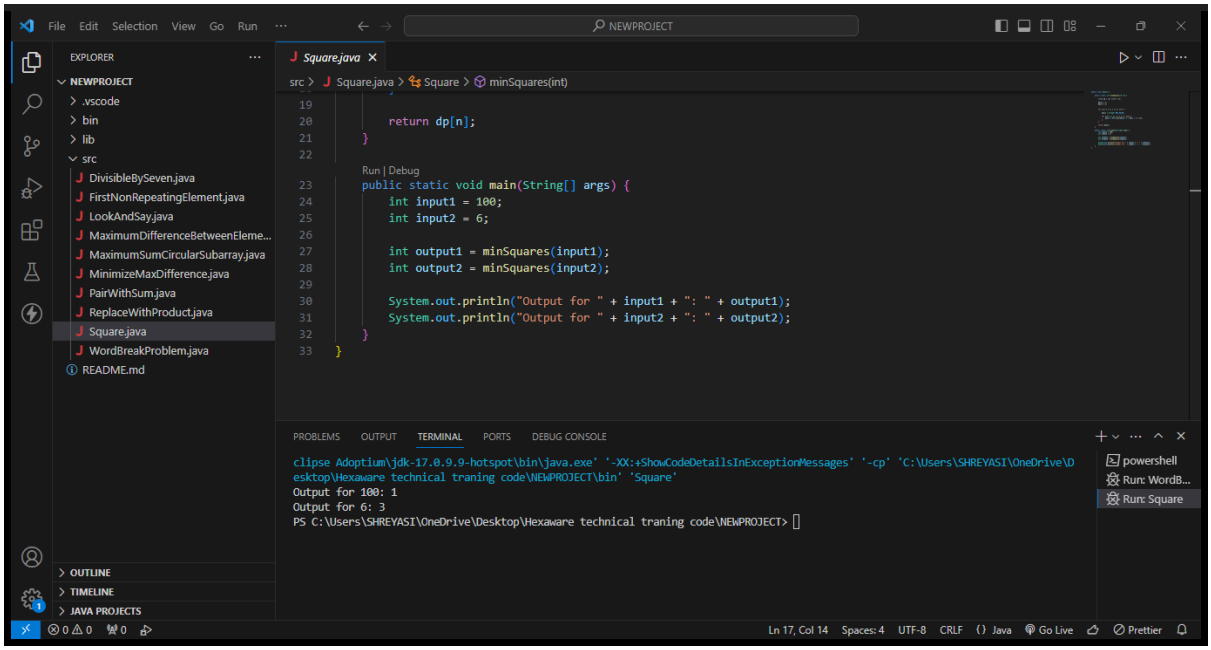
Input: n = 6

Output: 3

```
public class Square {

    public static int minSquares(int n) {

        int[] dp = new int[n + 1];

        dp[1] = 1;
        dp[2] = 2;

        for (int i = 3; i <= n; i++) {

            dp[i] = Integer.MAX_VALUE;

            for (int j = 1; j * j <= i; j++) {
                dp[i] = Math.min(dp[i], 1 + dp[i - j * j]);
            }
```

```
        }

        return dp[n];
    }

    public static void main(String[] args) {
        int input1 = 100;
        int input2 = 6;

        int output1 = minSquares(input1);
        int output2 = minSquares(input2);

        System.out.println("Output for " + input1 + ": " + output1);
        System.out.println("Output for " + input2 + ": " + output2);
    }
}
```





Ques-3

Given a number N, the task is to check if it is divisible by 7 or not.

Note: You are not allowed to use the modulo operator, floating point arithmetic is also not allowed.

Naive approach: A simple method is repeated subtraction. Following is another interesting method.

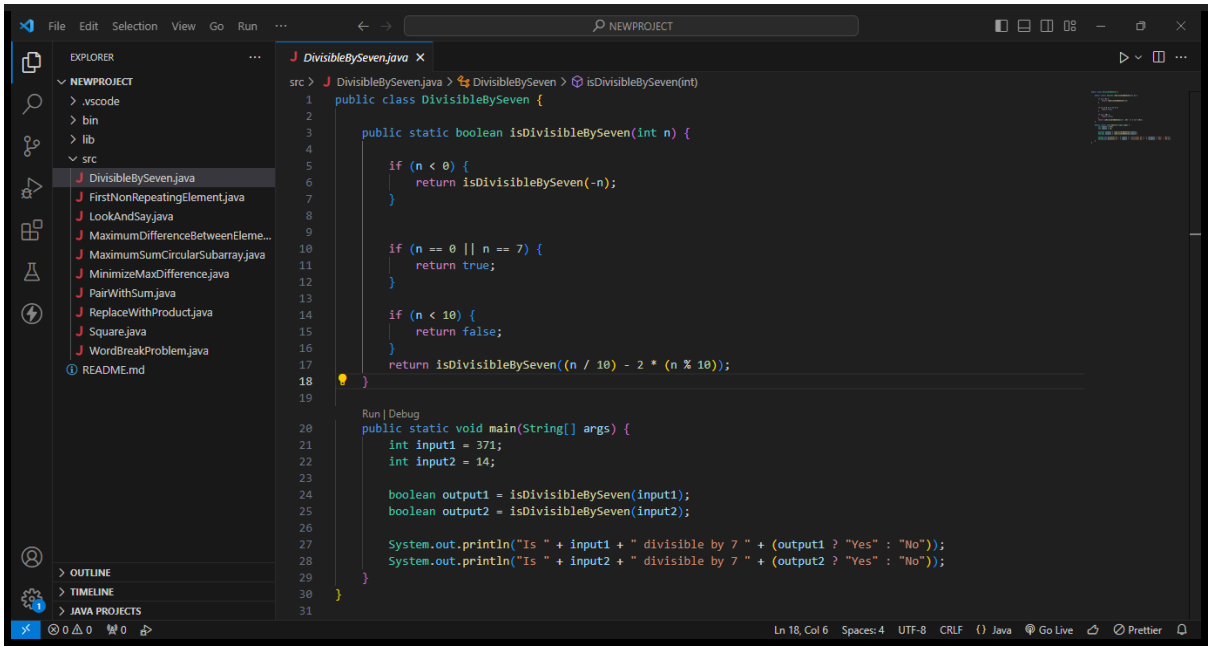
Divisibility by 7 can be checked by a recursive method. A number of the form 10a + b is divisible by 7 if and only if a – 2b is divisible by 7. In other words, subtract twice the last digit from the

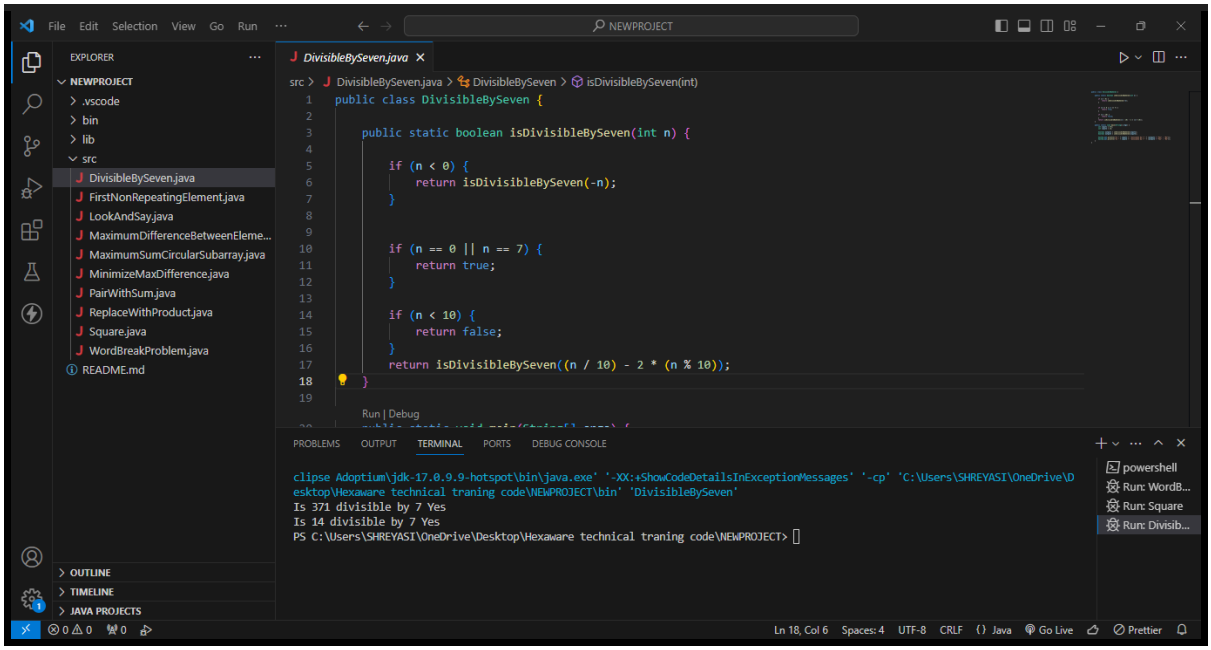
number formed by the remaining digits. Continue to do this until a small number.

Example: the number 371: 37 – (2×1) = 37 – 2 = 35; 3 – (2 × 5) = 3 – 10 = -7; thus, since -7 is divisible by 7, 371 is divisible by 7.

```
public class DivisibleBySeven {

  public static boolean isDivisibleBySeven(int n) {

    if (n < 0) {
      return isDivisibleBySeven(-n);
    }

    if (n == 0 || n == 7) {
      return true;
    }

    if (n < 10) {
      return false;
    }
    return isDivisibleBySeven((n / 10) - 2 * (n % 10));
  }
```

```
    public static void main(String[] args) {
        int input1 = 371;
        int input2 = 14;

        boolean output1 = isDivisibleBySeven(input1);
        boolean output2 = isDivisibleBySeven(input2);

        System.out.println("Is " + input1 + " divisible by 7 " + (output1 ? "Yes" : "No"));
        System.out.println("Is " + input2 + " divisible by 7 " + (output2 ? "Yes" : "No"));
    }
}
```





Question-4

Find the n’th term in Look-and-say (Or Count and Say) Sequence. The look-and-say sequence is the sequence of the below integers:

1, 11, 21, 1211, 111221, 312211, 13112221, 1113213211, …

How is the above sequence generated?

n’th term is generated by reading (n-1)’th term.

The first term is "1"

Second term is "11", generated by reading first term as "One 1"

(There is one 1 in previous term)

Third term is "21", generated by reading second term as "Two 1"

Fourth term is "1211", generated by reading third term as "One 2 One 1"

and so on

Input: n = 3

Output: 21

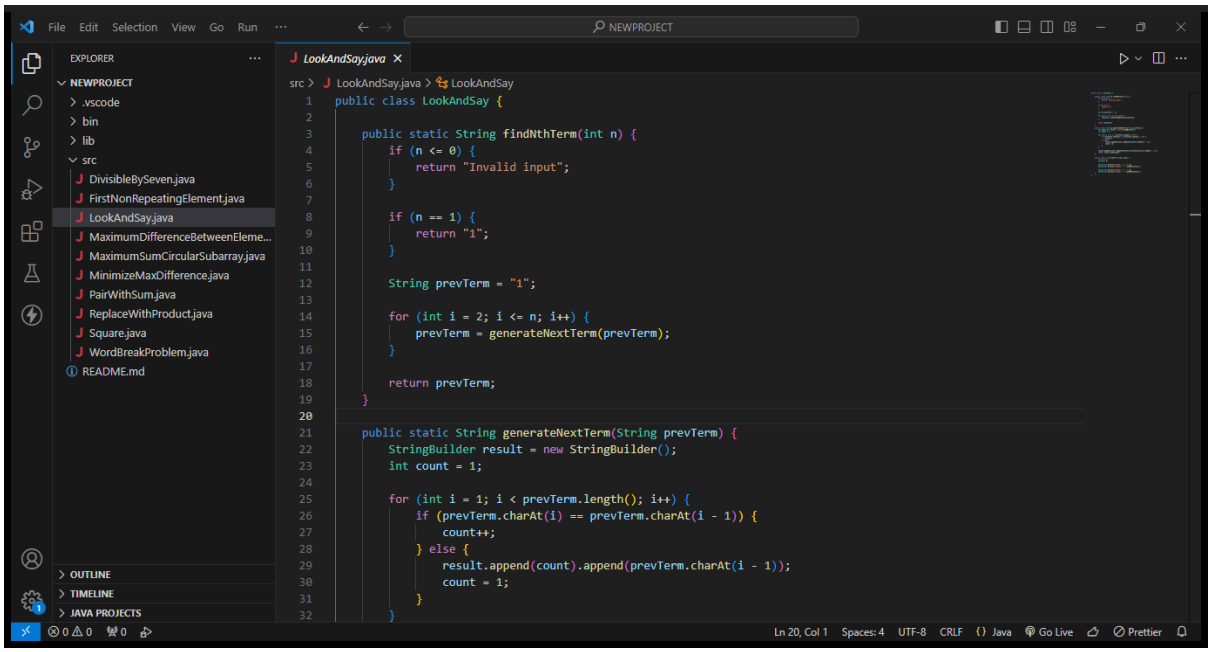
Input: n = 5

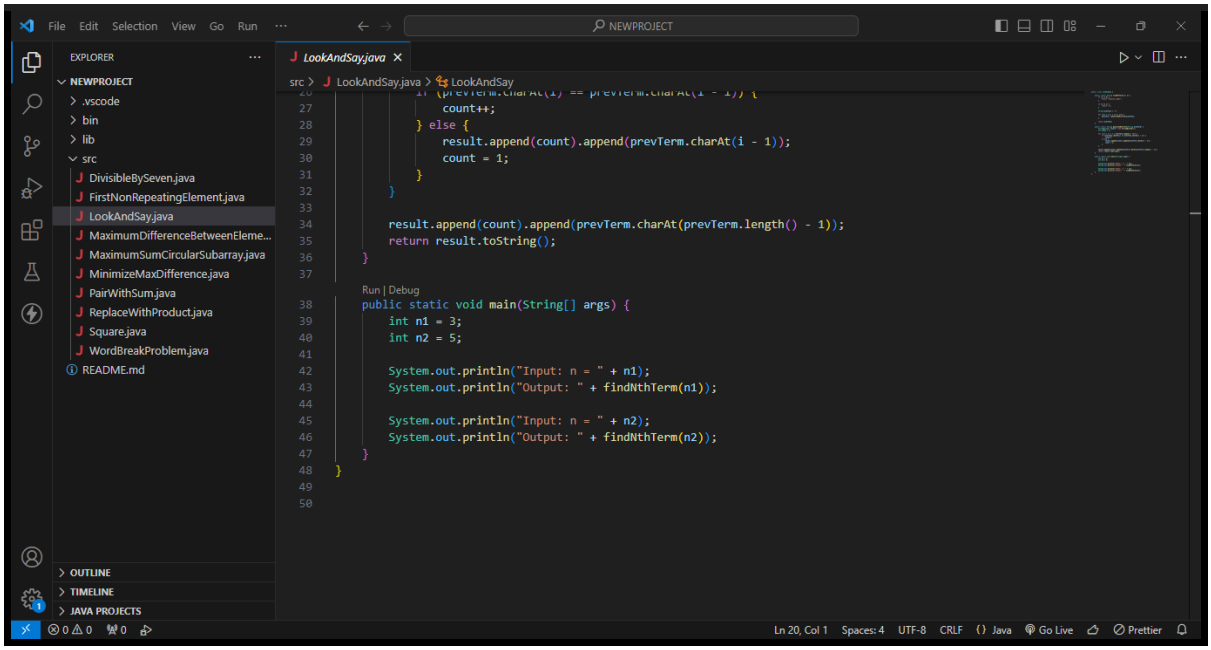
Output: 111221

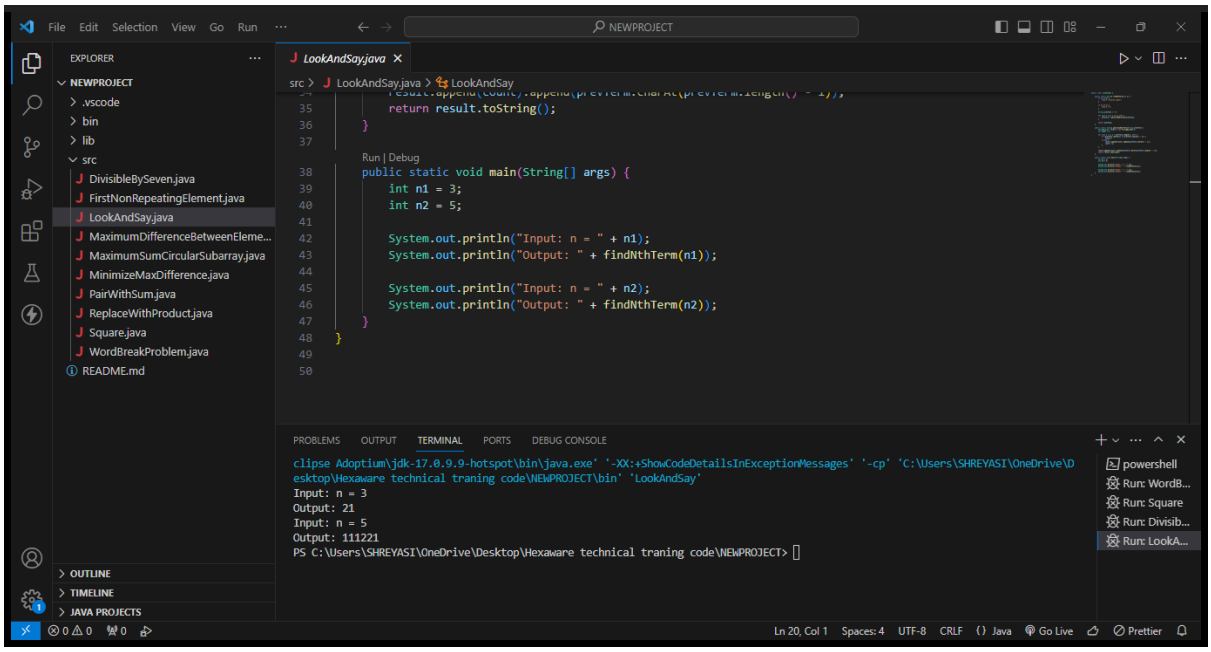
```
public class LookAndSay {

    public static String findNthTerm(int n) {
        if (n <= 0) {
            return "Invalid input";
        }

        if (n == 1) {
            return "1";
        }

        String prevTerm = "1";
```

```
    for (int i = 2; i <= n; i++) {
        prevTerm = generateNextTerm(prevTerm);
    }

    return prevTerm;
}

public static String generateNextTerm(String prevTerm) {
    StringBuilder result = new StringBuilder();
    int count = 1;

    for (int i = 1; i < prevTerm.length(); i++) {
        if (prevTerm.charAt(i) == prevTerm.charAt(i - 1)) {
            count++;
        } else {
            result.append(count).append(prevTerm.charAt(i - 1));
            count = 1;
        }
    }

    result.append(count).append(prevTerm.charAt(prevTerm.length() - 1));
    return result.toString();
}

public static void main(String[] args) {
    int n1 = 3;
```

```
        int n2 = 5;

        System.out.println("Input: n = " + n1);
        System.out.println("Output: " + findNthTerm(n1));

        System.out.println("Input: n = " + n2);
        System.out.println("Output: " + findNthTerm(n2));
    }
}
```




File Edit Selection View Go Run
NEWPROJECT
EXPLORER
NEWPROJECT
.vscode
bin
lib
src
DivisibleBySeven.java
FirstNonRepeatingElement.java
LookAndSay.java
MaximumDifferenceBetweenEleme...
MaximumSumCircularSubarray.java
MinimizeMaxDifference.java
PairWithSum.java
ReplaceWithProduct.java
Square.java
WordBreakProblem.java
README.md
OUTLINE
TIMELINE
JAVA PROJECTS
LookAndSay.java
src > LookAndSay.java > LookAndSay
count++;
} else {
result.append(count).append(prevTerm.charAt(i - 1));
count = 1;
}
}
result.append(count).append(prevTerm.charAt(prevTerm.length() - 1));
return result.toString();
}
Run | Debug
public static void main(String[] args) {
int n1 = 3;
int n2 = 5;
System.out.println("Input: n = " + n1);
System.out.println("Output: " + findNthTerm(n1));
System.out.println("Input: n = " + n2);
System.out.println("Output: " + findNthTerm(n2));
}
}
Ln 20, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live Prettier


File Edit Selection View Go Run
NEWPROJECT
EXPLORER
NEWPROJECT
.vscode
bin
lib
src
DivisibleBySeven.java
FirstNonRepeatingElement.java
LookAndSay.java
MaximumDifferenceBetweenEleme...
MaximumSumCircularSubarray.java
MinimizeMaxDifference.java
PairWithSum.java
ReplaceWithProduct.java
Square.java
WordBreakProblem.java
README.md
OUTLINE
TIMELINE
JAVA PROJECTS
LookAndSay.java
src > LookAndSay.java > LookAndSay
return result.toString();
}
Run | Debug
public static void main(String[] args) {
int n1 = 3;
int n2 = 5;
System.out.println("Input: n = " + n1);
System.out.println("Output: " + findNthTerm(n1));
System.out.println("Input: n = " + n2);
System.out.println("Output: " + findNthTerm(n2));
}
}
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE
clipse Adoptium\jdk-17.0.9.9-hotspot\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\SHREYASI\OneDrive\Desktop\Hexaware technical traning code\NEWPROJECT\bin' 'LookAndSay'
Input: n = 3
Output: 21
Input: n = 5
Output: 111221
PS C:\Users\SHREYASI\OneDrive\Desktop\Hexaware technical traning code\NEWPROJECT>
powershell
Run: WordB...
Run: Square
Run: Divisib...
Run: LookA...
Ln 20, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live Prettier