

## 1. Reverse a String

Write a program to reverse a given string without using the built-in reverse() method.

**Example:**

- Input: "Java"
- Output: "avaJ"

```
java Copy Edit

public class ReverseString {
    public static void main(String[] args) {
        String str = "Java";
        String reversed = "";
        for (int i = str.length() - 1; i >= 0; i--) {
            reversed += str.charAt(i);
        }
        System.out.println("Reversed String: " + reversed);
    }
}
```

## 2. Check if a String is Palindrome

Write a program to check if a given string is a palindrome (reads the same backward and forward).

**Example:**

- Input: "madam"
- Output: true

```
java Copy Edit

public class Palindrome {
    public static void main(String[] args) {
        String str = "madam";
        String reversed = new StringBuilder(str).reverse().toString();
        if (str.equals(reversed)) {
            System.out.println("The string is a palindrome.");
        } else {
            System.out.println("The string is not a palindrome.");
        }
    }
}
```

### 3. Count the Number of Vowels in a String

Write a program to count the number of vowels in a given string.

**Example:**

- Input: "Automation"
- Output: 5

```
java Copy Edit

public class CountVowels {
    public static void main(String[] args) {
        String str = "Automation";
        int count = 0;
        for (int i = 0; i < str.length(); i++) {
            char ch = str.charAt(i);
            if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
                ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
                count++;
            }
        }
        System.out.println("Number of vowels: " + count);
    }
}
```

### 4. Count Occurrences of a Character in a String

Write a program to count the occurrences of a specific character in a string.

**Example:**

- Input: "hello world", Character: 'o'
- Output: 2

```

Java

import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class CharacterCounter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String inputString = scanner.nextLine();
        scanner.close();

        Map<Character, Integer> charCountMap = new HashMap<>();

        for (char ch : inputString.toCharArray()) {
            charCountMap.put(ch, charCountMap.getOrDefault(ch, 0) + 1);
        }

        System.out.println("Character occurrences:");
        for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
            System.out.println("'" + entry.getKey() + "' : " + entry.getValue());
        }
    }
}

```

## 5. Remove Duplicate Characters from a String

Write a program to remove duplicate characters from a string.

**Example:**

- Input: "aabbc"
- Output: "abc"

```

// USE THIS EDITOR TO WRITE, COMPILER AND RUN YOUR JAVA CODE ONLINE
import java.util.*;
class Main {
    public static void main(String[] args) {
        String str="lavanya";
        Set<Character> set = new LinkedHashSet<>();
        for(char c : str.toCharArray())
        {
            set.add(c);
        }
        StringBuilder sb = new StringBuilder();
        for(char c : set)
        {
            sb.append(c);
        }
        System.out.println("Duplicate Elements:"+sb);
    }
}

```

## 6. Find the nth non-repeating character in a string using Java

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        String str = "programming";
        int k=2;
        Map<Character, Integer> charCounts = new HashMap<>();
        for (int i=0;i<str.length();i++) {
            char c=str.charAt(i);
            charCounts.put(c, charCounts.getOrDefault(c, 0) + 1);
        }

        int nonRepeatingCount = 0;
        for(int i=0;i<str.length();i++){
            char c=str.charAt(i);
            if(charCounts.get(c) == 1){
                nonRepeatingCount++;
                if(nonRepeatingCount == k){
                    System.out.println("Nth Non-repeating character is:"+c);
                }
            }
        }
    }
}
```

## 7. Check if Two Strings are Anagrams

Write a program to check if two strings are anagrams (contain the same characters in any order).

**Example:**

- Input: "listen", "silent"
- Output: true

```
java
```

 Copy  Edit

```
import java.util.Arrays;

public class AnagramCheck {
    public static void main(String[] args) {
        String str1 = "listen";
        String str2 = "silent";
        char[] charArray1 = str1.toCharArray();
        char[] charArray2 = str2.toCharArray();
        Arrays.sort(charArray1);
        Arrays.sort(charArray2);
        if (Arrays.equals(charArray1, charArray2)) {
            System.out.println("The strings are anagrams.");
        } else {
            System.out.println("The strings are not anagrams.");
        }
    }
}
```

## 8. Remove Whitespace from a String

Write a program to remove all spaces or whitespace characters from a string.

### Example:

- Input: "Hello World"
- Output: "HelloWorld"

```
java
```

 Copy  Edit

```
public class RemoveWhitespace {
    public static void main(String[] args) {
        String str = "Hello World";
        String result = str.replaceAll("\\s", "");
        System.out.println("String without spaces: " + result);
    }
}
```

## 9. Reverse the Words in a String

Write a program to reverse the words of a sentence.

### Example:

- Input: "Hello World"
- Output: "World Hello"

java

 Copy  Edit

```
public class ReverseWords {  
    public static void main(String[] args) {  
        String str = "Hello World";  
        String[] words = str.split(" ");  
        StringBuilder reversed = new StringBuilder();  
        for (int i = words.length - 1; i >= 0; i--) {  
            reversed.append(words[i]).append(" ");  
        }  
        System.out.println("Reversed words: " + reversed.toString().trim());  
    }  
}
```

## 10. Check if a String is a Substring of Another String

Write a program to check if one string is a substring of another string.

### Example:

- Input: "hello world", Substring: "world"
- Output: true

java

 Copy  Edit

```
public class SubstringCheck {  
    public static void main(String[] args) {  
        String str1 = "hello world";  
        String str2 = "world";  
        if (str1.contains(str2)) {  
            System.out.println(str2 + " is a substring of " + str1);  
        } else {  
            System.out.println(str2 + " is not a substring of " + str1);  
        }  
    }  
}
```

## 11. Duplicate words in a string

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        String str = "programming is language is java";
        String[] words = str.toLowerCase().split("\s+");
        Map<String, Integer> wordCountMap = new HashMap<>();
        for (String word : words) {
            wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);
        }
        System.out.println("Duplicate words:");
        int duplicateWordCount = 0;
        for(Map.Entry<String, Integer> entry : wordCountMap.entrySet()) {
            if(entry.getValue() > 1) {
                duplicateWordCount++;
                System.out.println(entry.getKey() + ":" + entry.getValue());
            }
        }
        System.out.println("Number of duplicate words: "+duplicateWordCount);
    }
}
```

## 12. Print first letter of each word in a string in java

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        String sentence = "Learn with Krishna Sandhi";
        StringBuilder initials = new StringBuilder();

        String[] words = sentence.split(" ");
        for (String word : words) {
            if (!word.isEmpty()) {
                initials.append(word.charAt(0));
            }
        }
        System.out.println("The first letters are: " + initials.toString());
    }
}
```

### 13. Java program to count the number of words in a string

```
package test;

public class Test
{
    public static void main(String[] args)
    {
        String s = "Welcome to Java World ";
        int count = 1;
        for (int i = 0; i < s.length() - 1; i++)
        {
            if ((s.charAt(i) == ' ') && (s.charAt(i + 1) != ' '))
            {
                count++;
            }
        }
        System.out.println("Number of words in a string: " + count);
    }
}
```

```
Number of words in a string: 4
```

### 14. Java program to find all permutations of a given string

```
package test;

public class Test
{
    public static void main(String[] args)
    {
        String str = "abc";
        permute(str, "");
    }

    static void permute(String str, String prefix)
    {
        if (str.length() == 0)
        {
            System.out.println(prefix);
        }
        else
        {
            for(int i = 0; i < str.length(); i++)
            {
                String rem = str.substring(0,i) + str.substring(i + 1);
                permute(rem, prefix + str.charAt(i));
            }
        }
    }
}
```

```
abc  
acb  
bac  
bca  
cab  
cba
```

## 15. Java program to print unique characters

```
public class Test  
{  
    public static void main(String[] args)  
    {  
        String str = "Java Automation ";  
        boolean[] unique = new boolean[128];  
        for (int i = 0; i < str.length(); i++)  
        {  
            char ch = str.charAt(i);  
            if (!unique[ch])  
            {  
                unique[ch] = true;  
                System.out.print(ch + " ");  
            }  
        }  
    }  
}
```

```
J a v   A u t o m i n
```

## 16. Java program to print even indexed characters

```
3 public class Test  
4 {  
5     public static void main(String[] args)  
6     {  
7         String str = "Automation ";  
8         for (int i = 0; i < str.length(); i++)  
9         {  
10            if (i % 2 == 0)  
11            {  
12                System.out.print(str.charAt(i));  
13            }  
14        }  
15    }  
16}  
17
```

```
Console × Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdl  
Atmto
```

## 17. Java program to print each letter twice from a given string

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named Test with a main method. The main method initializes a String variable str to "Automation", creates a StringBuilder object doubled, and then iterates through each character of str, appending it twice to doubled. Finally, it prints the result to the console. The console tab at the bottom shows the output: "Doubled String:AAuuttoommmaatttioonn".

```
1 package test;
2
3 public class Test
4 {
5     public static void main(String[] args)
6     {
7         String str = "Automation";
8         StringBuilder doubled = new StringBuilder();
9         for (int i = 0; i < str.length(); i++) {
10             char ch = str.charAt(i);
11             doubled.append(ch).append(ch);
12         }
13         System.out.println("Doubled String:" + doubled);
14     }
15 }
16
```

Console X Problems Debug Shell

<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.7.v20250502-0916\jre\bin  
Doubled String:AAuuttoommmaatttioonn

## 18. Java program to swap two string without using 3rd variable

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named Test with a main method. The main method initializes two String variables str1 ("hello") and str2 ("world"). It then swaps their values by concatenating them and using substrings to get the original values back. Finally, it prints the swapped values to the console. The console tab at the bottom shows the output: "Before swapping: str1 = hello, str2 = world" and "After swapping: str1 = world, str2 = hello".

```
1 package test;
2
3 public class Test {
4     public static void main(String[] args) {
5         String str1 = "hello";
6         String str2 = "world";
7         System.out.println("Before swapping: str1 = " + str1 + ", str2 = " + str2);
8         str1 = str1 + str2;
9         str2 = str1.substring(0, str1.length() - str2.length());
10        str1 = str1.substring(str2.length());
11        System.out.println("After swapping: str1 = " + str1 + ", str2 = " + str2);
12    }
13 }
14
```

Console X Problems Debug Shell

<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.7.v20250502-0916\jre\bin  
Before swapping: str1 = hello, str2 = world  
After swapping: str1 = world, str2 = hello

**19. Java program to gives two Output: “abcde”, “ABCDE” for the Input String Str = “aBACbcEDed”**

```
1 package test;
2
3 public class Test {
4     public static void main(String[] args) {
5         String str = "aBACbcEDed";
6         StringBuilder lowerCase = new StringBuilder();
7         StringBuilder upperCase = new StringBuilder();
8         for(char ch : str.toCharArray())
9         {
10            if(Character.isLowerCase(ch))
11            {
12                lowerCase.append(ch);
13            }
14            else
15            {
16                upperCase.append(ch);
17            }
18        }
19        System.out.println("Output in lowercase: "+lowerCase);
20        System.out.println("Output in uppercase "+upperCase); |
21    }
22 }
23
24
```

The screenshot shows the Eclipse IDE interface. The top part displays the Java code for the 'Test' class. The bottom part shows the 'Console' view with the following output:

```
Console X Problems Debug Shell
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0
Output in lowercase: abced
Output in uppercase BACED
```

## 20. Java program to gives two Output: “Subburaj”, “123” for the Input String Str = “Subbu123raj”

```
1 package test;
2
3 public class Test {
4     public static void main(String[] args) {
5         String str="Subbu123raj";
6         StringBuilder alphaPart = new StringBuilder();
7         StringBuilder numericPart = new StringBuilder();
8         for(char ch : str.toCharArray())
9         {
10             if(Character.isLetter(ch))
11             {
12                 alphaPart.append(ch);
13             }
14             else if (!Character.isDigit(ch))
15             {
16                 numericPart.append(ch);
17             }
18         }
19         System.out.println("Output in Alpha: "+alphaPart.toString());
20         System.out.println("Output in Numeric: " + numericPart.toString());
21     }
22 }
23 }
```

The screenshot shows the Eclipse IDE interface. The top part displays the Java code for the 'Test' class. The bottom part shows the 'Console' view where the program's output is printed. The output consists of two lines: 'Output in Alpha: Subburaj' and 'Output in Numeric: 123'. The console tab is active, and other tabs like 'Problems' and 'Debug Shell' are visible.

```
Console X Problems Debug Shell
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250
Output in Alpha: Subburaj
Output in Numeric: 123
```

**21. Java program to gives Output: “3241212000” for the Input String Str = “32400121200”**

```
1 package test;
2
3 public class Test {
4     public static void main(String[] args) {
5         String input = "32400121200";
6         StringBuilder digits = new StringBuilder();
7         StringBuilder nonDigits = new StringBuilder();
8
9         for (char c : input.toCharArray()) {
10             if (Character.isDigit(c)) {
11                 digits.append(c);
12             } else {
13                 nonDigits.append(c);
14             }
15         }
16         String result = digits.toString() + nonDigits.toString();
17         System.out.println("Output::" + result);
18     }
19 }
20
```

The screenshot shows the Eclipse IDE interface. The Java code for question 21 is displayed in the editor. In the bottom right corner of the editor, there is a small preview window showing the output of the program. Below the editor, the Eclipse tool bar has tabs for 'Console', 'Problems', and 'Debug Shell'. The 'Console' tab is active, showing the terminal output. The output text is: '<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\nOutput::32400121200'.

**22. Java program to gives Output: “00003241212” for the Input String Str = “32400121200”**

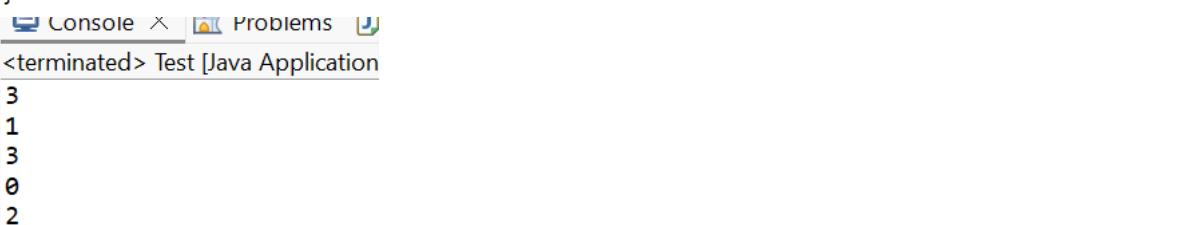
```
1 package test;
2
3 public class Test {
4     public static void main(String[] args) {
5         String input = "32400121200";
6         String formattedOutput = String.format("%011d", Long.parseLong(input));
7         System.out.println("Formatted output: " + formattedOutput);
8     }
9 }
10
```

The screenshot shows the Eclipse IDE interface. The Java code for question 22 is displayed in the editor. In the bottom right corner of the editor, there is a small preview window showing the output of the program. Below the editor, the Eclipse tool bar has tabs for 'Console', 'Problems', and 'Debug Shell'. The 'Console' tab is active, showing the terminal output. The output text is: '<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.7.v20250\nFormatted output: 32400121200'.

## 23. Java program to find the longest without repeating characters

```
import java.util.HashSet;
public class Test {
    public int lengthOfLongestSubstring(String s) {
        int n = s.length();
        if (n == 0) {
            return 0;
        }
        Set<Character> charSet = new HashSet<>();
        int maxLength = 0;
        int left = 0;

        for (int right = 0; right < n; right++) {
            char currentChar = s.charAt(right);
            while (charSet.contains(currentChar)) {
                charSet.remove(s.charAt(left));
                left++;
            }
            charSet.add(currentChar);
            maxLength = Math.max(maxLength, right - left + 1);
        }
        return maxLength;
    }
    public static void main(String[] args) {
        Test solution = new Test();
        System.out.println(solution.lengthOfLongestSubstring("abcabcbb")); // Output: 3
        System.out.println(solution.lengthOfLongestSubstring("bbbbbb")); // Output: 1
        System.out.println(solution.lengthOfLongestSubstring("pwwkew")); // Output: 3
        System.out.println(solution.lengthOfLongestSubstring(""))); // Output: 0
        System.out.println(solution.lengthOfLongestSubstring("au")); // Output: 2
    }
}
```



The screenshot shows an IDE interface with a code editor and a terminal window. The code editor contains the provided Java code. The terminal window is titled 'Console' and shows the output of the program for five different input strings: "abcabcbb", "bbbbbb", "pwwkew", "", and "au". The outputs are 3, 1, 3, 0, and 2 respectively, which correspond to the expected results mentioned in the code's comments.

```
Console × Problems
<terminated> Test [Java Application]
3
1
3
0
2
```