Revise this code as many time u can with each method and meaning

charAt() method used in the string

```
class Main {
    public static void main(String[] args) {
      //reverse of the string
        Scanner scanner= new Scanner(System.in);
        System.out.println("enter the String :");
       String name=scanner.nextLine();
      char c;
      String strrev="";
      for(int i=0;i<name.length();i++){
         c= name.charAt(i);
         strrev=c+strrev;
      }
      System.out.println("The reverse String :"+ strrev);
    }
}
```

```
output: enter the String :
nisha
The reverse String :ahsin
```

String[] words = input.split(" ");

Input: "Hello World Java"

After split → ["Hello", "World", "Java"]

// Online Java Compiler

```
// Use this editor to write, compile and run your Java code online

import java.util.*;

class Main {

    public static void main(String[] args) {

    // to reverse each word of a given string

        Scanner scanner= new Scanner(System.in);



        System.out.println("enter the String :");

       String input=scanner.nextLine();

        StringBuilder result = new StringBuilder();

        String[] words= input.split(" ");

        for(String word: words){

            StringBuilder reverseword = new StringBuilder(word);

            result.append(reverseword.reverse().append(" "));}

              System.out.println("enter  before reverse String :"+input);

        System.out.println("enter after reverse String :"+result.toString().trim());



    }

enter the String :

nisha prasad

enter  before reverse String :nisha prasad

enter after reverse String :ahsin dasarp

}
```

+++++++++++++++++++++++++++++++++

&&&&&&&&&&&&&&&&&&&&&&&&&&

%%%%%%%%%%%%%%%%%%%%%%

Type	Correct Usage	Returns
String	word.length()	Number of characters in the string
Array	array.length	Number of elements in the array

```
String word = "Hello";

System.out.println(word.length()); // Output: 5


int[] nums = {1, 2, 3, 4};

System.out.println(nums.length); // Output: 4
```

without stringbuilder

```
// Online Java Compiler

// Use this editor to write, compile and run your Java code online

import java.util.*;

class Main {

    public static void main(String[] args) {

    // to reverse each word of a given string// Online Java Compiler

// Use this editor to write, compile and run your Java code online

import java.util.*;

class Main {

    public static void main(String[] args) {

        System.out.println("Try programiz.pro");

        Scanner sc= new Scanner(System.in);

        String strs=sc.nextLine();

        String reverseString="";

        String[] words=strs.split(" ");

        for(int i=0;i<words.length;i++){
            String word= words[i];

            char c;

            String instr="";

            for(int j=0;j<=word.length()-1;j++){

               c=word.charAt(j);

               instr=c+instr;


            }

            reverseString=reverseString+instr+" ";


        }

        System.out.print("orginal string : "+strs);

        System.out.print(" reversre the string : "+reverseString);

    }

}
```

```
enter the String :

nisha prasad hello

enter  before reverse String :nisha prasad hello

enter after reverse String : ahsin dasarp olleh
```

"Hello World".replaceAll(" ", ""); // → "HelloWorld"

Note: replaceAll takes a regex. For literal single-character replacement, replace(" ", "") is slightly faster and clearer.

input.toCharArray()

Converts the String to a char[] so you can iterate characters easily.

"abc".toCharArray(); // → new char[]{'a','b','c'}

for (char c : "abc".toCharArray()) { ... }

Map vs HashMap — short & clear

Map is a Java interface (a contract). It defines operations for key→value collections (e.g., put, get, remove, entrySet, ...).

Declaring Map<Key,Value> m = ... is good practice because your code programs to an interface — you can later swap implementations without changing code that uses m.

HashMap is a concrete implementation of Map that stores entries in a hash table. Characteristics:

Average-case O(1) for get and put.

Does not preserve order (insertion order is not guaranteed).

Allows one null key and multiple null values.

Not synchronized (not thread-safe). For concurrent use, prefer ConcurrentHashMap.

Default initial capacity = 16, default load factor = 0.75 (automatically resizes when size > capacity * loadFactor).

map.put(c, map.getOrDefault(c, 0) + 1)

This line updates the count of character c:

If c not in map → count = 0, so we put 0+1 = 1.

If c already in map → count = old value, so we put old+1.

Example Walkthrough with "hello"

Initial map = {} (empty)

First iteration (c = 'h'):

map.getOrDefault('h', 0) → 0

map.put(’h’, 0+1) → map = {h=1}

Second iteration (c = ’e’):

map.getOrDefault(’e’, 0) → 0

map.put(’e’, 1) → map = {h=1, e=1}

Third iteration (c = ’l’):

map.getOrDefault(’l’, 0) → 0

map.put(’l’, 1) → map = {h=1, e=1, l=1}

Fourth iteration (c = ’l’ again):

map.getOrDefault(’l’, 0) → 1

map.put(’l’, 2) → map = {h=1, e=1, l=2}

Fifth iteration (c = ’o’):

map.getOrDefault(’o’, 0) → 0

map.put(’o’, 1) → map = {h=1, e=1, l=2, o=1}

************&&&&&&&&&&&&&&&&%%%%%%%%%%$$$$$$$$$$########@@@@@@@@@@@

Map<Character, Integer> charCount = new HashMap<>(); explain this format why this is wirtten like this ?

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

1. Map<K, V>

Map is an interface in Java that represents a collection of key-value pairs.

It takes two generic types:

K → the type of the key

V → the type of the value

So when you see:

Map<Character, Integer>

It means:

Key (K) → Character (each letter in the string)

Value (V) → Integer (the count of how many times that letter appears)

2. charCount

This is just the variable name we chose for the map.

It will hold our HashMap object.

3. new HashMap<>()

HashMap is a class that implements the Map interface.

new HashMap<>() creates an empty map in memory.

When we write:

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

It means:

We are declaring a variable of type Map (interface) → charCount

We are initializing it with a HashMap object (implementation).

✅ Why not just write HashMap<Character, Integer> charCount = new HashMap<>(); ?

You can write it like this:

HashMap<Character, Integer> charCount = new HashMap<>();

But using Map on the left-hand side is considered better practice because:

It makes the code more flexible.

Tomorrow, if you want to switch to another implementation (like LinkedHashMap or TreeMap), you don't need to change all your code. Example:

// Easy to change

Map<Character, Integer> charCount = new LinkedHashMap<>();

// Harder if you had HashMap on left side, since you must update everywhere

So, the general rule of thumb is:

Declare variables using the interface (Map)

Create objects using the implementation (HashMap, TreeMap, etc.)

✅ Example for clarity

```
Map<String, Integer> ages = new HashMap<>();
ages.put("Alice", 25);
ages.put("Bob", 30);
System.out.println(ages.get("Alice")); // 25
System.out.println(ages.get("Charlie")); // null (not found)
```

Here:

Keys = String (names)

Values = Integer (ages)

So "Alice" → 25, "Bob" → 30.

In your duplicate-character program:

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

means:

Each character in the string will be the key.

The count (number of times it appears) will be the value.

********&&&&&&&&&&&&&$$$$$$$$$$$##########@@@@@@@@@@@@@@@

```
import java.util.*;
class Main {
    public static void main(String[] args) {
   // to  find more than duplicates charcter in a given string(practice)
        Scanner scanner= new Scanner(System.in);
        System.out.println("enter the String :");
       String input=scanner.nextLine();
       input= input.toLowerCase().replaceAll(" ", "");
       Map<Character,Integer> charcount = new HashMap<>();
       for(char c: input.toCharArray()){
           charcount.put(c, charcount.getOrDefault(c,0)+1);
       }
       for(Map.Entry<Character,Integer> entry : charcount.entrySet() ){
           if (entry.getValue() > 1) {
               System.out.println(entry.getKey() + " → " + entry.getValue() + " times");
            }
       }


    }
}
```

```
o/p-->enter the String :
nisha prasad
a ? 3 times
s ? 2 times
```

if you want to count each of charchter of string then just remove the condition of "if" in above code.

Explaination:

input = "nishaprasad"

index: 0 1 2 3 4 5 6 7 8 9 10

chars: n i s h a p r a s a d

i	c	charcount.getOrDefault(c,0) (before)	new count (= +1)	charcount after this step (key:value pairs)
0	n	0	1	{n=1}
1	i	0	1	{n=1, i=1}
2	s	0	1	{n=1, i=1, s=1}
3	h	0	1	{n=1, i=1, s=1, h=1}
4	a	0	1	{n=1, i=1, s=1, h=1, a=1}
5	p	0	1	{n=1, i=1, s=1, h=1, a=1, p=1}
6	r	0	1	{n=1, i=1, s=1, h=1, a=1, p=1, r=1}
7	a	1	2	{n=1, i=1, s=1, h=1, a=2, p=1, r=1}

8	s	1	2	{n=1, i=1, s=2, h=1, a=2, p=1, r=1}
9	a	2	3	{n=1, i=1, s=2, h=1, a=3, p=1, r=1}
10	d	0	1	{n=1, i=1, s=2, h=1, a=3, p=1, r=1, d=1}

Final counts:

a = 3

s = 2

n, i, h, p, r, d = 1 each

Your print loop prints only entries whose value > 1, so you get:

bash

Copy code

a → 3 times

s → 2 times

Piece of code:

String[] words = input.trim().split("\\s+");

// Count words

int wordCount = (input.trim().isEmpty()) ? 0 : words.length;

1) input.trim()

What it does: returns a new String with leading and trailing whitespace removed. It does not remove spaces between words.

Why used: to avoid empty tokens at the start or end when splitting (e.g. " hello " → "hello").

Examples

" Hello " → "Hello"

"\t\n Java \n" → "Java"

" " → "" (empty string)

Note: since trim() returns a new string, the original input is not changed.

2) .split("\\s+")

What it does: split() splits the string into an array using a regular expression.

The Java string literal is "\\s+" which represents the regex \s+.

\s = any whitespace character (space, tab \t, newline \n, carriage return \r, form feed \f, etc.).

+ = one or more.

So \s+ matches one or more consecutive whitespace characters.

Why \\s+ (double backslash)? In a Java string you must escape the backslash, so "\\s+" becomes the regex \s+.

Effect: multiple spaces / tabs / newlines between words are treated as a single separator — you won't get empty tokens.

Examples

"Hello world".split("\\s+") → ["Hello", "world"]

"Hello world".split("\\s+") → ["Hello", "world"]

"one\ttwo\nthree".split("\\s+") → ["one","two","three"]

After trim(), " hi " becomes "hi", and splitting gives ["hi"].

3) input.trim().isEmpty()

What it does: checks whether the trimmed string has length 0.

Equivalent to: input.trim().length() == 0.

Purpose here: detect the case where the input contains only whitespace (or is empty) so you can return 0 words instead of 1 empty token or an incorrect count.

Examples

input = " " → input.trim() is "" → isEmpty() returns true.

input = "" → isEmpty() returns true.

input = " Hello " → trim() → "Hello" → isEmpty() returns false.

4) words.length vs String.length()

words is a String[] (array), so words.length is a field giving the number of elements in the array (no parentheses).

someString.length() is a method on String that returns the number of characters.

Example: if words = ["Hello","world"] then words.length == 2.

5) The ternary operator: (condition) ? valueIfTrue : valueIfFalse

In your code:

int wordCount = (input.trim().isEmpty()) ? 0 : words.length;

If the trimmed input is empty → wordCount = 0.

Otherwise → wordCount = words.length.

This prevents counting any words when the input is only spaces.

**&%$$$$$$$$$$$$$$$$$$$$&&&&&&&####@@@@@@@@

```
public class Main {

   public static void main(String[] args) {

      String word = "Hello";

      System.out.println(word.length()); // Output: 5

   }

}
```

word.length() → Used for Strings

In Java, String is a class.

To get the number of characters in a String, we use the method .length().

It returns an integer showing how many characters are in the string.

```
public class Main {

   public static void main(String[] args) {

      int[] numbers = {10, 20, 30, 40};

      System.out.println(numbers.length); // Output: 4

   }

}
```

word.length → Used for Arrays

length (without parentheses) is not a method, it is a final variable (a property) of arrays.

It gives the number of elements in an array.

```
import java.util.*;
class Main {
  public static void main(String[] args) {
  // to count the word of string
     Scanner scanner= new Scanner(System.in);
     System.out.println("enter the String :");
    String input=scanner.nextLine();
    String [] words = input.trim().split("\\s+");
    int wordcount = (input.trim().isEmpty())? 0 : words.length;
    System.out.println("The count of word String :"+wordcount);


  }
}
```

```
o/p-->enter the String :
nisha prasad is a good girl
The count of word String :6
```

&&&&&&&&%%%%%%%%%%%%%%%%%%

```
import java.util.*;
import java.util.stream.*;
public class PracticeSheet {

  // 1. Reverse a String
  public static String reverseString(String input) {
     return new StringBuilder(input).reverse().toString();
  }

  // 2. Check Palindrome
  public static boolean isPalindrome(String input) {
    String reversed = new StringBuilder(input).reverse().toString();
    return input.equalsIgnoreCase(reversed);
}

// 3. Find Duplicate Characters in a String
public static void findDuplicates(String input) {
    Map<Character, Long> freq = input.chars()
            .mapToObj(c -> (char) c)
            .collect(Collectors.groupingBy(c -> c, Collectors.counting()));
    freq.forEach((ch, count) -> {
        if (count > 1) {
            System.out.println(ch + " repeated " + count + " times");
        }
    });
}

// 4. Find Second Largest Number in Array
public static int secondLargest(int[] arr) {
    return Arrays.stream(arr)
            .boxed()
            .sorted(Comparator.reverseOrder())
            .distinct()
            .skip(1)
            .findFirst()
            .orElseThrow(() -> new RuntimeException("No second largest found"));
}

// 5. Find Missing Number in Array (1 to n)
public static int findMissing(int[] arr, int n) {
    int expectedSum = n * (n + 1) / 2;
    int actualSum = Arrays.stream(arr).sum();
    return expectedSum - actualSum;
}

// 6. Prime Check
public static boolean isPrime(int num) {
    if (num <= 1) return false;
    return IntStream.rangeClosed(2, (int) Math.sqrt(num))
            .allMatch(n -> num % n != 0);
}

// 7. Factorial using Recursion
public static long factorial(int n) {
    if (n == 0) return 1;
    return n * factorial(n - 1);
}

// 8. Java 8 Stream - Find Even Numbers
public static List<Integer> evenNumbers(List<Integer> numbers) {
    return numbers.stream()
            .filter(n -> n % 2 == 0)
            .collect(Collectors.toList());
}

// 9. Java 8 Stream - Find Max Number
public static int maxNumber(List<Integer> numbers) {
    return numbers.stream()
            .max(Integer::compareTo)
```

```
        .orElseThrow();
}

// 10. Java 8 Stream - Word Count
public static Map<String, Long> wordCount(List<String> words) {
    return words.stream()
            .collect(Collectors.groupingBy(w -> w, Collectors.counting()));
}

// 11. Group Employees by Department (Scenario-based)
static class Employee {
    String name;
    String department;
    double salary;

    Employee(String name, String dept, double salary) {
        this.name = name;
        this.department = dept;
        this.salary = salary;
    }

    @Override
    public String toString() {
        return name + " (" + department + ", " + salary + ")";
    }
}

public static Map<String, List<Employee>> groupByDept(List<Employee> employees) {
    return employees.stream()
```

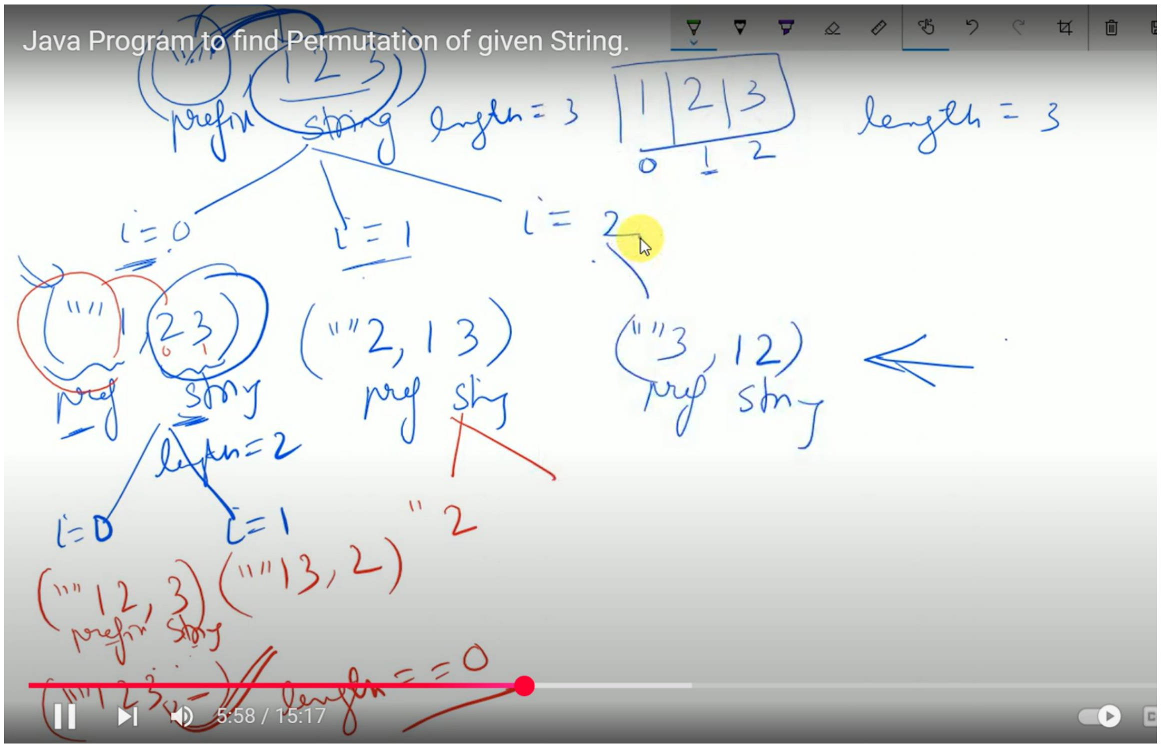
```
        .collect(Collectors.groupingBy(e -> e.department));
}


// Main method for testing
public static void main(String[] args) {
    // Test String
    System.out.println("Reverse: " + reverseString("hello"));
    System.out.println("Palindrome: " + isPalindrome("madam"));

    findDuplicates("programming");

    // Test Array
    int[] arr = {4, 2, 7, 7, 9, 1};
    System.out.println("Second Largest: " + secondLargest(arr));
    System.out.println("Missing Number (1-5): " + findMissing(new int[]{1, 2, 3, 5}, 5));

    // Numbers
    System.out.println("Is Prime(11): " + isPrime(11));
    System.out.println("Factorial(5): " + factorial(5));

    // Streams
    List<Integer> nums = Arrays.asList(1, 2, 3, 4, 5, 6);
    System.out.println("Even Numbers: " + evenNumbers(nums));
    System.out.println("Max Number: " + maxNumber(nums));

    List<String> words = Arrays.asList("apple", "banana", "apple", "orange");
    System.out.println("Word Count: " + wordCount(words));

    // Employees
```

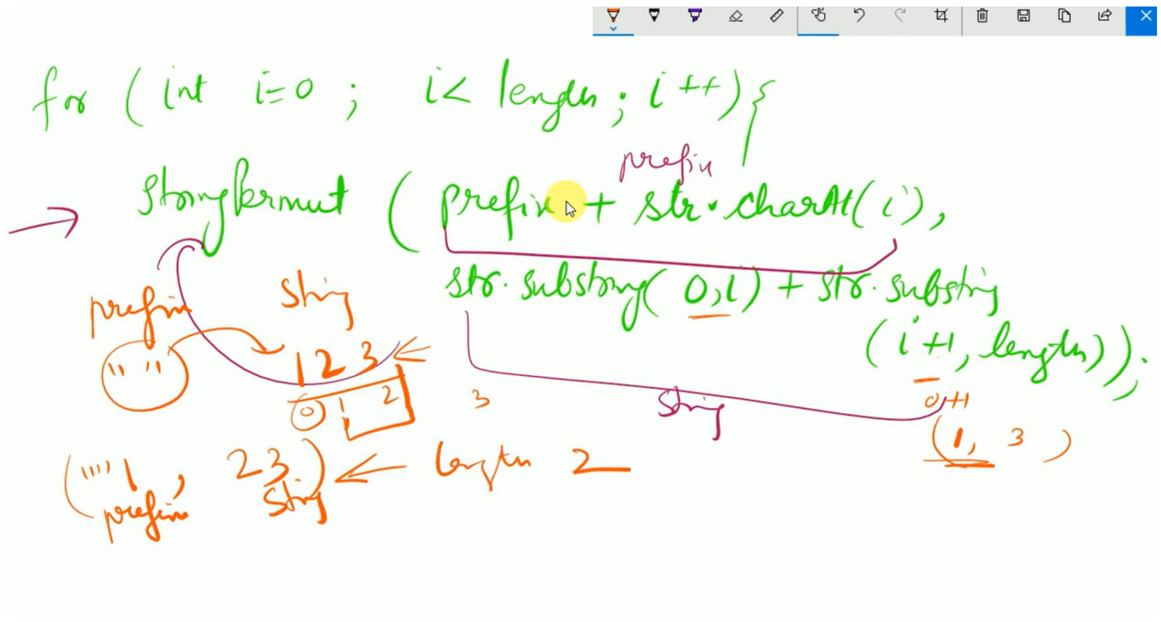
```
        List<Employee> employees = Arrays.asList(
                new Employee("Alice", "IT", 60000),
                new Employee("Bob", "HR", 50000),
                new Employee("Charlie", "IT", 70000)
        );
        System.out.println("Grouped by Dept: " + groupByDept(employees));
    }
}
++++++++++++++++))))))((((((((((**************&&&&&&&&&&&
import java.util.*;
class Main {
    public static void main(String[] args) {
        //palindrome for string
    String str="nin";
    System.out.println(isPalindrome(str));
    }
     static  boolean isPalindrome(String str){
        int start=0;
        int end =str.length()-1;
       while(start<end){
          if(str.charAt(start) != str.charAt(end)){
              return false;
          }
          start++;
          end--;
       }
       return true;
    }
}
```

and is i discard the "ana" and "nin" just add this below condtion

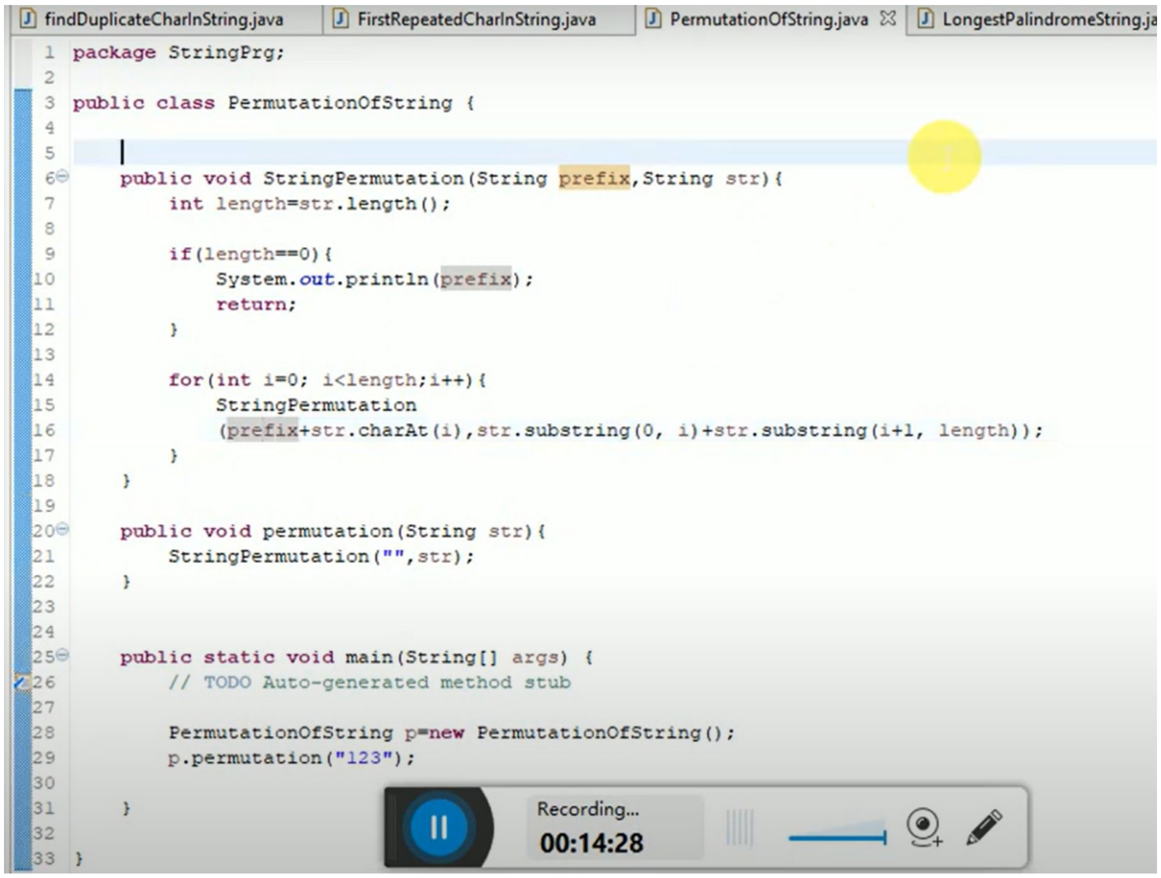
```
 if (str.length() % 2 != 0) {
     return false; // reject odd-length words like "ana", "nin"
  }
```

```
// Online Java Compiler
// Use this editor to write, compile and run your Java code online
import java.util.*;
class Main {
    public static void main(String[] args) {
        //all the permutation of given string
    String str="abc";
    permute("",str);
    }
    public static void  permute(String prefix, String str){
        int len=str.length();
        if(len == 0){
            System.out.println(prefix);
            return ;
        }
        for(int i=0;i<len;i++){
            permute(prefix+str.charAt(i),str.substring(0,i)+str.substring(i+1,len));


        }



    }
}
```


Java Program to find Permutation of given String.



```
for (int i=0; i< length; i++){
    StringPermut( prefix + str.charAt(i),
        str.substring(0,i) + str.substring
            (i+1, length));
```


findDuplicateCharInString.java
FirstRepeatedCharInString.java
PermutationOfString.java
LongestPalindromeString.ja
Recording...
00:14:28

```java
package StringPrg;

public class PermutationOfString {

    
    public void StringPermutation(String prefix,String str){
        int length=str.length();

        if(length==0){
            System.out.println(prefix);
            return;
        }

        for(int i=0; i<length;i++){
            StringPermutation
            (prefix+str.charAt(i),str.substring(0, i)+str.substring(i+1, length));
        }
    }

    public void permutation(String str){
        StringPermutation("",str);
    }


    public static void main(String[] args) {
        // TODO Auto-generated method stub

        PermutationOfString p=new PermutationOfString();
        p.permutation("123");

    }
}
```

```java
import java.util.*;
class Main {
  public static void main(String[] args) {
    // to check given strings are anagram or not
    String str1="listen";
    String str2="slient";
    if(isAnagram(str1,str2)){
      System.out.println("Both the strings are anagram :)");
    }
    else{
      System.out.println("Both the strings are not anagram :(");
    }
    }
    public static boolean isAnagram(String str1, String str2){
        str1=str1.replaceAll("\\s","").toLowerCase();
        str2=str2.replaceAll("\\s","").toLowerCase();
        if(str1.length() != str2.length()){
            return false;
        }
        char[] arr1=str1.toCharArray();
        char[] arr2=str2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        return Arrays.equals(arr1,arr2);
    }
}
```

Java Code Using Stream API(infosys question)

```
import java.util.stream.Collectors;
String str = "hello";

String reversed =
    java.util.stream.IntStream.range(0, str.length())  // 0,1,2,3,4
        .mapToObj(i -> str.charAt(str.length() - 1 - i)) // pick from end
        .map(String::valueOf)
        .collect(Collectors.joining());

System.out.println(reversed); // olleh
```

Explanation

Step 1: IntStream.range(0, str.length())

What it does: Creates an IntStream of numbers starting from 0 (inclusive) up to str.length() (exclusive).

For "hello", str.length() = 5.

So IntStream.range(0, 5) produces:

0, 1, 2, 3, 4

Type: IntStream (a stream of primitive int values, not objects).

Internally:

IntStream.range(start, end) basically loops from start to end-1 and pushes each int into the stream pipeline. No array is created — the numbers are generated on demand.

Step 2: .mapToObj(i -> str.charAt(str.length() - 1 - i))

What .mapToObj does:

It takes each int from the IntStream and maps it to an object (returns a Stream<T>).

Think: "for every number i, give me some object".

The lambda here:

i -> str.charAt(str.length() - 1 - i)

Input: i (from the IntStream)

Output: char (a character from the string)

Java autoboxes the char to Character object, so the new stream is a Stream<Character>.

Dry run:

For i = 0 → str.charAt(5 - 1 - 0) = str.charAt(4) = 'o'

For i = 1 → str.charAt(5 - 1 - 1) = str.charAt(3) = 'l'

For i = 2 → str.charAt(5 - 1 - 2) = str.charAt(2) = 'l'

For i = 3 → str.charAt(5 - 1 - 3) = str.charAt(1) = 'e'

For i = 4 → str.charAt(5 - 1 - 4) = str.charAt(0) = 'h'

So after .mapToObj(...) →

Stream contains: ['o','l','l','e','h']

Step 3: .map(String::valueOf)

Converts each Character into a String.

['o','l','l','e','h'] → ["o","l","l","e","h"].

Step 4: .collect(Collectors.joining())

Concatenates all strings in the stream into one string.

["o","l","l","e","h"] → "olleh"

Why not just .mapToObj(...) → .collect?

Because IntStream is a primitive stream of int. You can't directly join ints or chars into a string.

So:

.mapToObj(...) turns numbers → characters (objects).

.map(String::valueOf) turns characters → strings.

.collect(joining()) merges them.

Internal Flow Recap

Generate sequence of indices [0,1,2,3,4] using IntStream.range.

Lazy, values created only when needed.

For each index i, calculate reverse index str.length()-1-i and get that character.

Produces stream of characters in reverse order.

Convert each Character to String.

Join into "olleh".

//Stream use

```
import java.util.*;

public class Streamclass {

   public static void main(String[] args) {

      List<Integer> num= Arrays.asList(9,7,1,8,4,7,2,6);

      num.stream().filter(n ->n%2==0).map(n->n*2).forEach(n->System.out.println(n));


   }

}

// Online Java Compiler

// Use this editor to write, compile and run your Java code online

import java.util.*;

import java.util.stream.Collectors;
```

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

//to count number of vowels and consanant in a given string

```
         Scanner scanner= new Scanner(System.in);
        System.out.println("enter the String :");
       String name=scanner.nextLine();
       String str= name.replaceAll(" ","").toLowerCase();
       int vowels=0;
       int constant=0;
       for(char ch : str.toCharArray()){
          if(ch>='a' && ch<='z'){
             if(ch == 'a' ||ch == 'e' || ch == 'i'|| ch == 'o'|| ch == 'u' ){
                vowels++;
             }
             else{
                constant++;
             }
          }
       }
        System.out.println("the vowel are :"+vowels);
        System.out.println("teh contant are :"+constant);


   }
}
```

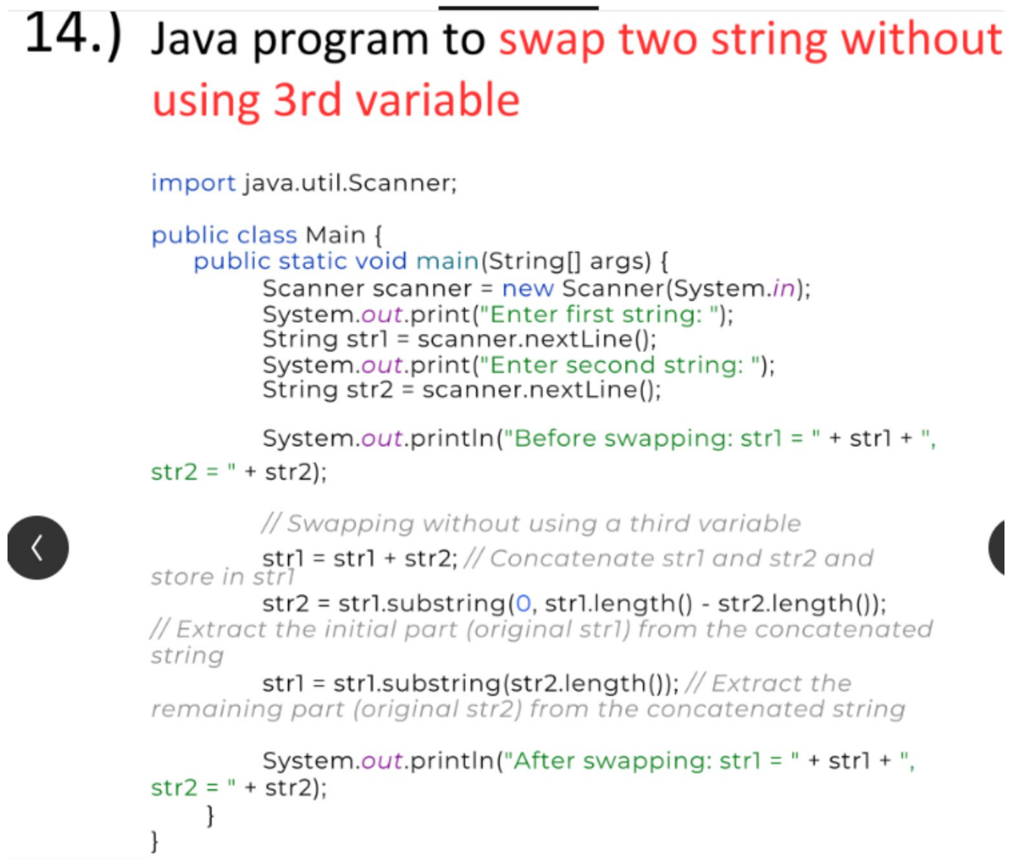
```
enter the String :
nisha prasad
the vowel are :4
teh contant are :7
```

//to print unqiue charactar

```
import java.util.*;
class Main {
    public static void main(String[] args) {
          Scanner scanner= new Scanner(System.in);
        System.out.println("enter the String :");
       String name=scanner.nextLine();
       Printuniquechar(name);
    }
    public static void Printuniquechar(String str){
        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.println(ch+"");
            }
        }
    }
}
```

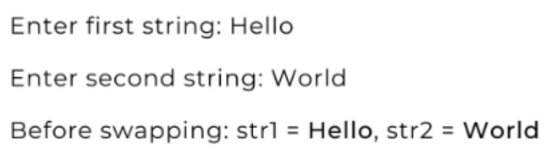
```
enter the String :
nishaprasad
n
i
s
h
a
p
r
```

d



14.) Java program to swap two string without using 3rd variable

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter first string: ");
            String str1 = scanner.nextLine();
            System.out.print("Enter second string: ");
            String str2 = scanner.nextLine();

            System.out.println("Before swapping: str1 = " + str1 + ",
str2 = " + str2);

            // Swapping without using a third variable
            str1 = str1 + str2; // Concatenate str1 and str2 and
store in str1
            str2 = str1.substring(0, str1.length() - str2.length());
// Extract the initial part (original str1) from the concatenated
string
            str1 = str1.substring(str2.length()); // Extract the
remaining part (original str2) from the concatenated string

            System.out.println("After swapping: str1 = " + str1 + ",
str2 = " + str2);
     }
}
```



```
Enter first string: Hello

Enter second string: World

Before swapping: str1 = Hello, str2 = World
```

//separate the lower and upper case

```
import java.util.*;
class Main {
   public static void main(String[] args) {
            System.out.println("Enter the string : ");
      Scanner scanner =new Scanner(System.in);
      String str=scanner.nextLine();
      StringBuilder lowercase= new StringBuilder();
      StringBuilder uppercase= new StringBuilder();
```

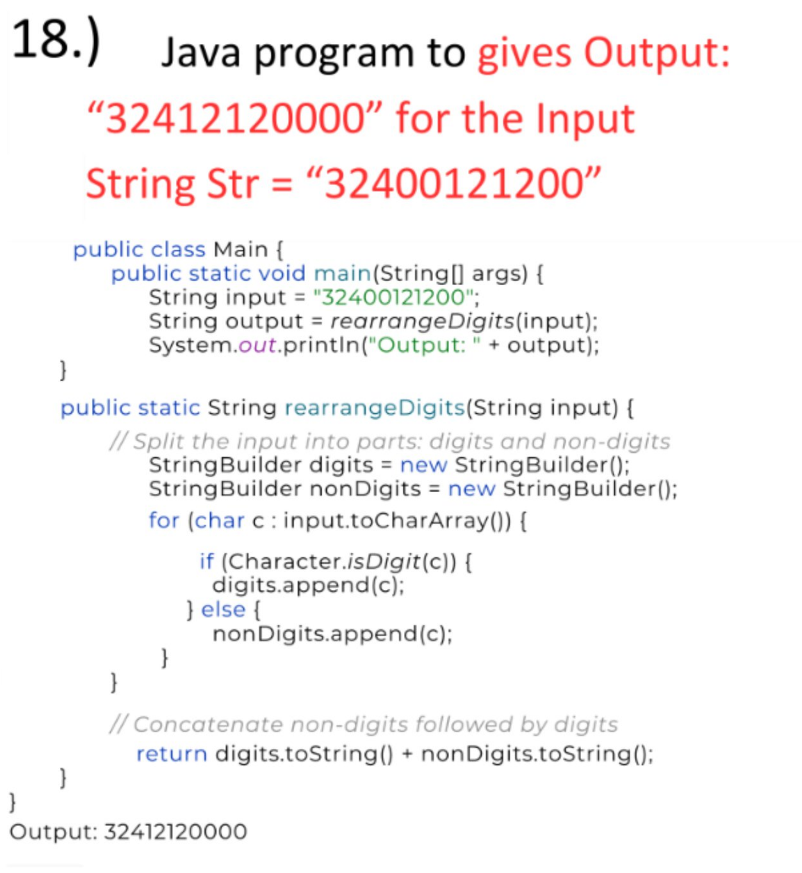
```
        for(char ch :str.toCharArray()){
        if(Character.isLowerCase(ch)){
           lowercase.append(ch);
        }
        else{
           uppercase.append(ch);
        }
        }
        System.out.println("Lowercase characters: " + lowercase.toString());
        System.out.println("Uppercase characters: " + uppercase.toString());
    }
}
```

```
Enter the string :
NhsjdyLKJAFHHjksdfgjhgkdj
Lowercase characters: hsjdyjksdfgjhgkdj
Uppercase characters: NLKJAFHH
```

// separate the number and letter from the string

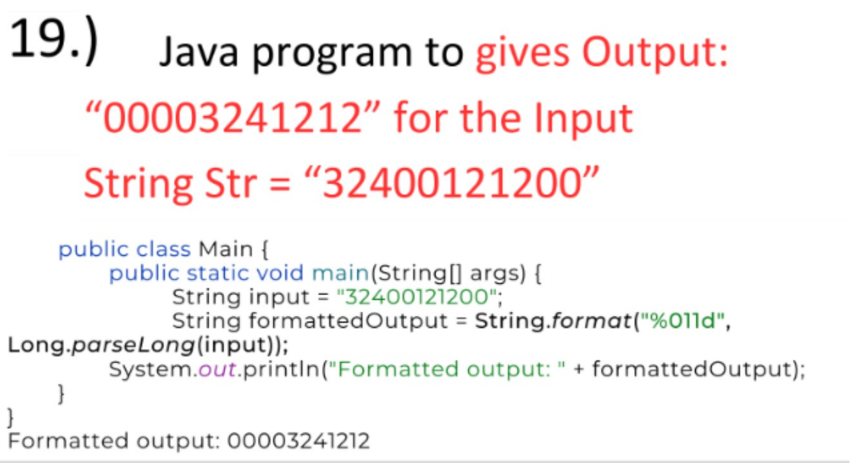
```
import java.util.*;
class Main {
    public static void main(String[] args) {
        //separate the lower and upper case
        System.out.println("Enter the string : ");
        Scanner scanner =new Scanner(System.in);
        String str=scanner.nextLine();
        StringBuilder letter= new StringBuilder();
        StringBuilder number= new StringBuilder();
        for(char ch :str.toCharArray()){
        if(Character.isLetter(ch)){
           letter.append(ch);
        }
        else if(Character.isDigit(ch)){
          number.append(ch);
        }
        }
        System.out.println("Characters: " + letter.toString());
        System.out.println("Number: " + number.toString());
    }
}
```

```
Enter the string :
skdjhf876aksdjhfk87382iykeiw83iweu
Characters: skdjhfaksdjhfkiykeiwiweu
Number: 8768738283
```



18.) Java program to gives Output: "32412120000" for the Input String Str = "32400121200"

```
public class Main {
    public static void main(String[] args) {
        String input = "32400121200";
        String output = rearrangeDigits(input);
        System.out.println("Output: " + output);
}
public static String rearrangeDigits(String input) {
    // Split the input into parts: digits and non-digits
        StringBuilder digits = new StringBuilder();
        StringBuilder nonDigits = new StringBuilder();
        for (char c : input.toCharArray()) {
            if (Character.isDigit(c)) {
              digits.append(c);
            } else {
              nonDigits.append(c);
            }
        }
    // Concatenate non-digits followed by digits
        return digits.toString() + nonDigits.toString();
    }
}
Output: 32412120000
```

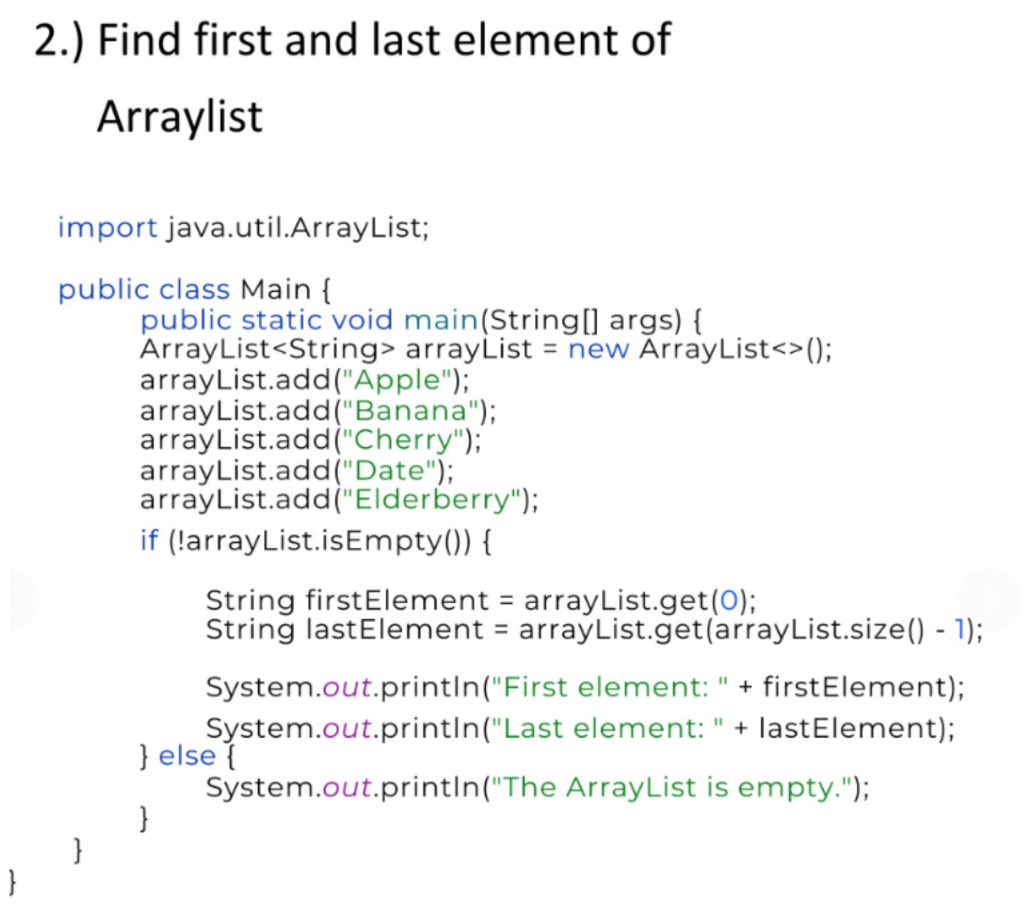


19.) Java program to gives Output:
“00003241212” for the Input
String Str = “32400121200”

```
public class Main {
        public static void main(String[] args) {
                String input = "32400121200";
                String formattedOutput = String.format("%011d",
Long.parseLong(input));
        System.out.println("Formatted output: " + formattedOutput);
    }
}
Formatted output: 00003241212
```

//find the common element between two array in java

```
import java.util.*;
class Main {
  public static void main(String[] args) {
    int[] arr1 = {2,8,3,6,4,9};
    int[] arr2 ={9,1,6,3} ;
    Set<Integer> set1 = new HashSet<>();
    Set<Integer> common = new HashSet<>();
    for(int num : arr1){
      set1.add(num);
    }
    for(int num: arr2){
      if(set1.contains(num)){
        common.add(num);
      }
    }
    System.out.println(common);
  }
}
[3, 6, 9]
```



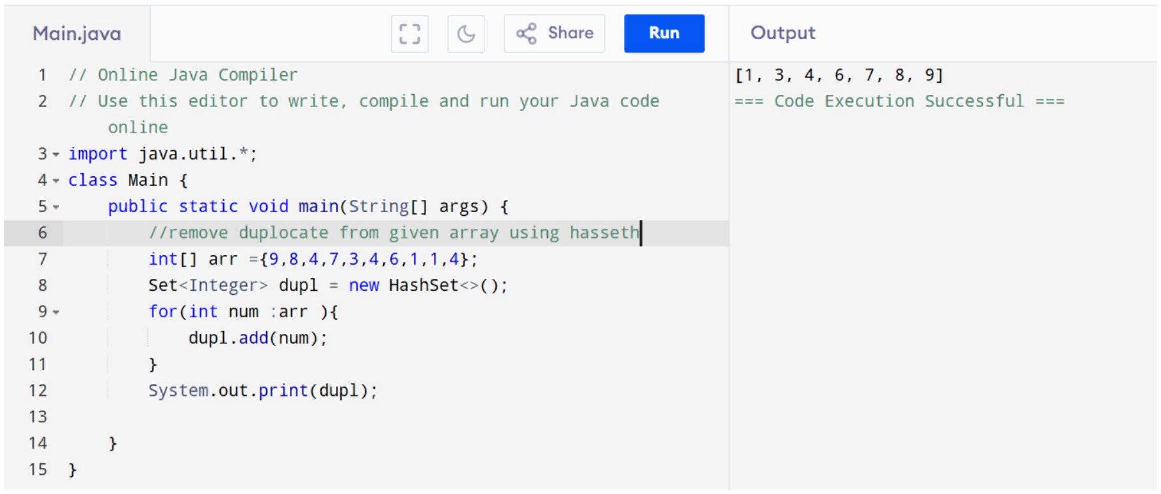
2.) Find first and last element of Arraylist

```java
import java.util.ArrayList;

public class Main {
        public static void main(String[] args) {
        ArrayList<String> arrayList = new ArrayList<>();
        arrayList.add("Apple");
        arrayList.add("Banana");
        arrayList.add("Cherry");
        arrayList.add("Date");
        arrayList.add("Elderberry");
        if (!arrayList.isEmpty()) {

            String firstElement = arrayList.get(0);
            String lastElement = arrayList.get(arrayList.size() - 1);

            System.out.println("First element: " + firstElement);
            System.out.println("Last element: " + lastElement);
        } else {
            System.out.println("The ArrayList is empty.");
        }
    }
}
```

//sort a array without inbuilt method in java

```java
import java.util.*;
class Main {
  public static void main(String[] args) {
          int[] arr ={9,8,4,7,3,2,6,1,0,4};
     Sortedrray(arr);
     for(int num: arr){
        System.out.println(num+" ");
     }
  }
  public static void Sortedrray(int[] arr){
     for(int i=0;i<arr.length-1;i++){
        int min=i;
```

```
            for(int j=i+1;j<arr.length;j++){
                if(arr[j]<arr[min]){
                    min=j;
                }
                }
            int temp=arr[i];
            arr[i]=arr[min];
            arr[min]=temp;
        }
    }
}
```



```
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online
import java.util.*;
class Main {
    public static void main(String[] args) {
        //remove duplocate from given array using hasseth
        int[] arr ={9,8,4,7,3,4,6,1,1,4};
        Set<Integer> dupl = new HashSet<>();
        for(int num :arr ){
            dupl.add(num);
        }
        System.out.print(dupl);

    }
}
```

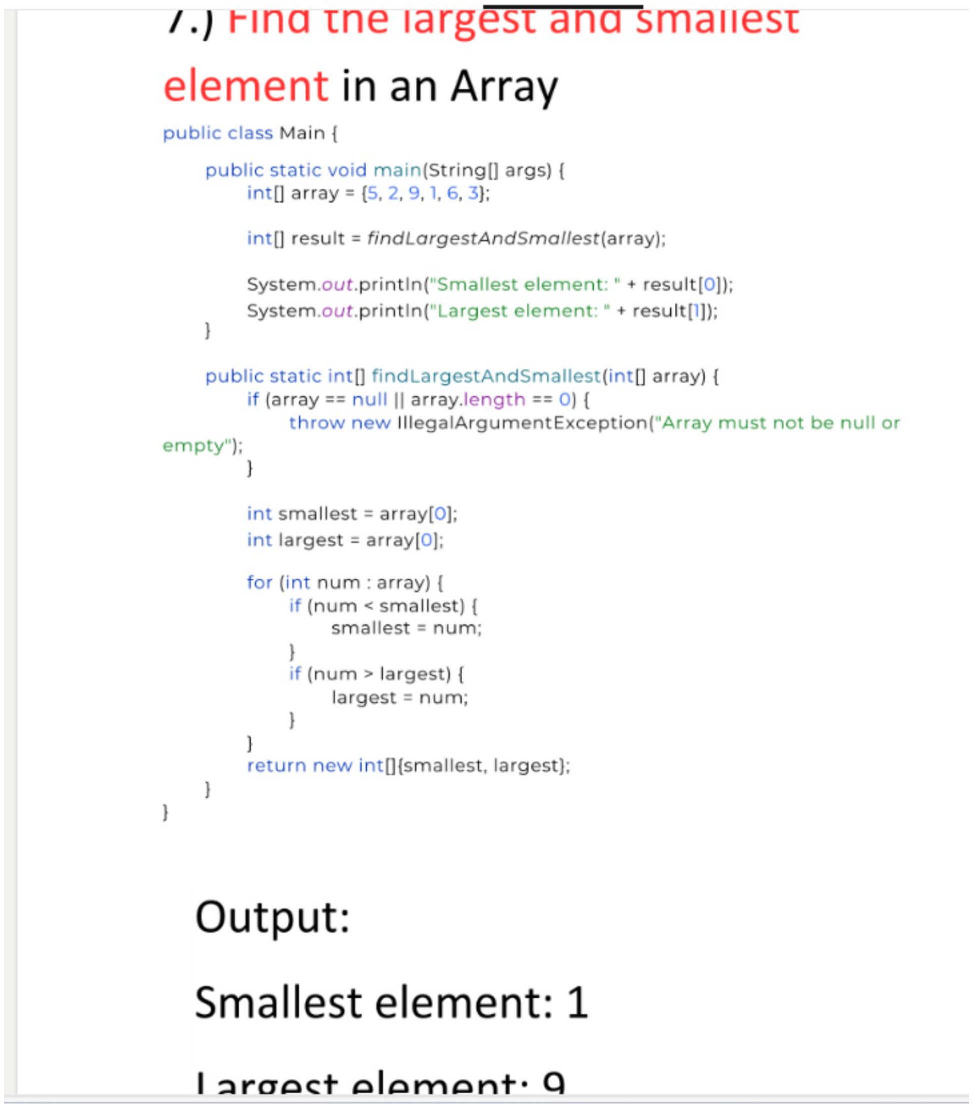
```
[1, 3, 4, 6, 7, 8, 9]
=== Code Execution Successful ===
```

//to search the index place of a given array

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        int[] arr ={9,8,7,3,6,1,4};
        int target=8;
      int index= searchthetarget(arr, target);
      if(index!= -1){
         System.out.println("the search taget index is at :"+index);
      }
      else{
          System.out.println("No index found :");
      }
    }
    public static int searchthetarget(int[] arr,int target){
       for(int i=0;i<arr.length;i++){
       if(arr[i]== target){
           return i;
       }


       }
              return -1;
    }
}
```

outpu:

```
the search taget index is at :1
```



7.) Find the largest and smallest element in an Array

```
public class Main {

    public static void main(String[] args) {
        int[] array = {5, 2, 9, 1, 6, 3};

        int[] result = findLargestAndSmallest(array);

        System.out.println("Smallest element: " + result[0]);
        System.out.println("Largest element: " + result[1]);
    }

    public static int[] findLargestAndSmallest(int[] array) {
        if (array == null || array.length == 0) {
            throw new IllegalArgumentException("Array must not be null or
empty");
        }

        int smallest = array[0];
        int largest = array[0];

        for (int num : array) {
            if (num < smallest) {
                smallest = num;
            }
            if (num > largest) {
                largest = num;
            }
        }
        return new int[]{smallest, largest};
    }
}
```

Output:

```
Smallest element: 1

Largest element: 9
```

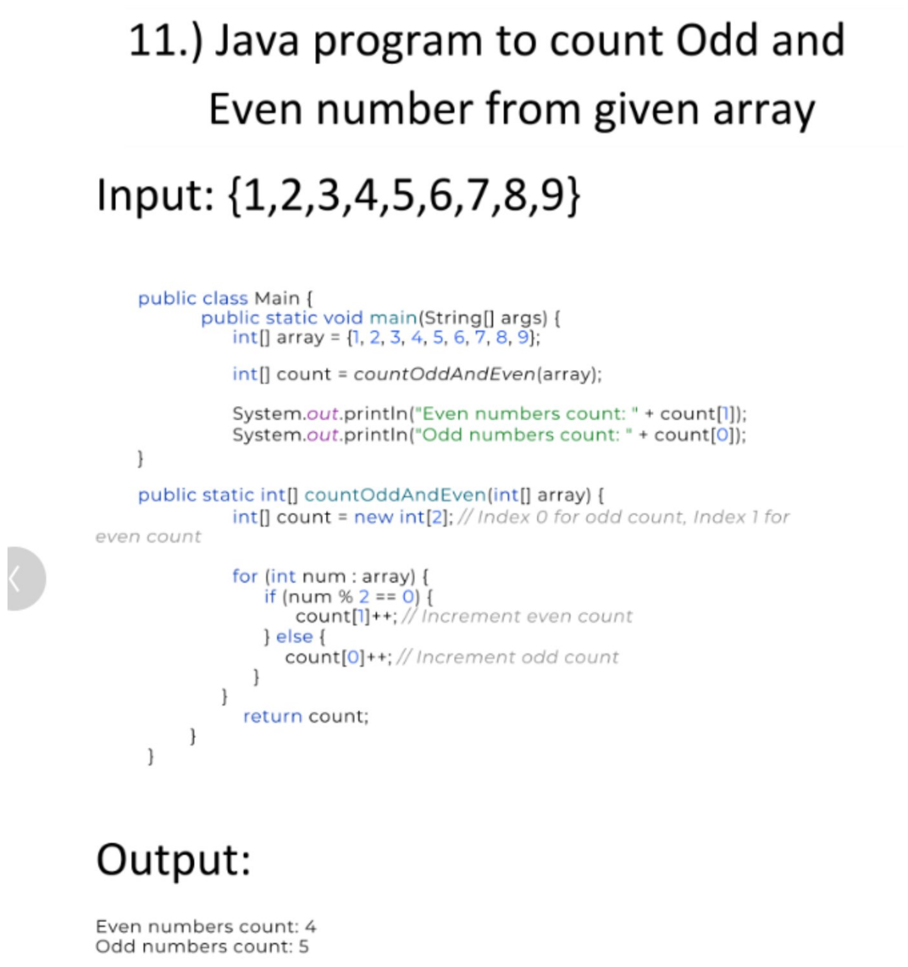
//addition of a array and ignore the other elements

```
import java.util.*;

class Main {

  public static void main(String[] args) {

    String[] arr ={"9","8","7","@","6","#","4"};

    int sum= addition(arr);

      System.out.println("the added number is  : "+sum);

  }

  public static int addition(String[] array){

    int sum=0;
```

```
        for(String str: array){
           try{
           int num = Integer.parseInt(str);
           sum+=num;
           }catch(NumberFormatException e){
        }
        }
        return sum;
    }
}
```

output:

```
the added number is  : 34
```



11.) Java program to count Odd and Even number from given array

Input: {1,2,3,4,5,6,7,8,9}

```
public class Main {
        public static void main(String[] args) {
            int[] array = {1, 2, 3, 4, 5, 6, 7, 8, 9};

            int[] count = countOddAndEven(array);

            System.out.println("Even numbers count: " + count[1]);
            System.out.println("Odd numbers count: " + count[0]);
}

public static int[] countOddAndEven(int[] array) {
            int[] count = new int[2]; // Index 0 for odd count, Index 1 for
even count

            for (int num : array) {
                if (num % 2 == 0) {
                    count[1]++; // Increment even count
                } else {
                   count[0]++; // Increment odd count
               }
            }
              return count;
        }
  }
```

Output:

```
Even numbers count: 4
Odd numbers count: 5
```

//find the non-repeative array and display in shorted array

```
import java.util.*;
class Main {
  public static void main(String[] args) {
      int[] arr ={1,1,2,2,3,2,4,5,8,9,7,3,4,5,3};
    List<Integer> uniques= findTheuniquies(arr);
      System.out.println(uniques);
  }
  public static List<Integer> findTheuniquies(int[] array){
    Map<Integer,Integer> count =new HashMap<>();
    for(int n: array){
      count.put(n,count.getOrDefault(n,0)+1);


    }
    List<Integer> result=new ArrayList<>();
    for(Map.Entry<Integer,Integer> e: count.entrySet()){
      if(e.getValue()==1){
        result.add(e.getKey());
      }
    }
    Collections.sort(result);
    return  result;
}
}
```

```
output:[7, 8, 9]
```

Explaintion of the above code:

```
for (int n : arr)
```

This means:

"Take each number n in the array one by one."

So, the loop will run like this:

First time → n = 1

Second time → n = 1 (again)

Third time → n = 2

… until the end.

Inside the loop:

```
count.put(n, count.getOrDefault(n, 0) + 1);
```

This looks scary, but let’s unpack:

```
count.getOrDefault(n, 0)
```

Try to get the value stored for key n.

If the key does not exist yet, return 0 instead.

Example:

First time we see 1: count is empty → no key 1.
→ getOrDefault(1, 0) gives 0.

Then we do 0 + 1 = 1.

We store count.put(1, 1).

So now the map is {1=1}.

Next time we see 1

Now count already has key 1 with value 1.

getOrDefault(1, 0) returns 1.

Add 1 → 1 + 1 = 2.

Store count.put(1, 2).

Now the map is {1=2}.

When we see 2 (first time)

No key 2 yet.

getOrDefault(2, 0) → returns 0.

0 + 1 = 1.

Store count.put(2, 1).

Now map is {1=2, 2=1}.

When we see 2 again

Key 2 already exists with value 1.

getOrDefault(2, 0) → returns 1.

1 + 1 = 2.

Store count.put(2, 2).

Now map is {1=2, 2=2}.

Full dry run with your array

Array = [1,1,2,2,3,4,5,5,6,6]

Step	n	Before map	getOrDefault(n,0)	New value	After map
1	1	{}	0	1	{1=1}
2	1	{1=1}	1	2	{1=2}
3	2	{1=2}	0	1	{1=2, 2=1}
4	2	{1=2,2=1}	1	2	{1=2, 2=2}
5	3	{1=2,2=2}	0	1	{1=2,2=2,3=1}
6	4	{...}	0	1	{1=2,2=2,3=1,4=1}
7	5	{...}	0	1	{1=2,2=2,3=1,4=1,5=1}
8	5	{...}	1	2	{1=2,2=2,3=1,4=1,5=2}
9	6	{...}	0	1	{1=2,2=2,3=1,4=1,5=2,6=1}
10	6	{...}	1	2	{1=2,2=2,3=1,4=1,5=2,6=2}

✅ Final count map:

{1=2, 2=2, 3=1, 4=1, 5=2, 6=2}

This means:

1 appears 2 times

2 appears 2 times

3 appears 1 time

4 appears 1 time

5 appears 2 times

6 appears 2 times

So that single line:

```
count.put(n, count.getOrDefault(n, 0) + 1);
```

is just a short way of saying:

If n already exists, increase its count by 1.

If n doesn’t exist, put it in the map with count = 1.

```
List<Integer> result = new ArrayList<>();
```

```
for (Map.Entry<Integer, Integer> e : count.entrySet()) {
    if (e.getValue() == 1) {
        result.add(e.getKey());
    }
}
```

Step 1: List<Integer> result = new ArrayList<>();

We are creating a new empty list named result.

This list will store all numbers that appear only once in the array.

Example: At the beginning,

result = []

Step 2: for (Map.Entry<Integer, Integer> e : count.entrySet())

count is our map, for example:

{1=2, 2=2, 3=1, 4=1, 5=2, 6=2}

count.entrySet() → means all key-value pairs of the map.
Example entries:

(1=2)

(2=2)

(3=1)

(4=1)

(5=2)

(6=2)

Map.Entry<Integer, Integer> e → each e is one of those key-value pairs.

So the loop will go like this:

First → e = (1=2)

Second → e = (2=2)

Third → e = (3=1)

... and so on.

Step 3: if (e.getValue() == 1)

e.getKey() → gives the number (e.g., 1, 2, 3, ...).

e.getValue() → gives the count (how many times it appeared).

We check:
"If this number appears exactly once (== 1) ..."

Step 4: result.add(e.getKey());

If the condition is true (value is 1), we add the key (the number itself) into the result list.

Full Dry Run with example

Map = {1=2, 2=2, 3=1, 4=1, 5=2, 6=2}

Step	e (key=value)	e.getValue()	Condition (==1)	Action (add key)	result
1	(1=2)	2	false	nothing	[]
2	(2=2)	2	false	nothing	[]
3	(3=1)	1	true	add 3	[3]
4	(4=1)	1	true	add 4	[3,4]
5	(5=2)	2	false	nothing	[3,4]
6	(6=2)	2	false	nothing	[3,4]

✅ Final result = [3, 4]

MCQs on Java pseudocode

DSA Problems:

- First non-repeated character in a string
- Candle problem (n = unburnt candles, k = burnt candles needed to make one new candle)

SQL Query: Hotel table → order records by rating DESC (rating ≥ 7)

Round 2: Technical Interview

DSA Problem: Remove duplicates from a sorted array → e.g. {1,2,2,3,3,4} → {1,2,3,4,null,null}

Core Java & OOPs: Encapsulation, Collections framework usage, HashMap internal working, Array vs ArrayList

Hashing: equals() & hashCode() contract with custom objects in HashSet

Multithreading: Synchronization & custom problem → Print numbers up to N using 3 threads in sequence

SQL: Group students by name, calculate total marks, and sort by total marks

Round 3: Client + HR Discussion

Introduction, Experience, Current & Expected CTC, Work location

Technical:

- Time complexity of HashMap.get()
- Check if LinkedList is palindrome
- WeakHashMap explanation
- equals() & hashCode() contract when using a class as a key in Map
- Project discussion

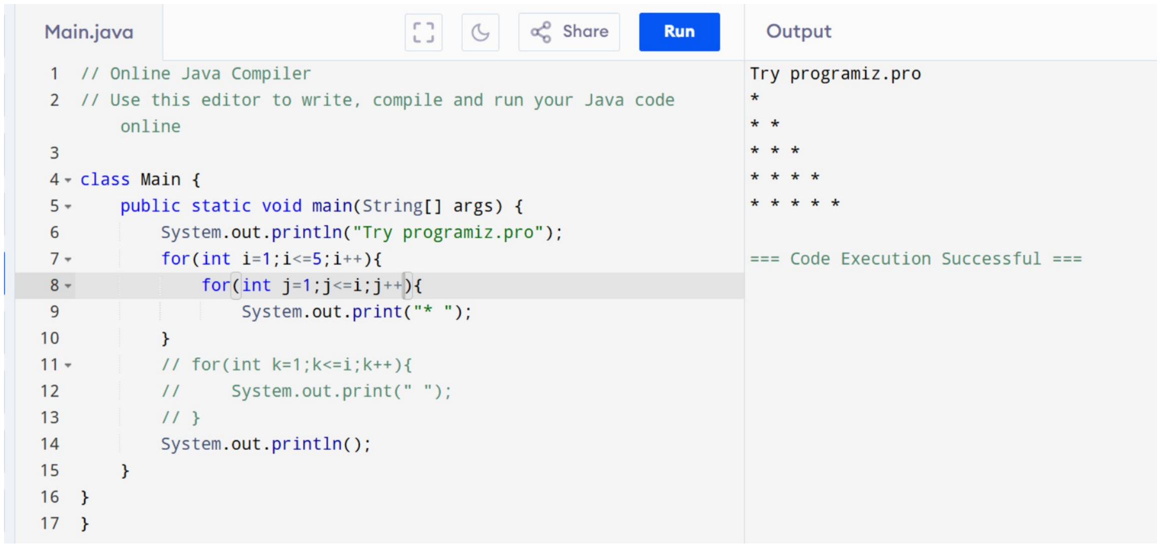
Q. Write code to filter out loans with an incomplete status using Java 8 features.

Ans. Code to filter out loans with incomplete status using Java 8 features.

Use stream() method to convert the list of loans into a stream

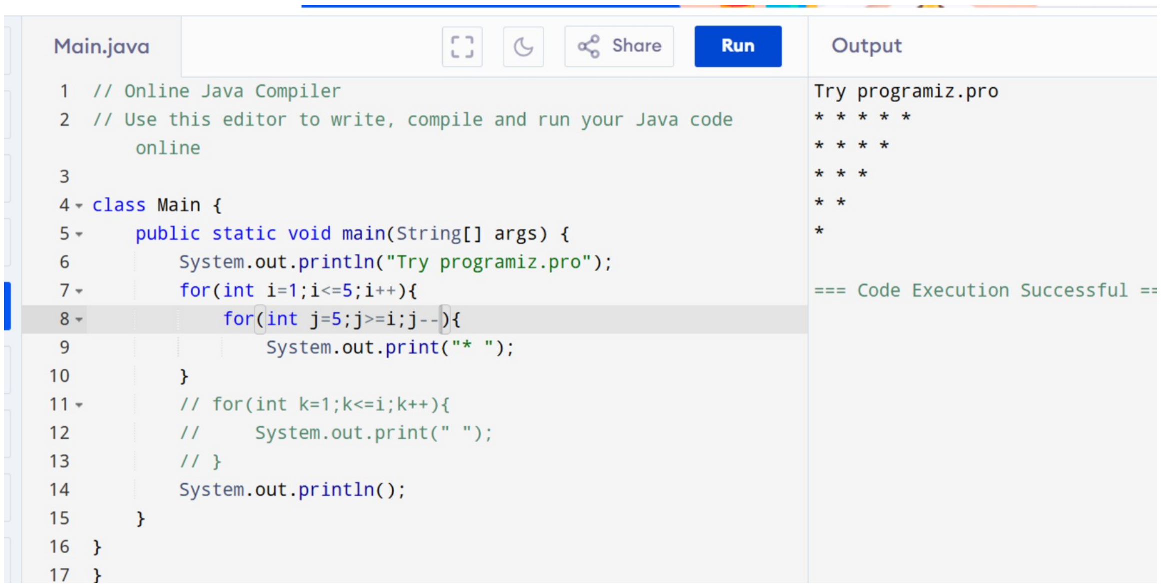
Use filter() method to filter out loans with incomplete status

Use collect() method to collect the filtered loans into a new list



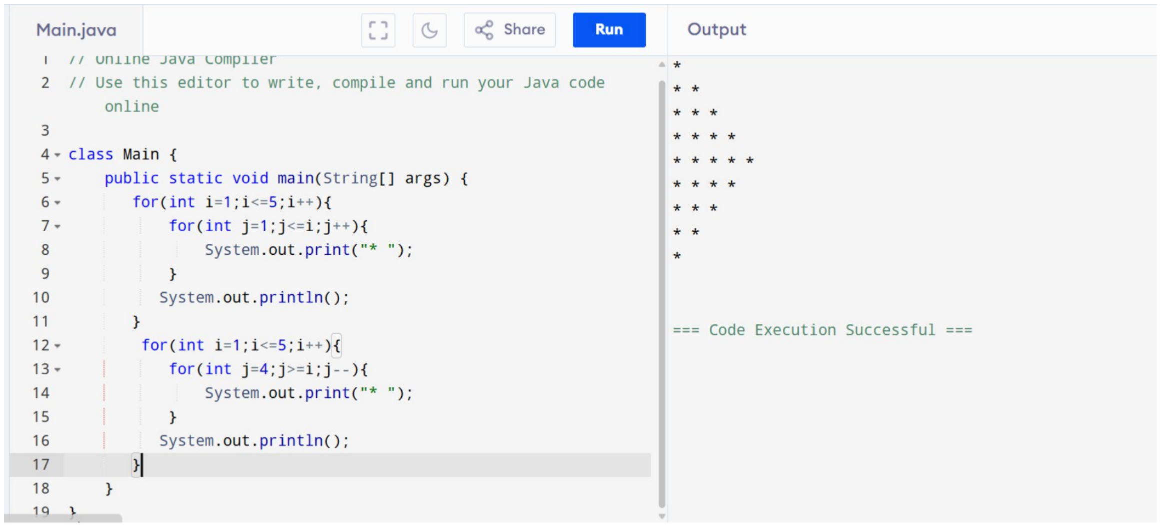
```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        System.out.println("Try programiz.pro");
        for(int i=1;i<=5;i++){
            for(int j=1;j<=i;j++){
                System.out.print("* ");
        }
        // for(int k=1;k<=i;k++){
        //      System.out.print(" ");
        // }
        System.out.println();
    }
}
}
```

```
Try programiz.pro
*
* *
* * *
* * * *
* * * * *

=== Code Execution Successful ===
```



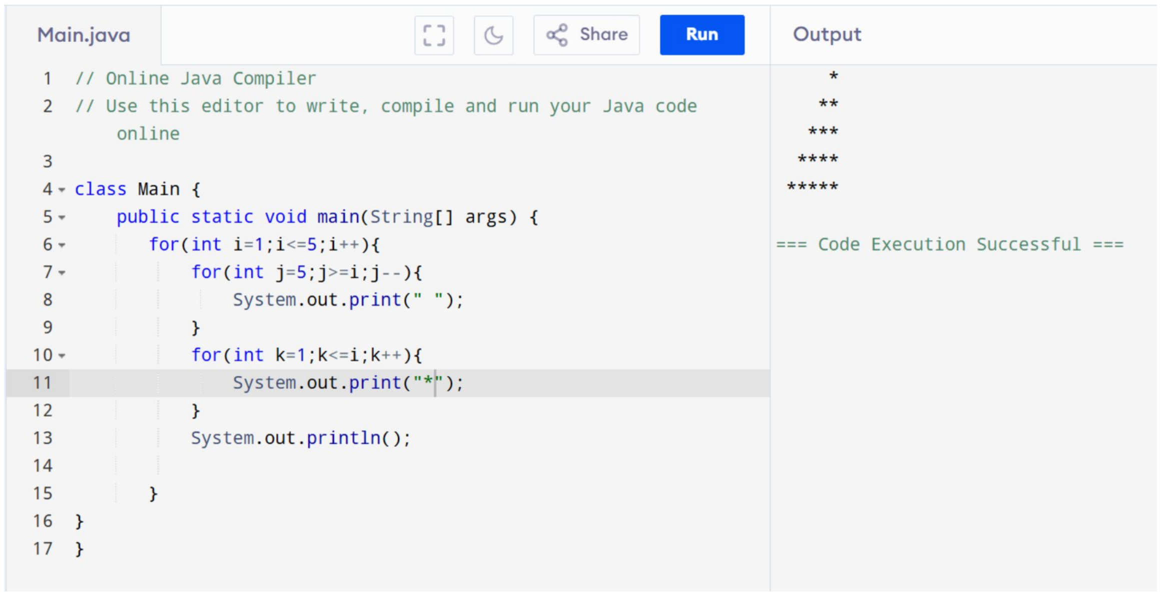
```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        System.out.println("Try programiz.pro");
        for(int i=1;i<=5;i++){
            for(int j=5;j>=i;j--){
                System.out.print("* ");
        }
        // for(int k=1;k<=i;k++){
        //     System.out.print(" ");
        // }
        System.out.println();
    }
}
}
```

```
Try programiz.pro
* * * * *
* * * *
* * *
* *
*


=== Code Execution Successful ==
```



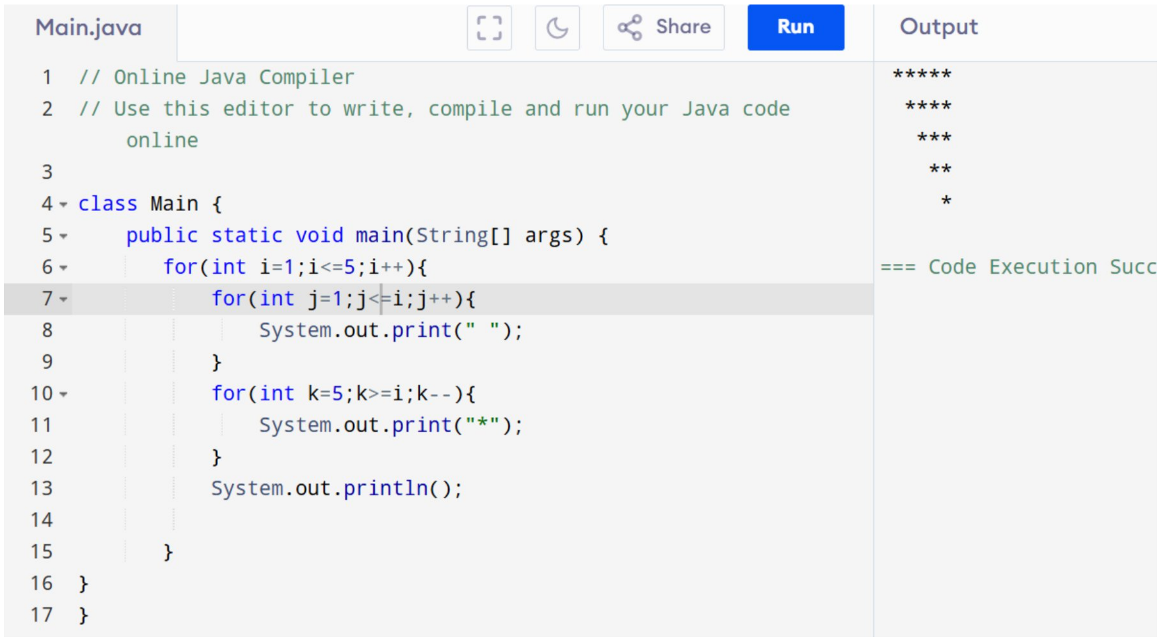
```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=1;j<=i;j++){
                System.out.print("* ");
            }
            System.out.println();
        }
        for(int i=1;i<=5;i++){
            for(int j=4;j>=i;j--){
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*



=== Code Execution Successful ===
```



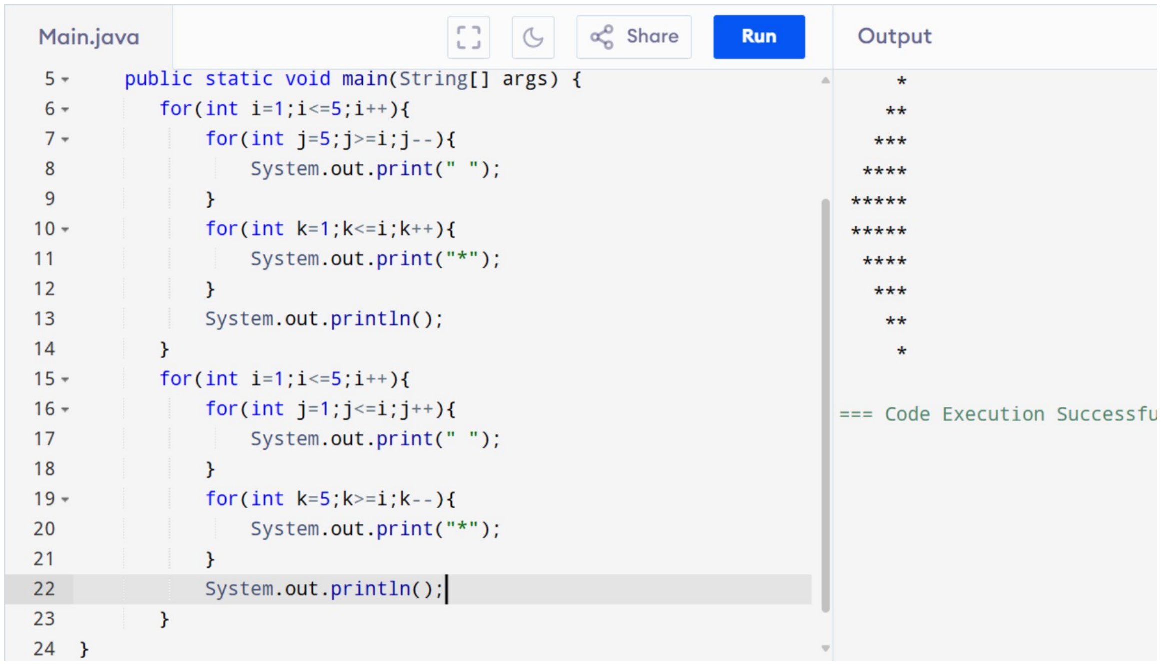
```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=5;j>=i;j--){
                System.out.print(" ");
            }
            for(int k=1;k<=i;k++){
                System.out.print("*");
            }
            System.out.println();

        }
}
}
```

```
     *
    **
   ***
  ****
 *****


=== Code Execution Successful ===
```



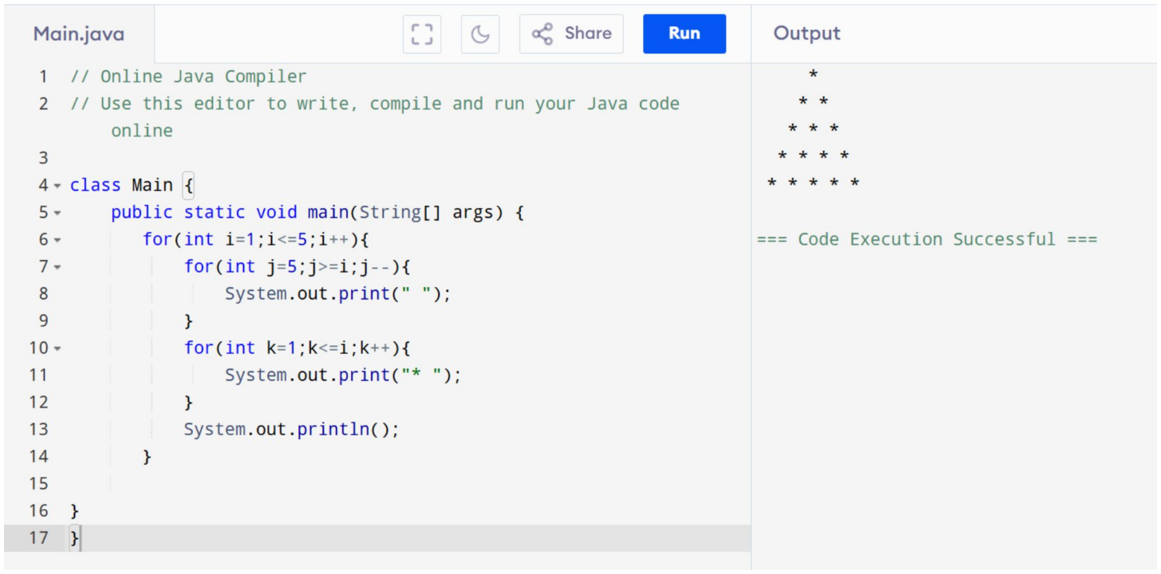
```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=1;j<=i;j++){
                System.out.print(" ");
            }
            for(int k=5;k>=i;k--){
                System.out.print("*");
            }
            System.out.println();

        }
    }
}
```

```
*****
 ****
  ***
   **
    *

=== Code Execution Succ
```



```java
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=5;j>=i;j--){
                System.out.print(" ");
            }
            for(int k=1;k<=i;k++){
                System.out.print("*");
            }
            System.out.println();
        }
        for(int i=1;i<=5;i++){
            for(int j=1;j<=i;j++){
                System.out.print(" ");
            }
            for(int k=5;k>=i;k--){
                System.out.print("*");
            }
            System.out.println();
        }
}
```

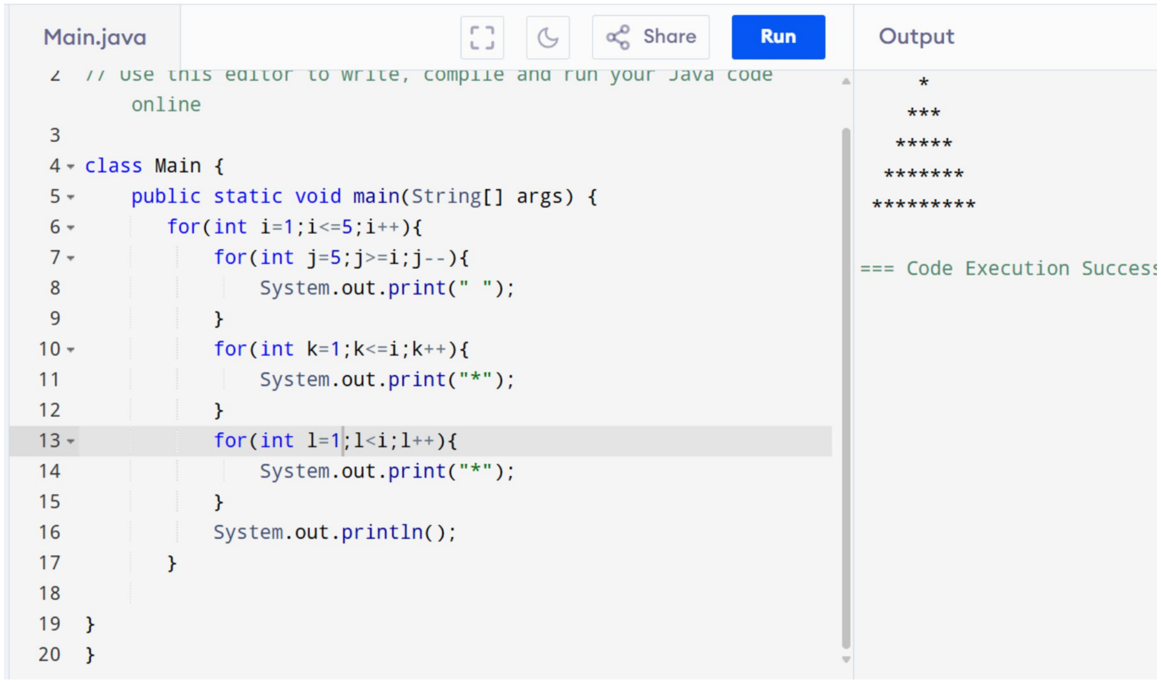
```
    *
   **
  ***
 ****
*****
*****
 ****
  ***
   **
    *

=== Code Execution Successfu
```



Main.java Share Run Output

```java
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=5;j>=i;j--){
                System.out.print(" ");
            }
            for(int k=1;k<=i;k++){
                System.out.print("* ");
            }
            System.out.println();
        }

}
}
```

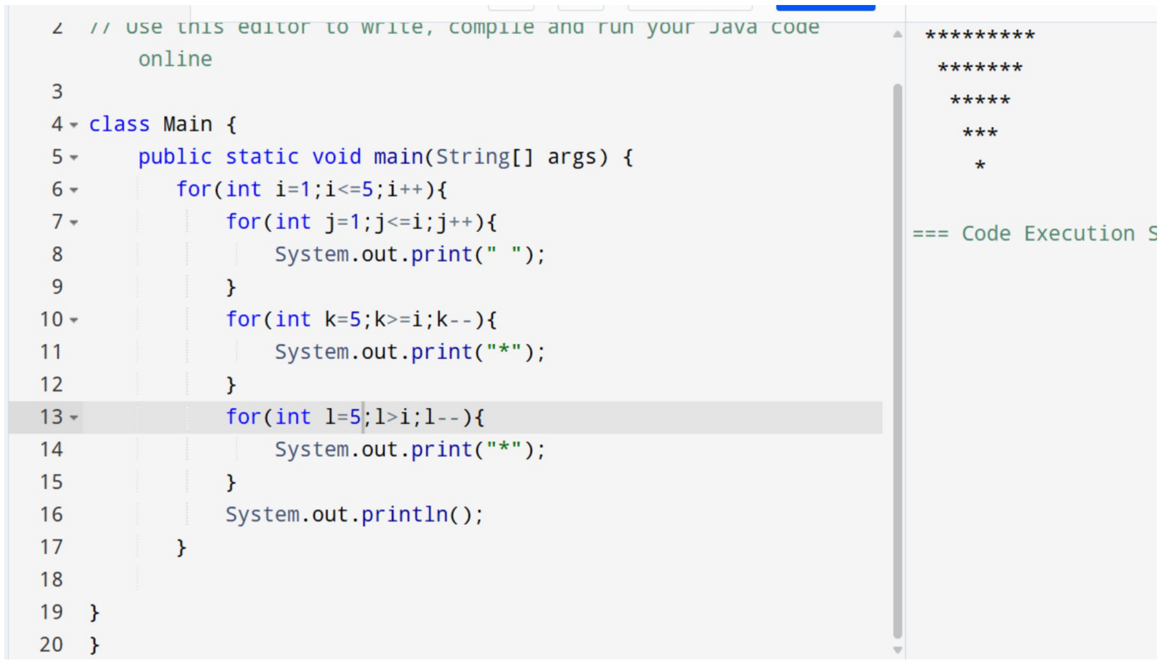
```
     *
    * *
   * * *
  * * * *
 * * * * *


=== Code Execution Successful ===
```



Main.java Share Run Output

```java
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++){
            for(int j=5;j>=i;j--){
                System.out.print(" ");
            }
            for(int k=1;k<=i;k++){
                System.out.print("*");
            }
            for(int l=1;l<i;l++){
                System.out.print("*");
            }
            System.out.println();
        }

}
}
```

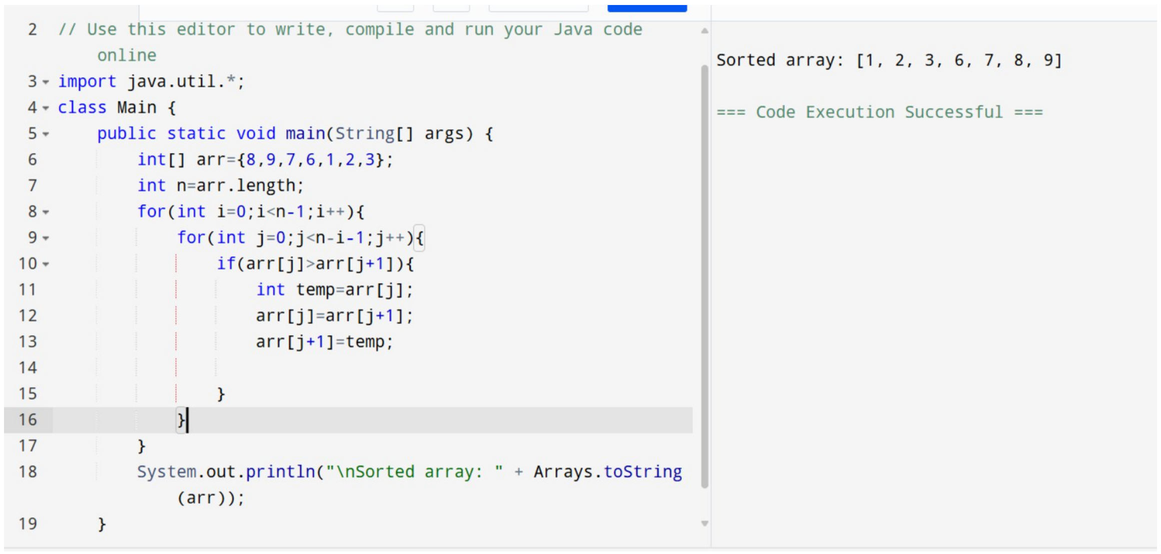
```
     *
    ***
   *****
  *******
 *********


=== Code Execution Success
```



```
// use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
       for(int i=1;i<=5;i++){
           for(int j=1;j<=i;j++){
               System.out.print(" ");
           }
           for(int k=5;k>=i;k--){
               System.out.print("*");
           }
           for(int l=5;l>i;l--){
               System.out.print("*");
           }
           System.out.println();
       }

}
}
```

```
*********
 *******
  *****
   ***
    *


=== Code Execution S
```

```
import java.util.*;


class Main {
  public static void main(String[] args) {
    List<Integer> list1 = Arrays.asList(1, 2, 3, 4);
    List<Integer> list2 = Arrays.asList(3, 4, 5, 6);


    List<Integer> result = new ArrayList<>();


    for (Integer num : list1) {
      if (list2.contains(num)) {
        result.add(num);
      }
    }


    System.out.println(result);
  }
}
```



```
// Use this editor to write, compile and run your Java code
    online
import java.util.*;
class Main {
    public static void main(String[] args) {
        int[] arr={8,9,7,6,1,2,3};
        int n=arr.length;
        for(int i=0;i<n-1;i++){
            for(int j=0;j<n-i-1;j++){
                if(arr[j]>arr[j+1]){
                    int temp=arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;

                }
            }
        }
        System.out.println("\nSorted array: " + Arrays.toString
            (arr));
    }
```

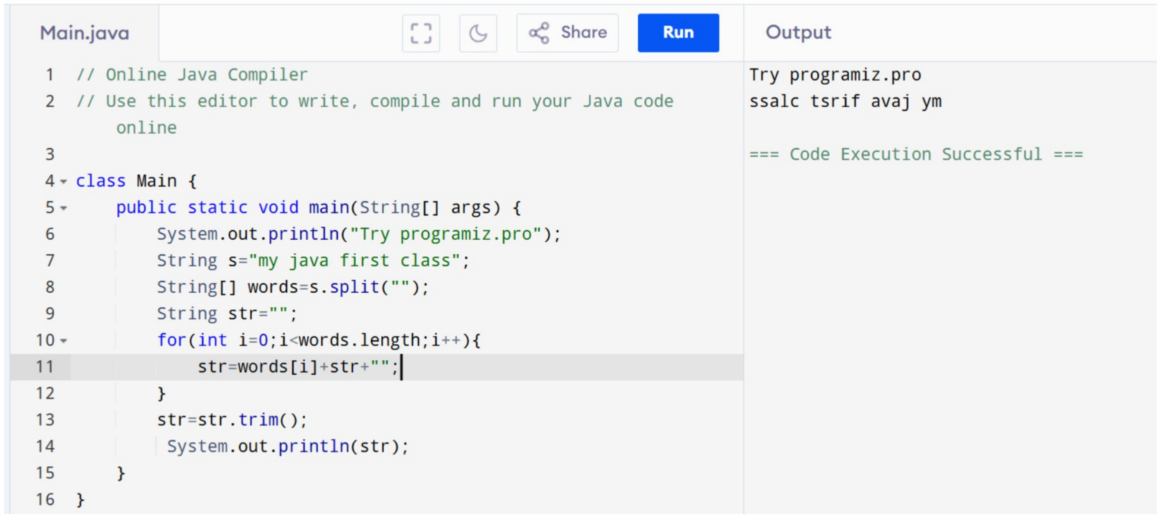
```
Sorted array: [1, 2, 3, 6, 7, 8, 9]

=== Code Execution Successful ===
```

```
public class ReverseWords {
  public static void main(String[] args) {
    String s = "my java first class";

    // Step 1: Split the string into words
    String[] words = s.split(" ");

    // Step 2: Reverse the order of words
    String reversed = "";
    for (int i = words.length - 1; i >= 0; i--) {
      reversed += words[i] + " ";
    }

    // Step 3: Trim the extra space at the end
    reversed = reversed.trim();
```

```
        // Step 4: Print output
        System.out.println(reversed);
    }
}
```

OUTPUT: Input: "my java first class"

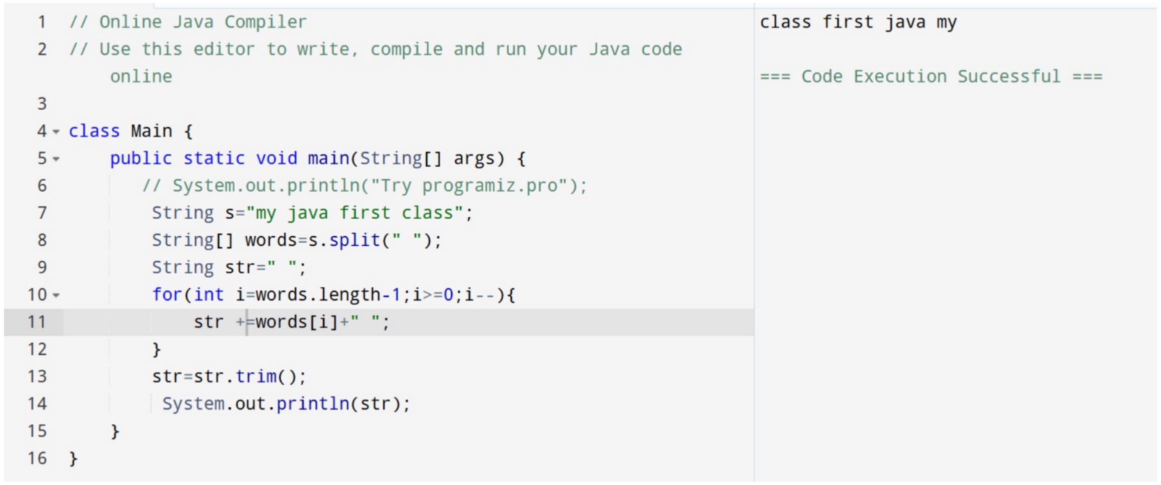
Output: "class first java my"



```
// Online Java Compiler
// Use this editor to write, compile and run your Java code online

class Main {
    public static void main(String[] args) {
        System.out.println("Try programiz.pro");
        String s="my java first class";
        String[] words=s.split("");
        String str="";
        for(int i=0;i<words.length;i++){
            str=words[i]+str+"";
        }
        str=str.trim();
         System.out.println(str);
    }
}
```

```
Try programiz.pro
ssalc tsrif avaj ym

=== Code Execution Successful ===
```

```
public class ReverseWords {

    public static String reverseEachWord(String sentence) {
        String[] words = sentence.split(" ");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            StringBuilder reversedWord = new StringBuilder(word);
            result.append(reversedWord.reverse()).append(" ");
        }

        return result.toString().trim();
    }
    public static void main(String[] args) {
        String input = "Java is powerful";
        System.out.println(reverseEachWord(input));
    }
}
```



```
// Online Java Compiler
// Use this editor to write, compile and run your Java code
    online

class Main {
    public static void main(String[] args) {
       // System.out.println("Try programiz.pro");
        String s="my java first class";
        String[] words=s.split(" ");
        String str=" ";
        for(int i=words.length-1;i>=0;i--){
            str +=words[i]+" ";
        }
        str=str.trim();
         System.out.println(str);
    }
}
```

```
class first java my

=== Code Execution Successful ===
```

Remove duplicate characters keeping order

```
public String removeDuplicateChars(String s) {
    if (s == null) return null;
    StringBuilder sb = new StringBuilder();
    Set<Character> seen = new HashSet<>();
    for (char c : s.toCharArray()) {
        if (!seen.contains(c)) {
            seen.add(c);
            sb.append(c);
        }
    }
    return sb.toString();
}
```

Move zeros to end of array

```
public void moveZerosToEnd(int[] arr) {
    if (arr == null) return;
    int index = 0;
    for (int num : arr) {
        if (num != 0) {
            arr[index++] = num;
        }
    }
    while (index < arr.length) {
        arr[index++] = 0;
    }
}
```

Check prime number

```
public boolean isPrime(int n) {
    if (n <= 1) return false;
    if (n == 2) return true;
    if (n % 2 == 0) return false;
    for (int i = 3; i * i <= n; i += 2) {
        if (n % i == 0) return false;
    }
    return true;
}
```

GCD of two numbers

```
public int gcd(int a, int b) {
    while (b != 0) {
        int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}
```

8. Merge two sorted arrays

```
public int[] mergeSorted(int[] a, int[] b) {
    int i = 0, j = 0, k = 0;
    int[] res = new int[a.length + b.length];
    while (i < a.length && j < b.length) {
        if (a[i] <= b[j]) {
            res[k++] = a[i++];
        } else {
            res[k++] = b[j++];
        }
    }
    while (i < a.length) res[k++] = a[i++];
    while (j < b.length) res[k++] = b[j++];
    return res;
}
```

Find second largest element in an array

```
public Integer secondLargest(int[] arr) {
    if (arr == null || arr.length < 2) return null;

    // 1. Find largest
    int largest = arr[0];
    for (int num : arr) {
        if (num > largest) {
            largest = num;
        }
    }

    // 2. Find second largest
    Integer secondLargest = null;
    for (int num : arr) {
        if (num != largest) {
            if (secondLargest == null || num > secondLargest) {
                secondLargest = num;
            }
        }
    }

    return secondLargest;
}
```

///Write a Java program to check whether two strings are rotations of each other.

Example:

"ABCD" and "CDAB" → true

```
public static boolean isRotation(String s1, String s2) {
    // Step 1: length check
    if (s1.length() != s2.length()) {
        return false;
    }

    // Step 2: concatenate s1 with itself
    String temp = s1 + s1;
    // Step 3: check if s2 is substring
    return temp.contains(s2);
}
```

///Write a Java program to find the first non-repeating character in a String.

Example:

Input: "swiss"

Output: 'w'

```
public static char firstNonRepeating(String s) {
    Map<Character, Integer> count = new HashMap<>();

    for (char c : s.toCharArray()) {
        count.put(c, count.getOrDefault(c, 0) + 1);
    }

    for (char c : s.toCharArray()) {
        if (count.get(c) == 1) {
            return c;
        }
    }
    return '\0';
}
```

///Longest Substring Without Repeating Characters

```
import java.util.HashSet;
import java.util.Set;
```

```
public class LongestSubstring {

    public static int lengthOfLongestSubstring(String s) {

        Set<Character> set = new HashSet<>();
        int left = 0, right = 0;
        int maxLength = 0;

        while (right < s.length()) {

            // If character not present, add and expand window
            if (!set.contains(s.charAt(right))) {
                set.add(s.charAt(right));
                maxLength = Math.max(maxLength, right - left + 1);
                right++;
            }
            // If duplicate found, shrink window from left
            else {
                set.remove(s.charAt(left));
                left++;
            }
        }
        return maxLength;
    }

    public static void main(String[] args) {
        System.out.println(lengthOfLongestSubstring("abcabcbb")); // 3
    }
}
```