IMP JAVA Programs for QA/SDET Interview

1.) Java program to Find Odd or Even number

```java
import java.util.Scanner;

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

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter any number: ");
        int number = scanner.nextInt();

        if (number % 2 == 0) {
            System.out.println(number + " is even.");
        } else {
            System.out.println(number + " is odd.");
        }
    }
}
```

2.) Java program to find Prime number

```java
import java.util.Scanner;

public class PrimeNumber {

        public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        if (isPrime(number)) {
            System.out.println(number + " is a prime number.");
        } else {
            System.out.println(number + " is not a prime number.");
        }
    }

public static boolean isPrime(int num) {
        for (int i = 2; i <= num / 2; i++) {
        //try each number by using %
            if (num % i == 0) {
                return false;
            }
        }
                return true;
    }
```

3.) Java program to find Fibonacci series upto a given number range

```java
import java.util.Scanner;

public class PrimeNumber {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("enter number of terms");
        int number = 6;
        int first = 0, second = 1, next;
        System.out.println("Fibonacci series is ");
        for ( int i = 0; i<=number; i++)
          {
             System.out.println(first + "");
             next = second+first;
             first = second;
             second = next;
          }
    }
}
```

Output: 0 1 1 2 3 5 8

4.) Java program to swap two numbers without using third variable

```java
import java.util.Scanner;

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

    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the first number: ");
    int a = 5,
    System.out.print("Enter the second number: ");
    int b = 10;
    System.out.println("Before swapping: a = " + a + ", b = " + b);

    a = a + b;
    b = a - b;
    a = a - b;
    System.out.println("After swapping: a = " + a + ", b = " + b);

    }
}
```

Output: After Swapping: a = 10 , b = 5

5.) Java program to Find Factorial on given Number

```java
import java.util.Scanner;

public class FactorialNumber {

    public static void main(String[] args) {
    int factorial =1;
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter any number ");
    int number = 5;

    for (int i = 1; i <= number; i++){
     factorial = factorial * i;
    }
    System.out.println("Factorial number is :" +factorial);

    }
}
```

```
Input: 5!
Output: 5! = 5*4*3*2*1 = 120
```

6.) Java program to Reverse Number

```java
import java.util.Scanner;

public class ReverseNumber {

    public static void main(String[] args) {
       int no, rev=0,r,a;
       Scanner scanner = new Scanner(System.in);
       System.out.println("Enter any number : ");
       no = scanner.nextInt();
       a = no;
       while(no>0)
       {
           r =  no%10;
           rev = rev*10+r;
           no=no/10;
       }

       System.out.println("Reverse : " +rev);

    }
}
```

```
Input: 15786
Output: 68751
```

7.) Java program to find Armstrong Number

```java
import java.util.Scanner;
public class ArmstrongNumber {

    public static void main(String[] args) {
    int arm=0, a,b,c,d,no;
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter any number : ");
    no = scanner.nextInt();
    d = no;
    while(no>0)
    {
        a =  no%10;
        no = no/10;
        arm =arm+a*a*a;
    }
    if(arm==d){
    System.out.println("Armstrong number");
    }
    else{
    System.out.println("Not Armstrong number");
    }
    }
}
```

8.) Java program to find number of digits in given number

```java
import java.util.Scanner;
public class NumberOfDigits {

     public static void main(String[] args) {
    int no = 0, a = 0;
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter any number : ");
    no = scanner.nextInt();
    if(no<0)
    {
        no = no * -1;

    } else if (no==0) {
        no=1;
    }
    while(no>0)
    {
        no=no/10;
        a++;}
    System.out.println("Number of digits in given number is :" +a); }
```

9.) Java program to find Palindrome number

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        if (isPalindrome(number)) {
            System.out.println(number + " is a palindrome.");
        } else {
            System.out.println(number + " is not a palindrome.");
        }
    }

    public static boolean isPalindrome(int num) {
        int originalNumber = num;
        int reversedNumber = 0;

        while (num != 0) {
            int digit = num % 10;
            reversedNumber = reversedNumber * 10 + digit;
            num =  num/10;
        }

        return originalNumber == reversedNumber;
    }
}
```

Enter a number: 1001

1001 is a palindrome.

10.) Java program to calculate the sum of digits of a number

```java
public class Main {
    public static void main(String[] args) {
        int number = 12345;

        int sumOfDigits = calculateSumOfDigits(number);

        System.out.println("Sum of digits of " + number + " is: " +
sumOfDigits);
    }

    public static int calculateSumOfDigits(int number) {
        int sum = 0;
        while (number > 0) {
            int digit = number % 10; // Extract the last digit
            sum = sum + digit; // Add the digit to sum
            number = number / 10; // Remove the last digit from number
        }
        return sum;
    }
}
```

Output:

Sum of digits of 12345 is: 15

Strings

1.) Java program to reverse a string

```java
import java.util.Scanner;
public class Test {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        char ch;
        String nstr = "";
        for (int i = 0; i < input.length(); i++) {
            ch = input.charAt(i);
            nstr = ch + nstr;
        }
    System.out.println("Reversed String is : " + nstr);
```

2.) Java program to reverse each word of a given string

```java
public static void main(String[] args) {
    reverseEachWordOfString("Java is good programming langauges");
}

static void reverseEachWordOfString(String inputString)
{
    String[] words = inputString.split(" ");

    String reverseString = "";
    for (int i = 0; i < words.length; i++) {
            String word = words[i];
            String nstr = "";
            char ch;
            for (int j = 0; j < word.length(); j++) {
                    ch = word.charAt(j);
                    nstr = ch + nstr;
            }
    reverseString = reverseString + nstr + " ";
}

    System.out.println(inputString);
    System.out.println(reverseString);
}

Input:  Java is good programming langauges
Output: avaJ si doog gnimmargorp seguagnal
```

3.) Java program to find duplicate characters in a string

```java
import java.util.HashMap;
import java.util.Set;

public class Main {

    public static void main(String[] args) {
        duplicateCharacterCount("Learn Java Programming");
    }

    static void duplicateCharacterCount(String inputString) {

        HashMap<Character, Integer> charCountMap = new HashMap<>();
        char[] strArray = inputString.toCharArray();
        for (char c : strArray) {
            if (charCountMap.containsKey(c)) {
                charCountMap.put(c, charCountMap.get(c) + 1);
            } else {
                charCountMap.put(c, 1);
            }
        }

        Set<Character> charsInString = charCountMap.keySet();
        System.out.println("Duplicate Characters in : " + inputString);

        for (Character ch : charsInString) {
            if (charCountMap.get(ch) > 1) {
                System.out.println(ch + " : " + charCountMap.get(ch));
            }
        }
    }
}
```

```
Duplicate Characters in : Learn Java Programming

a : 4

g : 2

m : 2

n : 2

r : 3
```

4.) Java program to count Occurrences of Each Character in String

```java
import java.util.HashMap;

public class Main {

    public static void main(String[] args) {
        CharacterCount("Test Automation Java Automation");
    }

    static void CharacterCount(String inputString) {
        HashMap<String,Integer> charCountMap = new HashMap<>();
        for(String s : inputString.split(" "))
        {
            if(charCountMap.containsKey(s))
            {
                charCountMap.put(s,charCountMap.get(s)+1);
            }
            else
            {
                charCountMap.put(s,1);
            }
        }
        System.out.println("Count of Characters in a given string : " +
charCountMap);
    }
}
```
Count of Characters in a given string : **{Java=1, Automation=2, Test=1}**

5.) Java program to count the number of words in a string

```java
public class Main {
    public static void main(String[] args) {
        System.out.println("Enter the String");
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        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);    }
}
```
Enter the String: Welcome to Java World
Number of words in a string: **4**

6.) Java program to find all permutations of a given string

```java
import java.util.Scanner;

public class Main {
    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

7.) Java program to find if a string is Palindrome

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        String str = "madam";
        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;
    }
}
```

8.) Java program to determine if Two Strings are Anagrams

```java
public class Main {

    public static void main(String[] args) {
        String str1 = "listen";
        String str2 = "silent";
        System.out.println(areAnagrams(str1,str2));
    }

    static boolean areAnagrams(String str1, String str2) {
        if(str1.length() != str2.length())
        {
            return false;
        }

        int[] charCount = new int[256];
        for( int i = 0; i < str1.length(); i++)
        {
            charCount[str1.charAt(i)]++;
            charCount[str2.charAt(i)]--;
        }

        for ( int count : charCount)
        {
            if ( count !=0 )
            {
                return false;
            }
        }
        return true;
    }
}
```

9.) Java program to Count Vowels and Consonants in a given string

```java
public class Main {
    public static void main(String[] args) {
        String str = "Hello World";
        VowelConsonantCount(str);
    }

    static void VowelConsonantCount(String str) {
        int vowels = 0, consonants = 0;
        str = str.toLowerCase();
        for (char c : str.toCharArray()) {
            if (c >= 'a' && c <= 'z') {
                if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')
{

                    vowels++;
                } else {
                    consonants++;
                }
            }
        }
        System.out.println("Vowels : " + vowels);
        System.out.println("Consonants : " + consonants);
    }
}
```

Vowels : 3

Consonants : 7

10.) Java program to print unqiue characters

```java
import java.util.Scanner;

public class Main {
    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("Unique characters in \"" + input + "\":");
        printUniqueCharacters(input);

    }



    public static void printUniqueCharacters(String str) {
        // Assume ASCII characters (0-127), use boolean array to track
character occurrences
        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 + " ");
            }
        }

    }
}
```

Enter a string: **Java Automation**

Unique characters in "Java Automation":

J a v   A u t o m i n

11.) Java program to print even indexed characters

```java
import java.util.Scanner;

public class Main {
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("Even indexed characters in \"" + input + "\":");
    printEvenIndexedCharacters(input);

}


public static void printEvenIndexedCharacters(String str) {
    for (int i = 0; i < str.length(); i++) {
        if (i % 2 == 0) {
            System.out.print(str.charAt(i));
        }
    }

}
}
```

Enter a string: Automation

Even indexed characters in "Automation":

Atmto

12.) Java program to remove space from a given string

```java
import java.util.Scanner;

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

        String stringWithoutSpaces = removeSpaces(input);
        System.out.println("String without spaces: " +
stringWithoutSpaces);
    }

    public static String removeSpaces(String str) {
        StringBuilder result = new StringBuilder();
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) != ' ') {
                result.append(str.charAt(i));
            }
        }
        return result.toString();
    }
}
```

```
Enter a string with spaces: Welcome to Java World
String without spaces: WelcometoJavaWorld
```

13.) Java program to print each letter twice from a given string

```java
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

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

        String doubledString = doubleCharacters(input);
        System.out.println("Doubled characters: " + doubledString);
    }

    public static String doubleCharacters(String str) {

        StringBuilder doubled = new StringBuilder();
        for (int i = 0; i < str.length(); i++) {
            char ch = str.charAt(i);
            doubled.append(ch).append(ch); // Append each character twice
        }
        return doubled.toString();
    }
}
```

```
Enter a string: hello
Doubled characters: hheelllloo
```

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

```java
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

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

15.)   Java program to gives Output: a2b2c3d2 for the Input String Str = "aabbcccdd"

```java
import java.util.Scanner;

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

        String output = getCharacterCount(input);
        System.out.println("Output: " + output);
    }


    public static String getCharacterCount(String str) {
        StringBuilder result = new StringBuilder();
        int count = 1;

        for (int i = 0; i < str.length(); i++) {
            // If the next character is the same, increase the count
            if (i + 1 < str.length() && str.charAt(i) == str.charAt(i
+ 1)) {
                count++;
            } else {
                // Append the character and its count to the result
                result.append(str.charAt(i)).append(count);
                count = 1; // Reset the count
            }
        }

        return result.toString();
    }
}
```

```
Enter a string: aabbcccdd

Output: a2b2c3d2
```

16.)   Java program to gives two Output: "abcde", "ABCDE" for the Input String Str = "aBACbcEDed"

```java
import java.util.Scanner;

public class Main {
    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("Original String is: "+ input);
                    separateCharacters(input);
        }

    public static void separateCharacters(String input)
        {
        StringBuilder lowerCase = new StringBuilder();
        StringBuilder upperCase = new StringBuilder();

        for(char ch : input.toCharArray())
        {
            if(Character.isLowerCase(ch))
            {
                lowerCase.append(ch);
            }
            else
            {
                upperCase.append(ch);
            }
        }
        System.out.println("Output in lowercase: "+lowerCase);
        System.out.println("Output in uppercase "+upperCase);
    }
```

Enter a string: **aBACbcEDed**

Output in lowercase: abced

Output in uppercase: ABCED

17.) Java program to gives two Output: "Subburaj", "123" for the Input String Str = "Subbu123raj"

```java
import java.util.Scanner;

public class Main {
    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("Original String is: "+ input);
                    separateAplhaAndNumeric(input);
        }

    public static void separateAlphaAndNumeric(String input)
        {
         StringBuilder alphaPart = new StringBuilder();
         StringBuilder numericPart = new StringBuilder();

         for(char ch : input.toCharArray())
         {
             if(Character.isLetter(ch))
             {
                 alphaPart.append(ch);
             }
             else if (Character.isDigit(ch))
             {
                 numericPart.append(ch);
             }
         }

         System.out.println("Output in Alpha: "+alphaPart.toString());
         System.out.println("Output in Numeric:
        "+numericPart.toString());
         }
```

Enter a string: **Subbu123raj**

Output in lowercase: Subburaj

Output in uppercase: 123

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

```java
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"

```java
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

20.) Java program to find the longest without repeating characters

```java
import java.util.HashSet;

public class Main {
    public static void main(String[] args) {
        String s1 = "abcabcbb"; // Expected: "abc", length 3
        String s2 = "bbbbb";    // Expected: "b", length 1
        String s3 = "pwwkew";   // Expected: "wke", length 3
        String s4 = "";         // Expected: "", length 0

        System.out.println("Longest substring without repeating
characters in s1: " + lengthOfLongestSubstring(s1)); // Output: 3
        System.out.println("Longest substring without repeating
characters in s2: " + lengthOfLongestSubstring(s2)); // Output: 1
        System.out.println("Longest substring without repeating
characters in s3: " + lengthOfLongestSubstring(s3)); // Output: 3
        System.out.println("Longest substring without repeating
characters in s4: " + lengthOfLongestSubstring(s4)); // Output: 0
    }

    public static int lengthOfLongestSubstring(String s) {
        HashSet<Character> set = new HashSet<>();
        int maxLength = 0;
        int start = 0;
        int end = 0;

        while (end < s.length()) {
            char currentChar = s.charAt(end);
            if (!set.contains(currentChar)) {
                set.add(currentChar);
                maxLength = Math.max(maxLength, end - start + 1);
                end++;
            } else {
                set.remove(s.charAt(start));
                start++;
            }
        }

        return maxLength;
    }
}
```

Arrays

1.) Find common elements between two arrays

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

public class CommonElements {
    public static void main(String[] args) {
        int[] array1 = {1, 2, 3, 4, 5};
        int[] array2 = {4, 5, 6, 7, 8};

        Set<Integer> commonElements = findCommonElements(array1,
array2);

        System.out.println("Common elements: " + commonElements);
    }

    public static Set<Integer> findCommonElements(int[] array1,
int[] array2) {
        Set<Integer> set1 = new HashSet<>();
        Set<Integer> commonSet = new HashSet<>();

        // Add elements of the first array to the set
        for (int num : array1) {
            set1.add(num);
        }

        // Check for common elements in the second array
        for (int num : array2) {
            if (set1.contains(num)) {
                commonSet.add(num);
            }
        }

        return commonSet;
    }
}
```

Input: array1 = {1,2,3,4,5} and

array2 = {4,5,6,7,8}

Output: Common elements: [4, 5]

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.");
        }
    }
}
```

Output:
First element: Apple
Last element: Elderberry

3.) Sort an array without using in-built method

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

        selectionSort(array);

      System.out.println("Sorted array:");
        for (int num : array) {
           System.out.print(num + " ");
         }
  }

public static void selectionSort(int[] array) {
        int n = array.length;
        for (int i = 0; i < n - 1; i++) {
        int minIndex = i;
           for (int j = i + 1; j < n; j++) {
             if (array[j] < array[minIndex]) {
             minIndex = j;
           }
         }

        // Swap array[i] and array[minIndex]
        int temp = array[i];
        array[i] = array[minIndex];
        array[minIndex] = temp;
      }
   }
}
```

Output:
Sorted array:
1 2 5 6 9

4.) **Remove duplicates** from an Array

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

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

        int[] uniqueArray = removeDuplicates(array);

        System.out.println("Array with duplicates removed:");
        for (int num : uniqueArray) {
            System.out.print(num + " ");
        }
    }

    public static int[] removeDuplicates(int[] array) {
        Set<Integer> set = new HashSet<>();
        for (int num : array) {
            set.add(num);
        }

        int[] result = new int[set.size()];
        int i = 0;
        for (int num : set) {
            result[i++] = num;
        }

        return result;
    }
}
```

Output:
Array with duplicates removed:

1 2 5 6 9

5.) Remove duplicates from an ArrayList

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

public class Main {
    public static void main(String[] args) {
        ArrayList<Integer> arrayList = new ArrayList<>();
        arrayList.add(5);
        arrayList.add(2);
        arrayList.add(9);
        arrayList.add(1);
        arrayList.add(6);
        arrayList.add(2);
        arrayList.add(5);

        ArrayList<Integer> uniqueList =
removeDuplicates(arrayList);

        System.out.println("ArrayList with duplicates
removed:");
        for (int num : uniqueList) {
            System.out.print(num + " ");
        }
    }

    public static ArrayList<Integer>
removeDuplicates(ArrayList<Integer> list) {
        Set<Integer> set = new HashSet<>(list);
        return new ArrayList<>(set);
    }
}
```

Output:
ArrayList with duplicates removed:

1 2 5 6 9

6.) Find the missing number in an Array

```java
public class Main {
    public static void main(String[] args) {
        int[] array = {1, 2, 4, 5, 6}; // Missing number is 3
        int missingNumber = findMissingNumber(array);
        System.out.println("The missing number is: " + missingNumber);
    }

    public static int findMissingNumber(int[] array) {
        int n = array.length + 1; // Since one number is missing, the length
should be n+1
        int totalSum = n * (n + 1) / 2; // Sum of first n natural numbers

        int arraySum = 0;
        for (int num : array) {
            arraySum += num;
        }
        return totalSum - arraySum;
    }
}
```

Output:
The missing number is: 3

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

```java
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

8.) Search element in an Array

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

        int index = linearSearch(array, target);

        if (index != -1) {
            System.out.println("Element " + target + " found at index: " +
index);
        } else {
            System.out.println("Element " + target + " not found in the
array.");
        }
    }

    public static int linearSearch(int[] array, int target) {
        for (int i = 0; i < array.length; i++) {
            if (array[i] == target) {
                return i; // Element found, return index
            }
        }
        return -1; // Element not found
    }
}
```

Output:

Element 6 found at index: 4

Element 10 not found in the array

9.) Array consists of integers and special characters,sum only integers

```java
public class Main {
    public static void main(String[] args) {
        String[] array = {"5", "2", "9", "a", "1", "6", "#", "3"};

        int sum = sumIntegers(array);

        System.out.println("Sum of integers in the array: " + sum);
    }


    public static int sumIntegers(String[] array) {
        int sum = 0;
        for (String element : array) {
        try {
            int num = Integer.parseInt(element);
            sum += num;
        } catch (NumberFormatException e) {
            // Ignore non-integer elements
        }
      }
      return sum;
    }
}
```

Output:

Sum of integers in the array: 26

10.) Find Minimum and Maximum from an Array

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

        // Find maximum and minimum
        int max = findMaximum(array);
        int min = findMinimum(array);

        // Print the results
        System.out.println("Minimum value in the array: " + min);
        System.out.println("Maximum value in the array: " + max);
    }

    public static int findMaximum(int[] array) {
        if (array.length == 0) {
            throw new IllegalArgumentException("Array must not be empty");
        }
        int max = array[0]; // Initialize max to the first element
        for (int i = 1; i < array.length; i++) {
            if (array[i] > max) {
                max = array[i]; // Update max if current element is larger
            }
        }
        return max;
    }

    public static int findMinimum(int[] array) {
        if (array.length == 0) {
            throw new IllegalArgumentException("Array must not be empty");
        }
        int min = array[0]; // Initialize min to the first element
        for (int i = 1; i < array.length; i++) {
            if (array[i] < min) {
                min = array[i]; // Update min if current element is smaller
            }
        }
        return min;     }
}
```

Output:

Minimum value in the array: 1

Maximum value in the array: 9

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

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

```java
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
```

12.) Java program – input array was given [1,1,2,2,3,4,5,5,6,6], Output – [3,4]

```java
import java.util.HashMap;
import java.util.Map;
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        int[] array = {1, 1, 2, 2, 3, 4, 5, 5, 6, 6};
        List<Integer> result = findNonRepeatedElements(array);
        System.out.println("Non-repeated elements: " + result);
    }

    public static List<Integer> findNonRepeatedElements(int[] array) {

        // Step 1: Count occurrences of each element using a HashMap
        Map<Integer, Integer> countMap = new HashMap<>();
        for (int num : array) {
            countMap.put(num, countMap.getOrDefault(num, 0) + 1);
        }

        // Step 2: Identify elements with count equal to 1 (non-repeated)
        List<Integer> nonRepeatedElements = new ArrayList<>();
        for (Map.Entry<Integer, Integer> entry : countMap.entrySet()) {
            if (entry.getValue() == 1) {
                nonRepeatedElements.add(entry.getKey());
            }
        }
        return nonRepeatedElements;
    }
}
```

Output :

Non-repeated elements: [3, 4]

Java program to implement hashcode and equals

```java
import java.util.Objects;

public class Student {
private int id;
private String name;

// Constructor
public Student(int id, String name) {
    this.id = id;
    this.name = name;
}

// Getters and setters (omitted for brevity)

// hashCode method
@Override
public int hashCode() {
    return Objects.hash(id, name);
}

// equals method
@Override
public boolean equals(Object obj) {
    if (this == obj)
        return true;
    if (obj == null || getClass() != obj.getClass())
        return false;
    Student student = (Student) obj;
    return id == student.id && Objects.equals(name, student.name);
}

public static void main(String[] args) {
    // Creating objects of Student class
    Student student1 = new Student(1, "Alice");
    Student student2 = new Student(2, "Bob");
    Student student3 = new Student(1, "Alice");

    // Testing equals method
    System.out.println("student1.equals(student2): " +
student1.equals(student2)); // Output: false
    System.out.println("student1.equals(student3): " +
student1.equals(student3)); // Output: true

    // Testing hashCode method
    System.out.println("Hashcode of student1: " + student1.hashCode());
    System.out.println("Hashcode of student2: " + student2.hashCode());
    System.out.println("Hashcode of student3: " + student3.hashCode());
}
}
```

IMPORTANT JAVA PROGRAMS BASED ON STRING

Check if two strings are anagrams

```java
import java.util.Arrays;

public class AnagramCheck {
    public static void main(String[] args) {
        String str1 = "listen";
        String str2 = "silent";

        if (isAnagram(str1, str2)) {
            System.out.println("Strings are anagrams.");
        } else {
            System.out.println("Strings are not anagrams.");
        }
    }

    public static boolean isAnagram(String str1, String str2) {
        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);
    }
}
```

Explanation:

- **if (str1.length() != str2.length())**: Checks if both strings have the same length.
- **Arrays.sort(arr1)**: Sorts the character array of the first string.
- **Arrays.equals(arr1, arr2)**: Compares the sorted character arrays of both strings.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Check if a string is a palindrome

```java
public class PalindromeCheck {
    public static void main(String[] args) {
        String str = "madam";

        if (isPalindrome(str)) {
            System.out.println("String is a palindrome.");
        } else {
            System.out.println("String is not a palindrome.");
        }
    }

    public static boolean isPalindrome(String str) {
        int left = 0;
        int right = str.length() - 1;

        while (left < right) {
            if (str.charAt(left) != str.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
}
```

Explanation:

- **while (left < right)**: Loops through the string comparing characters from the start and end.
- **str.charAt(left) != str.charAt(right)**: If characters don't match, it's not a palindrome.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Count the number of vowels and consonants in a string

```java
public class CountVowelsConsonants {
    public static void main(String[] args) {
        String str = "automation";
        int[] count = countVowelsAndConsonants(str);

        System.out.println("Vowels: " + count[0]);
        System.out.println("Consonants: " + count[1]);
    }

    public static int[] countVowelsAndConsonants(String str) {
        int vowelCount = 0;
        int consonantCount = 0;
        String vowels = "aeiouAEIOU";

        for (char ch : str.toCharArray()) {
            if (vowels.indexOf(ch) != -1) {
                vowelCount++;
            } else if (Character.isLetter(ch)) {
                consonantCount++;
            }
        }

        return new int[]{vowelCount, consonantCount};
    }
}
```

Explanation:

- **vowels.indexOf(ch) != -1**: Checks if the character is a vowel.
- **Character.isLetter(ch)**: Ensures that only letters are counted as consonants.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Find the first non-repeating character in a string

```java
import java.util.LinkedHashMap;
import java.util.Map;

public class FirstNonRepeatingChar {
    public static void main(String[] args) {
        String str = "automation";
        char result = findFirstNonRepeating(str);
        System.out.println("First non-repeating character: " + result);
    }

    public static char findFirstNonRepeating(String str) {
        Map<Character, Integer> charCountMap = new LinkedHashMap<>();

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

        for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
            if (entry.getValue() == 1) {
                return entry.getKey();
            }
        }
        return '\0';
    }
}
```

Explanation:

- **charCountMap.getOrDefault(ch, 0) + 1**: Increments the count of each character.
- **if (entry.getValue() == 1)**: Finds the first character that appears only once.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Reverse a string

```java
public class ReverseString {
    public static void main(String[] args) {
        String str = "Selenium";
        String reversed = reverse(str);
        System.out.println("Reversed string: " + reversed);
    }

    public static String reverse(String str) {
        StringBuilder reversedStr = new StringBuilder();

        for (int i = str.length() - 1; i >= 0; i--) {
            reversedStr.append(str.charAt(i));
        }

        return reversedStr.toString();
    }
}
```

Explanation:

- **for (int i = str.length() - 1; i >= 0; i--)**: Loops through the string from the end to the beginning.
- **reversedStr.append(str.charAt(i))**: Appends each character to the reversed string.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Check if a string contains only digits

```java
public class CheckDigits {
    public static void main(String[] args) {
        String str = "12345";

        if (containsOnlyDigits(str)) {
            System.out.println("String contains only digits.");
        } else {
            System.out.println("String contains non-digit characters.");
        }
    }

    public static boolean containsOnlyDigits(String str) {
        for (char ch : str.toCharArray()) {
            if (!Character.isDigit(ch)) {
                return false;
            }
        }
        return true;
    }
}
```

Explanation:

- **Character.isDigit(ch)**: Checks if each character is a digit.
- If any character is not a digit, it returns false.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Count the occurrence of each character in a string

```java
import java.util.HashMap;
import java.util.Map;

public class CharOccurrence {
    public static void main(String[] args) {
        String str = "testing";
        countCharOccurrence(str);
    }

    public static void countCharOccurrence(String str) {
        Map<Character, Integer> charCountMap = new HashMap<>();

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

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

Explanation:

- **charCountMap.put(ch, charCountMap.getOrDefault(ch, 0) + 1)**: Increments the count of each character.
- **for (Map.Entry<Character, Integer> entry)**: Iterates through the map to print the count of each character.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Remove duplicate characters from a string

```java
public class RemoveDuplicates {
    public static void main(String[] args) {
        String str = "automation";
        String result = removeDuplicates(str);
        System.out.println("String after removing duplicates: " + result);
    }

    public static String removeDuplicates(String str) {
        StringBuilder result = new StringBuilder();

        for (char ch : str.toCharArray()) {
            if (result.indexOf(String.valueOf(ch)) == -1) {
                result.append(ch);
            }
        }

        return result.toString();
    }
}
```

Explanation:

- **result.indexOf(String.valueOf(ch)) == -1**: Checks if the character is already present in the result.
- If not present, the character is appended to the result.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Find all substrings of a string

```java
public class Substrings {
    public static void main(String[] args) {
        String str = "abc";
        findAllSubstrings(str);
    }

    public static void findAllSubstrings(String str) {
        for (int i = 0; i < str.length(); i++) {
            for (int j = i + 1; j <= str.length(); j++) {
                System.out.println(str.substring(i, j));
            }
        }
    }
}
```

Explanation:

- **str.substring(i, j)**: Extracts all substrings starting from index i to j.
- Nested loops ensure that all possible substrings are printed.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Find the most frequent character in a string

```java
import java.util.HashMap;
import java.util.Map;

public class MostFrequentChar {
    public static void main(String[] args) {
        String str = "success";
        char mostFrequent = findMostFrequentChar(str);
        System.out.println("Most frequent character: " + mostFrequent);
    }

    public static char findMostFrequentChar(String str) {
        Map<Character, Integer> charCountMap = new HashMap<>();
        int maxCount = 0;
        char mostFrequent = '\0';

        for (char ch : str.toCharArray()) {
            int count = charCountMap.getOrDefault(ch, 0) + 1;
            charCountMap.put(ch, count);

            if (count > maxCount) {
                maxCount = count;
                mostFrequent = ch;
            }
        }
        return mostFrequent;
    }
}
```

Explanation:

- **if (count > maxCount)**: Tracks the character with the highest frequency.
- Updates the most frequent character during iteration.

IMPORTANT JAVA PROGRAMS BASED ON STRING

Convert the first letter of each word in a string to uppercase

```java
public class FirstLetterUppercase {
    public static void main(String[] args) {
        String sentence = "quality assurance automation testing";
        String result = convertToUpperCase(sentence);
        System.out.println("Converted sentence: " + result);
    }

    public static String convertToUpperCase(String sentence) {
        StringBuilder result = new StringBuilder();
        boolean capitalize = true;

        for (char ch : sentence.toCharArray()) {
            if (capitalize && Character.isLetter(ch)) {
                result.append(Character.toUpperCase(ch));
                capitalize = false;
            } else {
                result.append(ch);
            }

            if (ch == ' ') {
                capitalize = true;
            }
        }

        return result.toString();
    }
}
```

Explanation:

- **boolean capitalize = true**: A flag to indicate when to capitalize a letter.
- **Character.toUpperCase(ch)**: Converts the character to uppercase if it is the first letter of a word.
- The flag is reset after every space character, allowing the first letter of the next word to be capitalized.