2) Addition of Two Numbers:

public class AddNumbers {

 public static void main(String[] args) {

 int num1 = 5, num2 = 10, sum;

 sum = num1 + num2;

 System.out.println(“Sum of ” + num1 + ” and ” + num2 + ” is: ” + sum);

 }

}

Output:

Sum of 5 and 10 is: 15

3) Find Maximum of Three Numbers:

public class MaxOfThreeNumbers {

 public static void main(String[] args) {

 int num1 = 10, num2 = 20, num3 = 15, max;

 max = (num1 > num2) ? (num1 > num3 ? num1 : num3) : (num2 > num3 ? num2 : num3);

 System.out.println(“Maximum of ” + num1 + “, ” + num2 + “, and ” + num3 + ” is: ” + max);

 }

}

Output:

Maximum of 10, 20, and 15 is: 20

4) Check Even or Odd Number:

public class EvenOdd {

 public static void main(String[] args) {

 int num = 5;

 if(num % 2 == 0)

 System.out.println(num + ” is even.”);

 else

 System.out.println(num + ” is odd.”);

 }

}

Output:

5 is odd.

5) Factorial of a Number:

public class Factorial {

 public static void main(String[] args) {

 int num = 5, factorial = 1;

 for(int i = 1; i <= num; ++i) {

 factorial \*= i;

 }

 System.out.println(“Factorial of ” + num + ” is: ” + factorial);

 }

}

Output:

Factorial of 5 is: 120

6) Print Pattern in Java:

public class PrintPattern {

 public static void main(String[] args) {

 int rows = 5;

 for (int i = 1; i <= rows; ++i) {

 for (int j = 1; j <= i; ++j) {

 System.out.print(“\* “);

 }

 System.out.println();

 }

 }

}

Output:

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

7) Add Two Binary Numbers in Java:

public class AddBinaryNumbers {

 public static void main(String[] args) {

 long binary1 = 1010, binary2 = 1101;

 int i = 0, remainder = 0;

 long[] sum = new long[20];

 while (binary1 != 0 || binary2 != 0) {

 sum[i++] = (binary1 % 10 + binary2 % 10 + remainder) % 2;

 remainder = (int) (binary1 % 10 + binary2 % 10 + remainder) / 2;

 binary1 = binary1 / 10;

 binary2 = binary2 / 10;

 }

 if (remainder != 0) {

 sum[i++] = remainder;

 }

 –i;

 System.out.print(“Sum of two binary numbers: “);

 while (i >= 0) {

 System.out.print(sum[i–]);

 }

 }

}

Output:

Sum of two binary numbers: 11011

8) Add Two Complex Numbers in Java:

class Complex {

 double real, imaginary;

 Complex(double r, double i) {

 this.real = r;

 this.imaginary = i;

 }

 public static Complex add(Complex c1, Complex c2) {

 Complex temp = new Complex(0, 0);

 temp.real = c1.real + c2.real;

 temp.imaginary = c1.imaginary + c2.imaginary;

 return temp;

 }

}

public class AddComplexNumbers {

 public static void main(String[] args) {

 Complex c1 = new Complex(4.5, 5);

 Complex c2 = new Complex(2.5, 3.5);

 Complex temp = Complex.add(c1, c2);

 System.out.println(“Sum = ” + temp.real + ” + ” + temp.imaginary + “i”);

 }

}

Output:

Sum = 7.0 + 8.5i

9) Multiply Two Numbers in Java:

public class MultiplyTwoNumbers {

 public static void main(String[] args) {

 double first = 2.5, second = 4.5;

 double product = first \* second;

 System.out.println(“The product is: ” + product);

 }

}

Output:

The product is: 11.25

10) Check Leap Year in Java:

public class LeapYear {

 public static void main(String[] args) {

 int year = 2024;

 if (((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0)) {

 System.out.println(year + ” is a leap year.”);

 } else {

 System.out.println(year + ” is not a leap year.”);

 }

 }

}

Output:

2024 is a leap year.

11) Check Vowel or Consonant in Java:

public class VowelConsonant {

 public static void main(String[] args) {

 char ch = ‘A’;

 if (ch == ‘a’ || ch == ‘e’ || ch == ‘i’ || ch == ‘o’ || ch == ‘u’

 || ch == ‘A’ || ch == ‘E’ || ch == ‘I’ || ch == ‘O’ || ch == ‘U’) {

 System.out.println(ch + ” is a vowel.”);

 } else {

 System.out.println(ch + ” is a consonant.”);

 }

 }

}

Output:

A is a vowel.

12) Calculate Compound Interest in Java:

public class CompoundInterest {

 public static void main(String[] args) {

 double principal = 15000, rate = 5.5, time = 3;

 double compoundInterest = principal \* (Math.pow((1 + rate / 100), time)) – principal;

 System.out.println(“Compound Interest: ” + compoundInterest);

 }

}

Output:

Compound Interest: 2653.4375

13) Java Program to Calculate Simple Interest:

import java.util.Scanner;

public class SimpleInterest {

 public static void main(String[] args) {

 Scanner input = new Scanner(System.in);

 System.out.print(“Enter principal amount: “);

 double principal = input.nextDouble();

 System.out.print(“Enter rate of interest: “);

 double rate = input.nextDouble();

 System.out.print(“Enter time period in years: “);

 double time = input.nextDouble();

 double simpleInterest = (principal \* rate \* time) / 100;

 System.out.println(“Simple Interest: ” + simpleInterest);

 input.close();

 }

}

Output:

Enter principal amount: 5000

Enter rate of interest: 2.5

Enter time period in years: 3

Simple Interest: 375.0

14) Java Program to Find Quotient and Remainder:

import java.util.Scanner;

public class QuotientRemainder {

 public static void main(String[] args) {

 Scanner input = new Scanner(System.in);

 System.out.print(“Enter dividend: “);

 int dividend = input.nextInt();

 System.out.print(“Enter divisor: “);

 int divisor = input.nextInt();

 int quotient = dividend / divisor;

 int remainder = dividend % divisor;

 System.out.println(“Quotient: ” + quotient);

 System.out.println(“Remainder: ” + remainder);

 input.close();

 }

}

Output:

Enter dividend: 17

Enter divisor: 5

Quotient: 3

Remainder: 2

15) Java Program to Calculate Power of a Number:

import java.util.Scanner;

public class PowerOfNumber {

 public static void main(String[] args) {

 Scanner input = new Scanner(System.in);

 System.out.print(“Enter base: “);

 int base = input.nextInt();

 System.out.print(“Enter exponent: “);

 int exponent = input.nextInt();

 long result = 1;

 while (exponent != 0) {

 result \*= base;

 –exponent;

 }

 System.out.println(“Result: ” + result);

 input.close();

 }

}

Output:

Enter base: 3

Enter exponent: 4

Result: 81

16) Java Program to Convert char to String and String to Char:

public class CharStringConversion {

 public static void main(String[] args) {

 // Convert char to String

 char ch = ‘A’;

 String str = Character.toString(ch);

 System.out.println(“Char to String: ” + str);

 // Convert String to char

 String s = “Hello”;

 char c = s.charAt(0);

 System.out.println(“String to Char: ” + c);

 }

}

17) Java Program to Find Duplicate Characters in a String:

import java.util.HashMap;

import java.util.Map;

public class DuplicateCharacters {

 public static void main(String[] args) {

 String str = “programming”;

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

 for (char ch : str.toCharArray()) {

 if (charCountMap.containsKey(ch)) {

 charCountMap.put(ch, charCountMap.get(ch) + 1);

 } else {

 charCountMap.put(ch, 1);

 }

 }

 System.out.println(“Duplicate Characters:”);

 for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {

 if (entry.getValue() > 1) {

 System.out.println(entry.getKey() + ” – ” + entry.getValue() + ” times”);

 }

 }

 }

}

18) Java Program to Check Palindrome String using Stack, Queue, For, and While loop:

import java.util.LinkedList;

import java.util.Queue;

import java.util.Stack;

public class PalindromeCheck {

 public static void main(String[] args) {

 String str = “racecar”;

 Stack<Character> stack = new Stack<>();

 Queue<Character> queue = new LinkedList<>();

 for (char ch : str.toCharArray()) {

 stack.push(ch);

 queue.add(ch);

 }

 boolean isPalindrome = true;

 while (!stack.isEmpty() && !queue.isEmpty()) {

 if (!stack.pop().equals(queue.remove())) {

 isPalindrome = false;

 break;

 }

 }

 if (isPalindrome) {

 System.out.println(str + ” is a palindrome.”);

 } else {

 System.out.println(str + ” is not a palindrome.”);

 }

 }

}

19) Java Program to Sort Strings in Alphabetical Order:

import java.util.Arrays;

public class SortStrings {

 public static void main(String[] args) {

 String[] strings = {“orange”, “apple”, “banana”, “grape”};

 Arrays.sort(strings);

 System.out.println(“Sorted Strings:”);

 for (String s : strings) {

 System.out.println(s);

 }

 }

}

20) Java Program to Reverse Words in a String:

public class ReverseWords {

 public static void main(String[] args) {

 String str = “Hello World”;

 String[] words = str.split(” “);

 StringBuilder reversed = new StringBuilder();

 for (int i = words.length – 1; i >= 0; i–) {

 reversed.append(words[i]).append(” “);

 }

 System.out.println(“Reversed Words: ” + reversed.toString().trim());

 }

}

21) Java Program to perform bubble sort on Strings:

import java.util.Arrays;

public class BubbleSortStrings {

 public static void main(String[] args) {

 String[] arr = {“banana”, “apple”, “orange”, “grapes”, “pineapple”};

 bubbleSort(arr);

 System.out.println(“Sorted Array: ” + Arrays.toString(arr));

 }

 public static void bubbleSort(String[] arr) {

 int n = arr.length;

 for (int i = 0; i < n – 1; i++) {

 for (int j = 0; j < n – i – 1; j++) {

 if (arr[j].compareTo(arr[j + 1]) > 0) {

 // swap arr[j] and arr[j+1]

 String temp = arr[j];

 arr[j] = arr[j + 1];

 arr[j + 1] = temp;

 }

 }

 }

 }

}

Output:

Sorted Array: [apple, banana, grapes, orange, pineapple]

22) Java program to find occurrence of a character in a String:

public class CharacterOccurrences {

 public static void main(String[] args) {

 String str = “hello world”;

 char ch = ‘o’;

 int count = 0;

 for (int i = 0; i < str.length(); i++) {

 if (str.charAt(i) == ch) {

 count++;

 }

 }

 System.out.println(“Occurrences of ‘” + ch + “‘ in the string: ” + count);

 }

}

Output:

Occurrences of ‘o’ in the string: 2

23) Java program to count vowels and consonants in a String:

public class VowelsConsonantsCount {

 public static void main(String[] args) {

 String str = “hello world”;

 int vowels = 0, consonants = 0;

 str = str.toLowerCase();

 for (int i = 0; i < str.length(); i++) {

 char ch = str.charAt(i);

 if (ch == ‘a’ || ch == ‘e’ || ch == ‘i’ || ch == ‘o’ || ch == ‘u’) {

 vowels++;

 } else if (ch >= ‘a’ && ch <= ‘z’) {

 consonants++;

 }

 }

 System.out.println(“Vowels: ” + vowels);

 System.out.println(“Consonants: ” + consonants);

 }

}

Output:

Vowels: 3

Consonants: 7

24) Java Program to check two strings are anagram or not:

import java.util.Arrays;

public class AnagramCheck {

 public static void main(String[] args) {

 String str1 = “listen”;

 String str2 = “silent”;

 boolean isAnagram = checkAnagram(str1, str2);

 if (isAnagram) {

 System.out.println(str1 + ” and ” + str2 + ” are anagrams.”);

 } else {

 System.out.println(str1 + ” and ” + str2 + ” are not anagrams.”);

 }

 }

 public static boolean checkAnagram(String str1, String str2) {

 if (str1.length() != str2.length()) {

 return false;

 }

 char[] chars1 = str1.toCharArray();

 char[] chars2 = str2.toCharArray();

 Arrays.sort(chars1);

 Arrays.sort(chars2);

 return Arrays.equals(chars1, chars2);

 }

}

Output:

listen and silent are anagrams.

25) Java Program to divide a string in ‘n’ equal parts:

public class DivideString {

 public static void main(String[] args) {

 String str = “abcdefghi”;

 int n = 3;

 int len = str.length();

 int partLength = len / n;

 int extraChars = len % n;

 int start = 0;

 for (int i = 0; i < n; i++) {

 int end = start + partLength + (i < extraChars ? 1 : 0);

 String part = str.substring(start, end);

 System.out.println(“Part ” + (i + 1) + “: ” + part);

 start = end;

 }

 }

}

Output:

Part 1: abc

Part 2: def

Part 3: ghi

26) Java Program to find all subsets of a string:

import java.util.ArrayList;

import java.util.List;

public class SubsetsOfString {

 public static void main(String[] args) {

 String str = “abc”;

 List<String> subsets = new ArrayList<>();

 generateSubsets(str, 0, “”, subsets);

 System.out.println(“All subsets of \”” + str + “\”: ” + subsets);

 }

 private static void generateSubsets(String str, int index, String current, List<String> subsets) {

 if (index == str.length()) {

 subsets.add(current);

 return;

 }

 generateSubsets(str, index + 1, current + str.charAt(index), subsets);

 generateSubsets(str, index + 1, current, subsets);

 }

}

Output:

All subsets of “abc”: [, c, b, bc, a, ac, ab, abc]

27) Java Program to find longest substring without repeating characters:

import java.util.HashSet;

import java.util.Set;

public class LongestSubstringWithoutRepeating {

 public static void main(String[] args) {

 String str = “abcabcbb”;

 System.out.println(“Longest substring without repeating characters: ” + longestSubstring(str));

 }

 public static int longestSubstring(String s) {

 Set<Character> set = new HashSet<>();

 int left = 0, right = 0, maxLen = 0;

 while (right < s.length()) {

 if (!set.contains(s.charAt(right))) {

 set.add(s.charAt(right++));

 maxLen = Math.max(maxLen, set.size());

 } else {

 set.remove(s.charAt(left++));

 }

 }

 return maxLen;

 }

}

Output:

Longest substring without repeating characters: 3

28) Java Program to find longest repeating sequence in a string:

public class LongestRepeatingSequence {

 public static void main(String[] args) {

 String str = “aabcaabdaab”;

 System.out.println(“Longest repeating sequence: ” + longestRepeatingSequence(str));

 }

 public static String longestRepeatingSequence(String str) {

 int n = str.length();

 int[][] dp = new int[n + 1][n + 1];

 int longest = 0, endIndex = 0;

 for (int i = 1; i <= n; i++) {

 for (int j = i + 1; j <= n; j++) {

 if (str.charAt(i – 1) == str.charAt(j – 1) && dp[i – 1][j – 1] < (j – i)) {

 dp[i][j] = dp[i – 1][j – 1] + 1;

 if (dp[i][j] > longest) {

 longest = dp[i][j];

 endIndex = i;

 }

 } else {

 dp[i][j] = 0;

 }

 }

 }

 return str.substring(endIndex – longest, endIndex);

 }

}

Output:

Longest repeating sequence: aab

29) Java Program to remove all the white spaces from a string:

public class RemoveWhiteSpaces {

 public static void main(String[] args) {

 String str = “This is a test string”;

 String trimmedStr = removeWhiteSpaces(str);

 System.out.println(“String after removing white spaces: \”” + trimmedStr + “\””);

 }

 public static String removeWhiteSpaces(String str) {

 return str.replaceAll(“\\s”, “”);

 }

}

Output:

String after removing white spaces: “Thisisateststring”

30) Program to find number of elements in an array:

public class ArrayLength {

 public static void main(String[] args) {

 int[] array = {1, 2, 3, 4, 5};

 int length = array.length;

 System.out.println(“Number of elements in the array: ” + length);

 }

}

Output:

Number of elements in the array: 5

31) Java Program to Calculate average of numbers using Array:

public class AverageOfArray {

 public static void main(String[] args) {

 int[] array = {5, 10, 15, 20, 25};

 int sum = 0;

 for (int num : array) {

 sum += num;

 }

 double average = (double) sum / array.length;

 System.out.println(“Average of numbers in the array: ” + average);

 }

}

Output:

Average of numbers in the array: 15.0

32) Java Program to Add the elements of an Array:

public class ArraySum {

 public static void main(String[] args) {

 int[] array = {1, 2, 3, 4, 5};

 int sum = 0;

 for (int num : array) {

 sum += num;

 }

 System.out.println(“Sum of elements in the array: ” + sum);

 }

}

Output:

Sum of elements in the array: 15

33) Java Program to reverse an array:

import java.util.Arrays;

public class ReverseArray {

 public static void main(String[] args) {

 int[] array = {1, 2, 3, 4, 5};

 reverseArray(array);

 System.out.println(“Reversed array: ” + Arrays.toString(array));

 }

 public static void reverseArray(int[] array) {

 int start = 0;

 int end = array.length – 1;

 while (start < end) {

 int temp = array[start];

 array[start] = array[end];

 array[end] = temp;

 start++;

 end–;

 }

 }

}

Output:

Reversed array: [5, 4, 3, 2, 1]

34) Java Program to sort an array in ascending order:

import java.util.Arrays;

public class SortArray {

 public static void main(String[] args) {

 int[] array = {5, 3, 9, 1, 7};

 Arrays.sort(array);

 System.out.println(“Sorted array in ascending order: ” + Arrays.toString(array));

 }

}

Output:

Sorted array in ascending order: [1, 3, 5, 7, 9]

35) Java Program to convert char Array to String:

public class CharArrayToString {

 public static void main(String[] args) {

 char[] charArray = {‘h’, ‘e’, ‘l’, ‘l’, ‘o’};

 String str = new String(charArray);

 System.out.println(“Converted String: ” + str);

 }

}

Output:

Converted String: hello