1. Given two integer numbers return their product only if the product is equal to or lower than 1000, else return their sum.

SOLUTION:

import java.util.Scanner;

public class ProductOrSum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the first number: ");

int x = scanner.nextInt();

System.out.print("Enter the second number: ");

int y = scanner.nextInt();

int product = x \* y;

if (product <= 1000) {

System.out.println("The product is " + product);

} else {

System.out.println("The sum is " + (x + y));

       }

    }

}

2. Write a program to iterate the first 10 numbers and in each iteration, print the sum of the current and previous number.

SOLUTION:

public class SumOfNumbers

{

public static void main(String[] args)

{

int sum = 0;

int previousNumber = 0;

for (int i = 1; i <= 10; i++)

{

sum += i;

System.out.println("The sum of the first " + i + " numbers is " + sum);

previousNumber = i;

       }

    }

}

3. Write a program to accept a string from the user and display characters that are present at an even index number. For example, str = "pynative" so you should display ‘p’, ‘n’, ‘t’, ‘v’.

SOLUTION:

import java.util.Scanner;

public class EvenIndexedCharacters {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a string: ");

String str = scanner.nextLine();

for (int i = 0; i < str.length(); i += 2) {

System.out.print(str.charAt(i));

}

System.out.println();

    }

}

4. Write a program to remove characters from a string starting from zero up to n and return a new string.

Note: n must be less than the length of the string.

SOLUTION:

import java.util.Scanner;

public class RemoveCharacter

{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: ");

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

arr[i] = scanner.nextInt();

}

int smallest = arr[0];

int largest = arr[0];

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

{

if (arr[i] < smallest)

{

smallest = arr[i];

}

if (arr[i] > largest)

{

largest = arr[i];

}

}

System.out.println("The smallest element is: " + smallest);

System.out.println("The largest element is: " + largest);

    }

}

5. Write a function to return True if the first and last number of a given list is same. If numbers are different then return False.

SOLUTION:

import java.util.Scanner;

public class CheckFirstAndLast

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the size of the list: ");

int size = scanner.nextInt();

System.out.println("Enter the elements of the list: ");

int[] list = new int[size];

for (int i = 0; i < size; i++)

{

list[i] = scanner.nextInt();

}

boolean isEqual = list[0] == list[size - 1];

System.out.println("The first and last number of the list are the same: " + isEqual);

    }

}

6. Iterate the given list of numbers and print only those numbers which are divisible by 5.

SOLUTION:

import java.util.Scanner;

public class DivisibleByFive

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the size of the list: ");

int size = scanner.nextInt();

System.out.println("Enter the elements of the list: ");

int[] list = new int[size];

for (int i = 0; i < size; i++)

{

list[i] = scanner.nextInt();

}

for (int number : list)

{

if (number % 5 == 0)

{

System.out.println(number);

}

       }

    }

}

7. Print the following pattern, Print Pattern using for loop.

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

SOLUTION:

public class Pattern

{

public static void main(String[] args)

{

for (int i = 1; i <= 5; i++)

{

for (int j = 1; j <= i; j++)

{

System.out.print(i);

}

System.out.println();

       }

    }

}

8. Write a program to check if the given number is a palindrome number. A palindrome number is a number that is same after reverse. For example 545, is the palindrome numbers.

SOLUTION:

import java.util.Scanner;

public class Palindrome

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a number: ");

int number = scanner.nextInt();

int reversedNumber = 0;

int originalNumber = number;

while (number > 0)

{

reversedNumber = reversedNumber \* 10 + number % 10;

number /= 10;

}

if (originalNumber == reversedNumber)

{

System.out.println("The number is a palindrome.");

}

else

{

System.out.println("The number is not a palindrome.");

       }

    }

}

9. Given a two list of numbers, write a program to create a new list such that the new list should contain odd numbers from the first list and even numbers from the second list.

SOLUTION:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Read the input from the user

System.out.println("Enter a number: ");

int number = scanner.nextInt();

// Check if the number is prime

boolean isPrime = true;

for (int i = 2; i <= number / 2; i++) {

if (number % i == 0) {

isPrime = false;

break;

}

}

if (isPrime)

{

System.out.println("The number is prime.");

}

else

{

System.out.println("The number is not prime.");

       }

    }

}

10. Write a Program to extract each digit from an integer in the reverse order. For example, If the given int is 7536, the output shall be “6 3 5 7“, with a space separating the digits.

SOLUTION:

import java.util.Scanner;

public class ReverseDigits {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Read the input from the user

System.out.println("Enter an integer: ");

int number = scanner.nextInt();

// Reverse the digits of the number

int reversedNumber = 0;

while (number > 0) {

reversedNumber = reversedNumber \* 10 + number % 10;

number /= 10;

}

// Print the reversed number

System.out.println("The reversed number is: " + reversedNumber);

    }

}

11. Hello, World!: Write a program that prints "Hello, World!" to the console.

SOLUTION:

12. Even or Odd: Write a program that takes an integer as input and prints whether it is even or odd.

SOLUTION:

import java.util.Scanner;

public class EvenOrOdd {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter an integer: ");

int number = scanner.nextInt();

if (number % 2 == 0) {

System.out.println("The number is even.");

} else {

System.out.println("The number is odd.");

       }

    }

}

13. Sum of Numbers: Write a program that asks the user for two numbers and prints their sum.

SOLUTION:

import java.util.Scanner;

public class SumOfNumbers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Read the first number from the user

System.out.println("Enter the first number: ");

int firstNumber = scanner.nextInt();

System.out.println("Enter the second number: ");

int secondNumber = scanner.nextInt();

int sum = firstNumber + secondNumber;

System.out.println("The sum of the two numbers is: " + sum);

    }

}

14. Factorial: Write a program that calculates the factorial of a given number. The factorial of a non-negative integer n is the product of all positive integers less than or equal to n.

SOLUTION:

import java.util.Scanner;

public class Factorial {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a non-negative integer: ");

int number = scanner.nextInt();

int factorial = 1;

for (int i = 1; i <= number; i++)

{

factorial \*= i;

}

System.out.println("The factorial of " + number + " is " + factorial);

    }

}

15. Fibonacci Sequence: Write a program that prints the Fibonacci sequence up to a given number of terms. The Fibonacci sequence starts with 0 and 1, and each subsequent term is the sum of the two previous terms.

SOLUTION:

import java.util.Scanner;

public class FibonacciSequence {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number of terms: ");

int numberOfTerms = scanner.nextInt();

int firstTerm = 0;

int secondTerm = 1;

System.out.println(firstTerm);

System.out.println(secondTerm);

for (int i = 2; i < numberOfTerms; i++) {

int nextTerm = firstTerm + secondTerm;

System.out.println(nextTerm);

firstTerm = secondTerm;

secondTerm = nextTerm;

       }

    }

}

16. Prime Numbers: Write a program that checks whether a given number is prime or not. A prime number is a number greater than 1 that has no divisors other than 1 and itself.

SOLUTION:

import java.util.Scanner;

public class PrimeNumberChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a number: ");

int number = scanner.nextInt();

boolean isPrime = true;

for (int i = 2; i <= Math.sqrt(number); i++) {

if (number % i == 0) {

isPrime = false;

break;

}

}

if (isPrime) {

System.out.println(number + " is a prime number.");

} else {

System.out.println(number + " is not a prime number.");

       }

    }

}

17. Palindrome Checker: Write a program that checks whether a given string is a palindrome. A palindrome is a word, phrase, number, or other sequence of characters that reads the same forward and backward.

SOLUTION:

import java.util.Scanner;

public class PalindromeChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a string: ");

String string = scanner.nextLine();

boolean isPalindrome = true;

for (int i = 0; i < string.length() / 2; i++) {

if (string.charAt(i) != string.charAt(string.length() - i - 1)) {

isPalindrome = false;

break;

}

}

if (isPalindrome) {

System.out.println(string + " is a palindrome.");

} else {

System.out.println(string + " is not a palindrome.");

}

}

}

18. Rock, Paper, Scissors: Write a program that allows the user to play Rock, Paper, Scissors against the computer. The program should randomly generate the computer's choice and determine the winner based on the rules of the game.

SOLUTION:

import java.util.Random;

import java.util.Scanner;

public class RockPaperScissors {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Random random = new Random();

// Get the user's choice

System.out.println("Enter your choice (rock, paper, or scissors): ");

String userChoice = scanner.nextLine();

// Generate the computer's choice

String computerChoice = random.nextInt(3) == 0 ? "rock" : random.nextInt(3) == 1 ? "paper" : "scissors";

// Determine the winner

String winner = "";

if (userChoice.equals(computerChoice)) {

winner = "Tie";

} else if (userChoice.equals("rock") && computerChoice.equals("paper")) {

winner = "Computer";

} else if (userChoice.equals("rock") && computerChoice.equals("scissors")) {

winner = "User";

} else if (userChoice.equals("paper") && computerChoice.equals("rock")) {

winner = "User";

} else if (userChoice.equals("paper") && computerChoice.equals("scissors")) {

winner = "Computer";

} else if (userChoice.equals("scissors") && computerChoice.equals("rock")) {

winner = "Computer";

} else if (userChoice.equals("scissors") && computerChoice.equals("paper")) {

winner = "User";

}

// Print the result

System.out.println("You chose " + userChoice + ".");

System.out.println("The computer chose " + computerChoice + ".");

System.out.println("The winner is " + winner + ".");

    }

}

19. Number Guessing Game: Write a program that generates a random number between 1 and 100 and asks the user to guess the number. The program should provide feedback on whether the guess is too high or too low until the correct number is guessed.

SOLUTION:

import java.util.Random;

import java.util.Scanner;

public class NumberGuessingGame {

public static void main(String[] args) {

// Generate a random number between 1 and 100

Random random = new Random();

int randomNumber = random.nextInt(100) + 1;

// Get the user's guess

Scanner scanner = new Scanner(System.in);

System.out.println("Enter your guess (1-100): ");

int guess = scanner.nextInt();

while (guess != randomNumber) {

if (guess < randomNumber) {

System.out.println("Your guess is too low.");

}

Else

{

System.out.println("Your guess is too high.");

}

System.out.println("Enter your guess (1-100): ");

guess = scanner.nextInt();

}

System.out.println("Congratulations! You guessed the correct number.");

    }

}

20. Reverse a String: Write a program that takes a string as input and prints the reverse of the string.

SOLUTION:

import java.util.Scanner;

public class ReverseString {

public static void main(String[] args) {

// Get the input string from the user

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a string: ");

String inputString = scanner.nextLine();

// Reverse the string

String reversedString = "";

for (int i = inputString.length() - 1; i >= 0; i--) {

reversedString += inputString.charAt(i);

}

// Print the reversed string

System.out.println("The reversed string is: " + reversedString);

    }

}

21. Word Count: Write a program that reads a text file and counts the number of occurrences of each word in the file. The program should then print the word and its count in descending order of frequency.

SOLUTION:

import java.io.File;

import java.io.FileNotFoundException;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

public class WordCount {

public static void main(String[] args) {

// Read the text file

File file = new File("text.txt");

Scanner scanner = null;

try {

scanner = new Scanner(file);

} catch (FileNotFoundException e) {

System.out.println("Error reading the file.");

return;

}

// Count the number of occurrences of each word

Map<String, Integer> wordCounts = new HashMap<>();

while (scanner.hasNext()) {

String word = scanner.next();

if (wordCounts.containsKey(word)) {

wordCounts.put(word, wordCounts.get(word) + 1);

} else {

wordCounts.put(word, 1);

}

}

// Print the word and its count in descending order of frequency

for (Map.Entry<String, Integer> entry : wordCounts.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

       }

    }

}

22. Anagram Checker: Write a program that takes two strings as input and checks whether they are anagrams of each other. Anagrams are words or phrases formed by rearranging the letters of another word or phrase.

SOLUTION:

import java.util.Arrays;

import java.util.Scanner;

public class AnagramChecker {

public static void main(String[] args) {

// Get the two strings from the user

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the first string:");

String firstString = scanner.nextLine();

System.out.println("Enter the second string:");

String secondString = scanner.nextLine();

// Check if the strings are anagrams

boolean areAnagrams = areAnagrams(firstString, secondString);

if (areAnagrams) {

System.out.println("The strings are anagrams.");

} else {

System.out.println("The strings are not anagrams.");

}

}

private static boolean areAnagrams(String firstString, String secondString) {

firstString = firstString.toLowerCase();

secondString = secondString.toLowerCase();

char[] firstStringChars = firstString.toCharArray();

char[] secondStringChars = secondString.toCharArray();

Arrays.sort(firstStringChars);

Arrays.sort(secondStringChars);

return Arrays.equals(firstStringChars, secondStringChars);

    }

}

23. Matrix Multiplication: Write a program that multiplies two matrices of arbitrary size. The program should take two matrices as input and output the resulting matrix.

SOLUTION:

import java.util.Scanner;

public class MatrixMultiplication {

public static void main(String[] args) {

// Get the two matrices from the user

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number of rows in the first matrix:");

int numRows1 = scanner.nextInt();

System.out.println("Enter the number of columns in the first matrix:");

int numCols1 = scanner.nextInt();

System.out.println("Enter the elements of the first matrix:");

int[][] matrix1 = new int[numRows1][numCols1];

for (int i = 0; i < numRows1; i++) {

for (int j = 0; j < numCols1; j++) {

matrix1[i][j] = scanner.nextInt();

}

}

System.out.println("Enter the number of rows in the second matrix:");

int numRows2 = scanner.nextInt();

System.out.println("Enter the number of columns in the second matrix:");

int numCols2 = scanner.nextInt();

System.out.println("Enter the elements of the second matrix:");

int[][] matrix2 = new int[numRows2][numCols2];

for (int i = 0; i < numRows2; i++) {

for (int j = 0; j < numCols2; j++) {

matrix2[i][j] = scanner.nextInt();

}

}

// Check if the matrices can be multiplied

if (numCols1 != numRows2) {

System.out.println("The matrices cannot be multiplied.");

return;

}

// Multiply the matrices

int[][] productMatrix = new int[numRows1][numCols2];

for (int i = 0; i < numRows1; i++) {

for (int j = 0; j < numCols2; j++) {

for (int k = 0; k < numCols1; k++) {

productMatrix[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

24. Password Generator: Write a program that generates a random password for the user. The password should be a combination of uppercase and lowercase letters, numbers, and special characters. The user should be able to specify the length of the password.

SOLUTION:

import java.util.Random;

import java.util.Scanner;

public class PasswordGenerator {

public static void main(String[] args) {

// Get the length of the password from the user

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the length of the password you want:");

int length = scanner.nextInt();

// Generate a random password

String password = generatePassword(length);

// Print the generated password

System.out.println("Your password is: " + password);

}

private static String generatePassword(int length) {

// Create a string of all the possible characters

String characters = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&\*()";

// Create a random object

Random random = new Random();

// Create a string buffer to store the password

StringBuffer password = new StringBuffer();

// Generate a random password

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

int index = random.nextInt(characters.length());

char character = characters.charAt(index);

password.append(character);

}

// Return the generated password

return password.toString();

    }

}

25. Hangman Game: Write a program that allows the user to play the game of Hangman. The program should randomly select a word from a predefined list and prompt the user to guess letters until they either solve the word or run out of guesses.

SOLUTION:

import java.util.Random;

import java.util.Scanner;

public class Hangman {

public static void main(String[] args) {

// Create a list of words to choose from

String[] words = {"apple", "banana", "cherry", "dog", "elephant", "fish", "grape", "horse", "ice cream", "juice", "kiwi", "lemon", "mango", "nut", "orange", "peach", "pear", "pineapple", "queen", "rabbit", "strawberry", "tiger", "umbrella", "violin", "watermelon", "x-ray", "yacht", "zebra"};

// Choose a random word from the list

Random random = new Random();

String word = words[random.nextInt(words.length)];

// Create a string to store the user's guesses

String guesses = "";

// Create a variable to store the number of guesses the user has left

int guessesLeft = 6;

// Start the game loop

while (guessesLeft > 0) {

// Display the word with the letters the user has guessed so far

System.out.println("Word: " + getGuessedWord(word, guesses));

// Prompt the user to guess a letter

System.out.println("Guess a letter:");

Scanner scanner = new Scanner(System.in);

String guess = scanner.nextLine();

// Check if the user's guess is correct

if (word.contains(guess)) {

// Add the letter to the list of guesses

guesses += guess;

// Check if the user has solved the word

if (getGuessedWord(word, guesses).equals(word)) {

System.out.println("You win!");

break;

}

} else {

// The user's guess was incorrect

guessesLeft--;

System.out.println("Incorrect guess. You have " + guessesLeft + " guesses left.");

}

}

// The user ran out of guesses

if (guessesLeft == 0) {

System.out.println("You lose!");

}

}

26. File Encryption/Decryption: Write a program that encrypts or decrypts a file using a simple encryption algorithm. The program should take a file and a secret key as input and output the encrypted or decrypted file.

SOLUTION:

27. Web Scraper: Write a program that extracts data from a website of your choice. The program should scrape the web page, extract relevant information, and save it to a file or display it to the user.

SOLUTION:

import java.io.IOException;

import java.net.MalformedURLException;

import java.net.URL;

import java.util.Scanner;

public class WebScraper {

public static void main(String[] args) {

try {

// Define the URL of the website you want to scrape

URL url = new URL("https://www.nytimes.com/");

// Get the HTML content of the website

Scanner scanner = new Scanner(url.openStream());

String htmlContent = "";

while (scanner.hasNext()) {

htmlContent += scanner.nextLine();

}

// Parse the HTML content using BeautifulSoup

// (assuming you have added the BeautifulSoup dependency to your project)

Document doc = Jsoup.parse(htmlContent);

// Find the title of the website

String title = doc.title();

// Find all the links on the website

Elements links = doc.select("a");

// Print the title and links of the website

System.out.println("Title: " + title);

System.out.println("Links:");

for (Element link : links) {

System.out.println(link.text() + " - " + link.attr("href"));

}

} catch (MalformedURLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

}

28. Sudoku Solver: Write a program that solves a Sudoku puzzle. The program should take a partially filled Sudoku grid as input and output the solution.

SOLUTION:

public class Sudoku {

static int N = 9;

static boolean solveSudoku(int grid[][], int row,

int col)

{

if (row == N - 1 && col == N)

return true;

if (col == N) {

row++;

col = 0;

}

if (grid[row][col] != 0)

return solveSudoku(grid, row, col + 1);

for (int num = 1; num < 10; num++) {

if (isSafe(grid, row, col, num)) {

grid[row][col] = num;

if (solveSudoku(grid, row, col + 1))

return true;

}

grid[row][col] = 0;

}

return false;

}

static void print(int[][] grid)

{

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

for (int j = 0; j < N; j++)

System.out.print(grid[i][j] + " ");

System.out.println();

}

}

static boolean isSafe(int[][] grid, int row, int col,

int num)

{

for (int x = 0; x <= 8; x++)

if (grid[row][x] == num)

return false;

for (int x = 0; x <= 8; x++)

if (grid[x][col] == num)

return false;

int startRow = row - row % 3, startCol

= col - col % 3;

for (int i = 0; i < 3; i++)

for (int j = 0; j < 3; j++)

if (grid[i + startRow][j + startCol] == num)

return false;

return true;

}

public static void main(String[] args)

{

int grid[][] = { { 3, 0, 6, 5, 0, 8, 4, 0, 0 },

{ 5, 2, 0, 0, 0, 0, 0, 0, 0 },

{ 0, 8, 7, 0, 0, 0, 0, 3, 1 },

{ 0, 0, 3, 0, 1, 0, 0, 8, 0 },

{ 9, 0, 0, 8, 6, 3, 0, 0, 5 },

{ 0, 5, 0, 0, 9, 0, 6, 0, 0 },

{ 1, 3, 0, 0, 0, 0, 2, 5, 0 },

{ 0, 0, 0, 0, 0, 0, 0, 7, 4 },

{ 0, 0, 5, 2, 0, 6, 3, 0, 0 } };

if (solveSudoku(grid, 0, 0))

print(grid);

else

System.out.println("No Solution exists");

}

}

29. Image Processing: Write a program that performs basic image processing operations, such as resizing, cropping, and applying filters, on an image file.

SOLUTION:

30. Tic-Tac-Toe AI: Write a program that allows the user to play Tic-Tac-Toe against an AI opponent. The AI should use a strategy to make intelligent moves and try to win the game.

SOLUTION:

import java.util.Scanner;

public class TicTacToe {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Create the board

char[][] board = new char[3][3];

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

board[i][j] = ' ';

}

}

// Print the board

printBoard(board);

// Get the player's move

System.out.println("Enter your move (1-9): ");

int playerMove = scanner.nextInt();

// Make the player's move

board[playerMove / 3][playerMove % 3] = 'X';

// Print the board

printBoard(board);

// Check if the player has won

if (hasWon(board, 'X')) {

System.out.println("You win!");

return;

}

// Make the AI's move

int aiMove = getAIMove(board);

// Make the AI's move

board[aiMove / 3][aiMove % 3] = 'O';

// Print the board

printBoard(board);

// Check if the AI has won

if (hasWon(board, 'O')) {

System.out.println("The AI wins!");

return;

}

// Check if the game is a draw

if (isDraw(board)) {

System.out.println("The game is a draw!");

return;

}

// Get the next player move

System.out.println("Enter your move (1-9): ");

playerMove = scanner.nextInt();

// Make the player's move

board[playerMove / 3][playerMove % 3] = 'X';

// Print the board

printBoard(board);

// Check if the player has won

if (hasWon(board, 'X')) {

System.out.println("You win!");

return;

}

// Make the AI's

31. Word Frequency Counter: Write a program that takes a sentence as input and counts the frequency of each word in the sentence. Store the word frequencies in a dictionary and print the result.

SOLUTION:

import java.util.\*;

public class WordFrequencyCounter {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Get the sentence from the user

System.out.println("Enter a sentence: ");

String sentence = scanner.nextLine();

// Split the sentence into words

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

// Create a dictionary to store the word frequencies

Map<String, Integer> wordFrequencies = new HashMap<>();

// Count the frequency of each word

for (String word : words) {

if (wordFrequencies.containsKey(word)) {

wordFrequencies.put(word, wordFrequencies.get(word) + 1);

} else {

wordFrequencies.put(word, 1);

}

}

// Print the word frequencies

System.out.println("Word frequencies:");

for (Map.Entry<String, Integer> entry : wordFrequencies.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

32. Unique Elements: Write a program that takes an array of integers as input and returns a new array containing only the unique elements from the original array. The order of the elements in the new array should be the same as in the original array.

SOLUTION:

import java.util.\*;

public class UniqueElements {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Get the input array

System.out.println("Enter the array elements: ");

int[] inputArray = new int[scanner.nextInt()];

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

inputArray[i] = scanner.nextInt();

}

// Create a set to store the unique elements

Set<Integer> uniqueElements = new HashSet<>();

for (int element : inputArray) {

uniqueElements.add(element);

}

// Convert the set back to an array

int[] uniqueArray = new int[uniqueElements.size()];

int index = 0;

for (int element : uniqueElements) {

uniqueArray[index++] = element;

}

// Print the unique array

System.out.println("Unique array: " + Arrays.toString(uniqueArray));

}

}

33. Dictionary Manipulation: Write a program that creates a dictionary representing a person's contact information, including their name, email address, and phone number. Allow the user to add, update, or delete entries in the dictionary.

SOLUTION:

import java.util.\*;

public class ContactDictionary {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Create a dictionary to store the contact information

Map<String, String> contacts = new HashMap<>();

// Add a new contact

System.out.println("Enter the name of the contact: ");

String name = scanner.nextLine();

System.out.println("Enter the email address of the contact: ");

String email = scanner.nextLine();

System.out.println("Enter the phone number of the contact: ");

String phone = scanner.nextLine();

contacts.put(name, email + ", " + phone);

// Update a contact

System.out.println("Enter the name of the contact you want to update: ");

name = scanner.nextLine();

if (contacts.containsKey(name)) {

System.out.println("Enter the new email address of the contact: ");

email = scanner.nextLine();

System.out.println("Enter the new phone number of the contact: ");

phone = scanner.nextLine();

contacts.put(name, email + ", " + phone);

} else {

System.out.println("Contact not found.");

}

// Delete a contact

System.out.println("Enter the name of the contact you want to delete: ");

name = scanner.nextLine();

if (contacts.containsKey(name)) {

contacts.remove(name);

} else {

System.out.println("Contact not found.");

}

// Print the dictionary

System.out.println("Contacts:");

for (Map.Entry<String, String> entry : contacts.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

34. Array Sorting: Write a program that takes an array of numbers as input and sorts the array in ascending order. You can use any sorting algorithm of your choice (e.g., bubble sort, insertion sort, or selection sort).

SOLUTION:

35. Word Anagrams: Write a program that takes a list of words as input and groups together words that are anagrams of each other. The program should return a dictionary where the keys are sorted strings of the characters in each word, and the values are lists of words that are anagrams.

SOLUTION: