1. Write a Java program to take input as a command line argument: your name, course, university roll no, and semester. Display the information.

public class CommandLineInput {

public static void main(String[] args){

String name = args[0];

String course = args[1];

int rollNo = Integer.parseInt(args[2]);

int semester = Integer.parseInt(args[3]);

System.out.println("Name: " + name);

System.out.println("Course: " + course);

System.out.println("Roll No: " + rollNo);

System.out.println("Semester: " + semester);

}

}

2. Using the switch statement, write a menu-driven program to calculate the maturity amount of a bank deposit for Term Deposit and Recurring Deposit options.

import java.util.Scanner;

public class BankDepositCalculator {

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

System.out.println("Choose option:

1. Term Deposit

2. Recurring Deposit");

int choice = sc.nextInt();

switch (choice){

case 1:

System.out.println("Enter Principal, Rate of Interest, and Time (years):");

double p = sc.nextDouble();

double r = sc.nextDouble();

int n = sc.nextInt();

double a = p \* Math.pow((1 + r / 100), n);

System.out.println("Maturity Amount:" + a);

break;

case 2:

System.out.println("Enter Monthly Installment, Rate of Interest, and Time

(months):");

double mp = sc.nextDouble();

double mr = sc.nextDouble();

int mn = sc.nextInt();

double ma = mp \* mn + mp \* mn \* (mn + 1) / 2.0 \* mr / 100 \* 1 / 12.0;

System.out.println("Maturity Amount: " + ma);

break;

default:

System.out.println("Invalid option selected.");

}

sc.close();

}

}

3. Write a Java program to check whether two numbers are Friendly Pairs or not. A Friendly Pair has equal abundance: (sum of divisors / number) is the same for both.

import java.util.Scanner;

public class FriendlyPair {

public static int sumOfDivisors(int num){

int sum = 0;

for (int i = 1; i <= num / 2; i++){

if (num % i == 0)

sum += i;

}

return sum;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter two numbers:");

int num1 = sc.nextInt();

int num2 = sc.nextInt();

int sum1 = sumOfDivisors(num1);

int sum2 = sumOfDivisors(num2);

double ratio1 = (double) sum1 / num1;

double ratio2 = (double) sum2 / num2;

if (Math.abs(ratio1 - ratio2) < 1e-6)

System.out.println("Friendly Pair");

else

System.out.println("Not Friendly Pair");

sc.close();

}

}

4. Write a Java program to replace all 0's with 1 in a given integer.

import java.util.Scanner;

public class ReplaceZeroWithOne {

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

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

int num = sc.nextInt();

String result = Integer.toString(num).replace('0', '1');

System.out.println("Modified number: " + result);

sc.close();

}

}

5. Write a Java program to print an array in Zigzag fashion: a < b > c < d > e < f.

import java.util.Scanner;

public class ZigZagArray {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) arr[i] = sc.nextInt();

boolean flag = true;

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

if (flag) {

if(arr[i] > arr[i + 1]){

int temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

}

} else {

if(arr[i] < arr[i + 1]){

int temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

}

}

flag = !flag;

}

for (inti : arr) System.out.print(i + "");

sc.close();

}

}

6. Write a Java program to rearrange positive and negative numbers in an array. Move negatives to the beginning.

import java.util.Scanner;

public class RearrangeArray {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int[] arr = new int[6];

for (int i = 0; i < 6; i++) arr[i] = sc.nextInt();

int j = 0;

for (int i = 0; i < 6; i++){

if (arr[i] < 0) {

inttemp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

j++;

}

}

for (int i : arr) System.out.print(i + "");

sc.close();

}

}

7. Write a Java program to find the saddle point in a matrix. It should be the min in its row and max in its column.

import java.util.Scanner;

public class SaddlePoint{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int[][] mat = new int[3][3];

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

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

mat[i][j] = sc.nextInt();

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

int minRow= mat[i][0];

int colIdx = 0;

for (int j = 1; j < 3; j++) {

if(mat[i][j] < minRow){

minRow = mat[i][j];

colIdx = j;

}

}

boolean isSaddle = true;

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

if(mat[k][colIdx] > minRow){

isSaddle = false;

break;

}

}

if(isSaddle){

System.out.println("Saddle Point:" + minRow + " at (" + i + "," + colIdx + ")");

sc.close();

return;

}

}

System.out.println("No Saddle Point Found");

sc.close();

}

}

8. Write a Java program to count all 0(1+)0 patterns in a binary string using String class.

import java.util.Scanner;

public class PatternCount {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s = sc.next();

int count = 0;

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

if (s.charAt(i) == '0') {

int j = i + 1;

while (j < s.length() && s.charAt(j) == '1') j++;

if (j < s.length() && s.charAt(j) == '0' && j > i + 1)

count++;

}

}

System.out.println("Count:" + count);

sc.close();

}

}

9. Write a Java program to delete vowels from a string using StringBuffer class.

import java.util.Scanner;

public class DeleteVowels {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String input = sc.nextLine();

StringBuffer sb = new StringBuffer();

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

char ch = input.charAt(i);

if (!"AEIOUaeiou".contains(ch + ""))

sb.append(ch);

}

System.out.println("Output: " + sb);

sc.close();

}

}

10. Write a Java program to create a class 'Bank' with account details. Include features like deposit, withdraw, and change address.

import java.util.Scanner;

class Bank {

String name, address;

int accNo;

double balance;

static int count = 1000;

Bank(String name, String address, double balance) {

this.name = name;

this.address = address;

this.balance = balance;

this.accNo = ++count;

}

void display(){

System.out.println("Name:" + name + "\nAddress:" + address +

"\nAccNo: " + accNo + "\nBalance: " + balance);

}

void deposit(double amt){

balance += amt;

}

void withdraw(double amt){

if (amt <= balance)

balance -= amt;

}

void changeAddress(String newAddress){

this.address = newAddress;

}

}

public class BankTest {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

Bank[] customers = new Bank[2];

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

System.out.println("Enter name, address and balance:");

String name = sc.next();

String address = sc.next();

double balance = sc.nextDouble();

customers[i] = new Bank(name, address, balance);

}

customers[0].deposit(5000);

customers[0].withdraw(2000);

customers[0].changeAddress("NewAddress");

customers[0].display();

sc.close();

}

}

11. Define a class WordExample having the following description: Data members/instance variables: Private String strdata: to store a sentence. Parameterized Constructor WordExample(String): Accept a sentence which may be terminated by either ‘.’, ‘?’ or ‘!’ only. The words may be separated by more than one blank space and are in UPPER CASE. Member Methods: void countWord(): Find the number of words beginning and ending with a vowel. void placeWord(): Place the words which begin and end with a vowel at the beginning, followed by the remaining words as they occur in the sentence.

import java.util.\*;

public class WordExample {

private String strdata;

publicWordExample(String s){

strdata = s.trim();

}

public void countWord(){

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

int count = 0;

for (String word : words) {

word = word.toUpperCase();

if ("AEIOU".indexOf(word.charAt(0)) != -1 &&

"AEIOU".indexOf(word.charAt(word.length()- 1))!= -1){

count++;

}

}

System.out.println("Number of words beginning and ending with a vowel:" + count);

}

public void placeWord(){

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

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

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

for (String word : words) {

word = word.toUpperCase();

if ("AEIOU".indexOf(word.charAt(0)) != -1 &&

"AEIOU".indexOf(word.charAt(word.length()- 1))!= -1){

vowelWords.add(word);

} else {

otherWords.add(word);

}

}

vowelWords.addAll(otherWords);

System.out.println("Rearranged Sentence: " + String.join(" ", vowelWords));

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter sentence in UPPER CASE ending with '.', '?' or '!'");

String s = sc.nextLine();

WordExample obj = newWordExample(s);

obj.countWord();

obj.placeWord();

sc.close();

}

}

12. Method Overloading (Compile time Polymorphism): Create a class called ArrayDemo and overload arrayFunc()function. 1. void arrayFunc(int [], int) – Find all pairs of elements in an array whose sum is equal to a given number. 2. void arrayFunc(int A[], int p, int B[], int q) – Merge two sorted arrays maintaining order and filling A with smallest p and B with remaining.

import java.util.\*;

public class ArrayDemo {

void arrayFunc(int[] arr, inttarget){

System.out.println("Pairs of elements whose sum is " + target + " are:");

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

for (int j = i + 1; j < arr.length; j++){

if (arr[i] + arr[j] == target) {

System.out.println(arr[i] + " + " + arr[j] + " = " + target);

}

}

}

}

void arrayFunc(int[] A, int p, int[] B, int q){

int[] merged = new int[p + q];

System.arraycopy(A, 0, merged, 0, p);

System.arraycopy(B, 0, merged, p, q);

Arrays.sort(merged);

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

A[i] = merged[i];

}

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

B[i] = merged[i + p];

}

System.out.println("Sorted Arrays:");

System.out.println("A: " + Arrays.toString(A));

System.out.println("B:" + Arrays.toString(B));

}

public static void main(String[] args){

ArrayDemo obj = new ArrayDemo();

int[] numbers = {4, 6, 5, -10, 8, 5, 20};

obj.arrayFunc(numbers, 10);

int[] A = {1, 5, 6, 7, 8, 10};

int[] B = {2, 4, 9};

obj.arrayFunc(A, A.length, B, B.length);

}

}

13. Method overriding (Runtime Polymorphism), Abstract class and Abstract method, Inheritance, interfaces: Write a java program to calculate the area of a rectangle, a square and a circle. Create an abstract class 'Shape' with three abstract methods namely rectangleArea() taking two parameters, squareArea() and circleArea() taking one parameter each. Now create another class ‘Area’ containing all the three methods rectangleArea(), squareArea() and circleArea() for printing the area of rectangle, square and circle respectively. Create an object of class Area and call all the three methods.

abstract class Shape {

abstract void rectangleArea(double l, double b);

abstract void squareArea(double side);

abstract void circleArea(double radius);

}

class Area extends Shape {

void rectangleArea(double l, double b) {

System.out.println("Rectangle Area:" + (l \* b));

}

void squareArea(double side) {

System.out.println("Square Area:" + (side \* side));

}

void circleArea(double radius) {

System.out.println("CircleArea:" + (Math.PI \* radius \* radius));

}

public static void main(String[] args){

Area a = new Area();

a.rectangleArea(5, 3);

a.squareArea(4);

a.circleArea(2.5);

}

}

14. Write a java program to implement abstract class and abstract method with following details: Create a abstract Base Class Temperature Data members: double temp; Method members: void setTempData(double) abstract void changeTemp() Sub Class Fahrenheit(subclass of Temperature) Override abstract method changeTemp()to convert Fahrenheit temperature into degree Celsius by using formula C=5/9\*(F-32) and display converted temperature Sub Class Celsius (subclass of Temperature) Override abstract method changeTemp()to convert degree Celsius into Fahrenheit temperature by using formula F=9/5\*c+32 and display converted temperature

abstract class Temperature {

double temp;

void setTempData(double t){

temp = t;

}

abstract void changeTemp();

}

class Fahrenheit extends Temperature {

void changeTemp() {

double c = (5.0/9) \* (temp - 32);

System.out.println("Celsius:" + c);

}

}

class Celsius extends Temperature {

void changeTemp() {

double f = (9.0/5) \* temp + 32;

System.out.println("Fahrenheit: " + f);

}

public static void main(String[] args){

Fahrenheit f = new Fahrenheit();

f.setTempData(98.6);

f.changeTemp();

Celsius c = new Celsius();

c.setTempData(37);

c.changeTemp();

}

}

15. Write a Java program to create an interface that consists of a method to display volume() as an abstract method and redefine this method in the derived classes to suit their requirements. Create classes called Cone, Hemisphere and Cylinder that implements the interface. Using these three classes, design a program that will accept dimensions of a cone, cylinder and hemisphere interactively and display the volumes. Volume of cone = (1/3)πr2h Volume of hemisphere = (2/3)πr3 Volume of cylinder = πr2h

import java.util.Scanner;

interface Volume {

void displayVolume();

}

class Cone implements Volume {

private double radius;

private double height;

public Cone(double radius, double height){

this.radius = radius;

this.height = height;

}

@Override

public void displayVolume() {

double volume = (1.0 / 3.0) \* Math.PI \* radius \* radius \* height;

System.out.println("Volume of Cone:" + String.format("%.2f", volume));

}

}

classHemisphere implements Volume {

private double radius;

public Hemisphere(double radius){

this.radius = radius;

}

@Override

public void displayVolume() {

double volume = (2.0 / 3.0) \* Math.PI \* radius \* radius \* radius;

System.out.println("Volume ofHemisphere:" + String.format("%.2f", volume));

}

}

class Cylinder implements Volume {

private double radius;

private double height;

public Cylinder(double radius, double height){

this.radius = radius;

this.height = height;

}

@Override

public void displayVolume() {

double volume = Math.PI \* radius \* radius \* height;

System.out.println("Volume of Cylinder:" + String.format("%.2f", volume));

}

}

public class VolumeCalculator {

public static void main(String

Scanner scanner = new Scanner(System.in);

System.out.println("Enter dimensions for Cone:");

System.out.print("Enter radius: ");

double coneRadius = scanner.nextDouble();

System.out.print("Enter height: ");

double coneHeight = scanner.nextDouble();

Cone cone = new Cone(coneRadius, coneHeight);

cone.displayVolume();

System.out.println("\nEnter dimensions for Hemisphere:");

System.out.print("Enter radius: ");

double hemiRadius = scanner.nextDouble();

Hemisphere hemisphere = new Hemisphere(hemiRadius);

hemisphere.displayVolume();

System.out.println("\nEnter dimensions for Cylinder:");

System.out.print("Enter radius: ");

double cylinderRadius = scanner.nextDouble();

System.out.print("Enter height: ");

double cylinderHeight = scanner.nextDouble();

Cylinder cylinder = new Cylinder(cylinderRadius, cylinderHeight);

cylinder.displayVolume();

scanner.close();

}

}

16. : Write a Java program to accept and print the employee details during runtime. The details will include employee id, name, dept\_Id. The program should raise an exception if user inputs incomplete or incorrect data. The entered value should meet the following conditions: a. First Letter of employee name should be in capital letter. b. Employee id should be between 2001 and 5001. c. Department id should be an integer between 1 and 5. If the above conditions are not met, then the application should raise specific exception else should complete normal execution.

importjava.util.InputMismatchException;

import java.util.Scanner;

class InvalidEmployeeDataException extends Exception {

public InvalidEmployeeDataException(String message) {

super(message);

}

}

class Employee {

private int employeeId;

private String name;

private int deptId;

public Employee(int employeeId, String name, int deptId)throws

InvalidEmployeeDataException {

if (name == null || name.isEmpty() || !Character.isUpperCase(name.charAt(0))) {

throw new InvalidEmployeeDataException("First letter of employee name must be in

capital letter.");

}

if(employeeId < 2001 || employeeId > 5001){

throw new InvalidEmployeeDataException("Employee ID must be between 2001 and

5001.");

}

if(deptId < 1 || deptId > 5) {

throw new InvalidEmployeeDataException("Department ID must be an integer

between 1 and 5.");

}

this.employeeId = employeeId;

this.name = name;

this.deptId = deptId;

}

public void displayDetails() {

System.out.println("Employee ID:" + employeeId);

System.out.println("Name: " + name);

System.out.println("Department ID: " + deptId);

}

}

public class EmployeeManagement{

public static void main(String

Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter Employee ID: ");

int id = scanner.nextInt();

scanner.nextLine();

System.out.print("Enter Employee Name: ");

String name = scanner.nextLine();

System.out.print("Enter Department ID: ");

int dept = scanner.nextInt();

Employee employee = new Employee(id, name, dept);

System.out.println("\nEmployee details accepted successfully:");

employee.displayDetails();

} catch (InputMismatchException e) {

System.err.println("Error: Invalid input type. Please enter correct data types.");

} catch (InvalidEmployeeDataException e) {

System.err.println("Error: " + e.getMessage());

} finally {

scanner.close();

}

}

}

17. Create a class MyCalculator which consists of a single method power (int, int). This method takes two integers, n and p, as parameters and finds np. If either n or p is negative, then the method must throw an exception which says, "n and p should be nonnegative". Input Format: Each line of the input contains two integers, n and p. Output Format: Each line of the output contains the result, if neither of n and p is negative. Otherwise, the output contains "n and p should be non- negative". Sample Input: 3 5 2 4 0 0 -1 -2 -1 3 Sample Output: 243 16 java.lang.Exception: n and p should not be zero. java.lang.Exception: n or p should not be negative. java.lang.Exception: n or p should not be negative. Explanation: In the first two cases, both n and p are positive. So, the power function returns the answer correctly. In the third case, both n and p are zero. So, the exception, "n and p should not be zero.” is printed. In the last two cases, at least one out of n and p is negative. So, the exception, "n or p should not be negative.” is printed for these two cases.

import java.util.Scanner;

class MyCalculator {

public long power(int n, int p)throws Exception {

if (n < 0 || p < 0) {

throw new Exception("n or p should not be negative.");

}

if (n == 0 && p == 0) {

throw new Exception("n and p should not be zero.");

}

return (long) Math.pow(n, p);

}

}

public class Solution {

public static void main(String

Scanner in = new Scanner(System.in);

MyCalculator myCalculator = new MyCalculator();

while (in.hasNextInt()){

int n = in.nextInt();

int p = in.nextInt();

try {

System.out.println(myCalculator.power(n, p));

} catch (Exception e){

System.out.println(e.toString());

}

}

in.close();

}

}

18. Write a Java file handling program to count and display the number of palindromes present in a text file "myfile.txt". Example: If the file "myfile.txt" contains the following lines, My name is NITIN Hello aaa and bbb word How are You ARORA is my friend Output will be => 4.

importjava.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class PalindromeCounter {

public static boolean isPalindrome(String str){

if (str == null || str.isEmpty()) {

return false;

}

String cleanedStr = str.replaceAll("

if(cleanedStr.isEmpty()){

return false;

}

int left = 0;

int right = cleanedStr.length() - 1;

while (left < right) {

if(cleanedStr.charAt(left)!= cleanedStr.charAt(right)){

return false;

}

left++;

right--;

}

return true;

}

public static void main(String

String fileName = "myfile.txt";

int palindromeCount = 0;

try (BufferedReader reader = new BufferedReader(new FileReader(fileName))){

String line;

while ((line = reader.readLine())!= null){

String

for (String word : words){

if (isPalindrome(word)) {

palindromeCount++;

}

}

}

System.out.println("Number of palindromes in \"" + fileName + "\":" +

palindromeCount);

} catch (IOException e){

System.err.println("Error reading file: " + e.getMessage());

}

}

}

19. Write a program MultiThreads that creates two threads-one thread with the name CSthread and the other thread named ITthread. Each thread should display its respective name and execute after a gap of 500 milliseconds. Each thread should also display a number indicating the number of times it got a chance to execute.

class CSthread extends Thread {

private int executionCount = 0;

public CSthread(){

super("CSthread");

}

@Override

public void run() {

while (executionCount < 10){

executionCount++;

System.out.println(getName() + " executed " + executionCount + "times.");

try {

Thread.sleep(500);

} catch (InterruptedException e) {

System.out.println(getName() + " interrupted.");

break;

}

}

}

}

class ITthread extends Thread {

private int executionCount = 0;

public ITthread(){

super("ITthread");

}

@Override

public void run() {

while (executionCount < 10){

executionCount++;

System.out.println(getName() + " executed " + executionCount + "times.");

try {

Thread.sleep(500);

} catch (InterruptedException e) {

System.out.println(getName() + " interrupted.");

break;

}

}

}

}

public class MultiThreads {

public static void main(String

CSthread csThread = newCSthread();

ITthread itThread = new ITthread();

csThread.start();

itThread.start();

try {

csThread.join();

itThread.join();

} catch (InterruptedException e) {

System.out.println("Main thread interrupted while waiting for child threads.");

}

System.out.println("Main thread finished.");

}

}

20. Write a Java program for to solve producer consumer problem in which a producer produces a value and consumer consume the value before producer generate the next value.

importjava.util.LinkedList;

import java.util.Queue;

import java.util.Random;

class SharedBuffer {

privateQueue<Integer> buffer = new LinkedList<>();

private final int CAPACITY = 1;

public void produce(intitem)throws InterruptedException {

synchronized (this) {

while (buffer.size() == CAPACITY) {

System.out.println("Buffer is full. Producer waiting...");

wait();

}

buffer.add(item);

System.out.println("Producer produced: " + item);

notify();

}

}

public int consume()throws InterruptedException {

synchronized (this) {

while (buffer.isEmpty()) {

System.out.println("Buffer is empty. Consumer waiting...");

wait();

}

int item = buffer.remove();

System.out.println("Consumer consumed:" + item);

notify();

return item;

}

}

}

class Producer extends Thread {

private SharedBuffer buffer;

private Random random = new Random();

public Producer(SharedBuffer buffer) {

this.buffer = buffer;

}

@Override

public void run() {

for (int i = 0; i < 5; i++) {

int item = random.nextInt(100);

try {

buffer.produce(item);

Thread.sleep(random.nextInt(500));

} catch (InterruptedException e) {

System.out.println("Producer interrupted.");

Thread.currentThread().interrupt();

}

}

}

}

class Consumer extends Thread {

private SharedBuffer buffer;

private Random random = new Random();

public Consumer(SharedBuffer buffer) {

this.buffer = buffer;

}

@Override

public void run() {

for (int i = 0; i < 5; i++){

try {

buffer.consume();

Thread.sleep(random.nextInt(500));

} catch (InterruptedException e) {

System.out.println("Consumer interrupted.");

Thread.currentThread().interrupt();

}

}

}

}

public class ProducerConsumerProblem {

public static void main(String

SharedBuffer buffer = newSharedBuffer();

Producer producer = new Producer(buffer);

Consumer consumer = new Consumer(buffer);

producer.start();

consumer.start();

try {

producer.join();

consumer.join();

} catch (InterruptedException e) {

System.out.println("Main thread interrupted during join.");

Thread.currentThread().interrupt();

}

System.out.println("Producer-Consumer simulation finished.");

}

}

21. Write a method removeEvenLength thattakes an ArrayList of Strings as a parameter and that removes allthe strings of even length from the list.(Use ArrayList)

importjava.util.ArrayList;

import java.util.Arrays;

public class ArrayListManipulator {

public static void removeEvenLength(ArrayList<String> list){

for (int i = list.size() - 1; i >= 0; i--) {

String currentString = list.get(i);

if(currentString.length() % 2 == 0){

list.remove(i);

}

}

}

public static void main(String[] args) {

ArrayList<String> list1 = new ArrayList<>(Arrays.asList("hello","world","java","code",

"program"));

System.out.println("Original list 1:" + list1);

removeEvenLength(list1);

System.out.println("List 1 after removing even length strings:" + list1);

System.out.println("\n---");

ArrayList<String> list2 = new ArrayList<>(Arrays.asList("four","score","and","seven",

"years", "ago"));

System.out.println("Original list 2:" + list2);

removeEvenLength(list2);

System.out.println("List 2 after removing even length strings:" + list2);

System.out.println("\n---");

ArrayList<String> list3 = new ArrayList<>(Arrays.asList("aa","bbbb","cccccc"));

System.out.println("Original list 3: " + list3);

removeEvenLength(list3);

System.out.println("List 3 after removing even length strings:" + list3);

System.out.println("\n---");

ArrayList<String> list4 = new ArrayList<>(Arrays.asList("a","bbb","eeeee"));

System.out.println("Original list 4: " + list4);

removeEvenLength(list4);

System.out.println("List 4 after removing even length strings:" + list4);

}

}

22. Write a method swapPairs that switches the order of values in an ArrayList of Strings in a pairwise fashion. Your method should switch the order of the first two values, then switch the order of the next two, switch the order of the next two, and so on. For example, if the list initially stores these values:{"four","score","and","seven","years", "ago"} your method should switch the first pair, "four", "score", the second pair, "and", "seven", and the third pair, "years", "ago", to yield this list: {"score", "four", "seven", "and", "ago", "years"} If there are an odd number of values in the list, the final element is not moved. For example, if the original list had been: {"to","be","or","not","to","be","hamlet"}It would again switch pairs of values, but the final value, "hamlet" would not be moved, yielding this list: {"be", "to", "not", "or", "be", "to", "hamlet"}

importjava.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class ListOperations {

public static void swapPairs(ArrayList<String> list){

for (int i = 0; i < list.size() - 1; i += 2) {

String first = list.get(i);

String second = list.get(i + 1);

list.set(i, second);

list.set(i + 1, first);

}

}

public static List<Integer> alternate(List<Integer> list1, List<Integer> list2){

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

int minLength = Math.min(list1.size(), list2.size());

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

result.add(list1.get(i));

result.add(list2.get(i));

}

if(list1.size() > minLength){

for (int i = minLength; i < list1.size(); i++){

result.add(list1.get(i));

}

} else if(list2.size() > minLength){

for (int i = minLength; i < list2.size(); i++){

result.add(list2.get(i));

}

}

return result;

}

public static void main(String[] args) {

System.out.println("--- Testing swapPairs method ---");

ArrayList<String> swapList1 = new ArrayList<>(Arrays.asList("four","score","and",

"seven", "years", "ago"));

System.out.println("Original swapList 1:" + swapList1);

swapPairs(swapList1);

System.out.println("Swapped swapList 1: " + swapList1);

System.out.println("\n---");

ArrayList<String> swapList2 = new ArrayList<>(Arrays.asList("to","be","or","not",

"to", "be", "hamlet"));

System.out.println("Original swapList 2 (odd length):" + swapList2);

swapPairs(swapList2);

System.out.println("Swapped swapList 2: " + swapList2);

System.out.println("\n--- Testing alternate method ---");

List<Integer> listA = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 5));

List<Integer> listB = new ArrayList<>(Arrays.asList(6, 7, 8, 9, 10, 11, 12));

System.out.println("List A: " + listA);

System.out.println("List B:" + listB);

List<Integer> alternatingList1 = alternate(listA, listB);

System.out.println("Alternating List 1:" + alternatingList1);

System.out.println("\n---");

List<Integer> listC = newArrayList<>(Arrays.asList(10, 20, 30));

List<Integer> listD = new ArrayList<>(Arrays.asList(1, 2));

System.out.println("List C: " + listC);

System.out.println("List D: " + listD);

List<Integer> alternatingList2 = alternate(listC, listD);

System.out.println("Alternating List 2:" + alternatingList2);

System.out.println("\n---");

List<Integer> listE = new ArrayList<>(Arrays.asList(100, 200));

List<Integer> listF = new ArrayList<>(Arrays.asList(1, 2, 3, 4));

System.out.println("List E: " + listE);

System.out.println("List F:" + listF);

List<Integer> alternatingList3 = alternate(listE, listF);

System.out.println("Alternating List 3: " + alternatingList3);

System.out.println("\n---");

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

List<Integer> listH= newArrayList<>(Arrays.asList(1, 2, 3));

System.out.println("List G: " + listG);

System.out.println("List H: " + listH);

List<Integer> alternatingList4 = alternate(listG, listH);

System.out.println("Alternating List 4:" + alternatingList4);

}

}

23. Write a method called alternate that accepts two Lists of integers as its parameters and returns a new List containing alternating elements from the two lists, in the following order: • First element from first list • First element from second list • Second element from first list • Second element from second list • Third element from first list • Third element from second list If the lists do not contain the same number of elements, the remaining elements from the longer list should be placed consecutively at the end.

importjava.util.ArrayList;

import java.util.List;

public class AlternateLists {

public static List<Integer> alternate(List<Integer> list1, List<Integer> list2){

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

int minSize = Math.min(list1.size(), list2.size());

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

result.add(list1.get(i));

result.add(list2.get(i));

}

if (list1.size() > minSize) {

result.addAll(list1.subList(minSize, list1.size()));

} else if (list2.size() > minSize) {

result.addAll(list2.subList(minSize, list2.size()));

}

return result;

}

public static void main(String[] args) {

List<Integer> list1 = List.of(1, 2, 3, 4, 5);

List<Integer> list2 = List.of(6, 7, 8, 9, 10, 11, 12);

List<Integer> result = alternate(list1, list2);

System.out.println(result);

}

}

24. Write a GUI program to develop an application that receives a string in one text field, and counts the number of vowels in the string and returns it in another text field, when the button named 'CountVowel' is clicked. When the button named 'Reset' is clicked it will reset the value of textfield one and textfield two. When the button named 'Exit' is clicked it will close the application.

import javax.swing.\*;

import java.awt.event.ActionEvent;

importjava.awt.event.ActionListener;

public class VowelCounterApp {

public static void main(String[] args){

JFrame frame = new JFrame("Vowel Counter");

frame.setSize(400, 200);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JTextField inputField = new JTextField(20);

JTextField outputField = new JTextField(20);

outputField.setEditable(false);

JButton countButton = new JButton("Count Vowel");

JButton resetButton = new JButton("Reset");

JButton exitButton = new JButton("Exit");

JPanel panel = new JPanel();

panel.add(newJLabel("Input:"));

panel.add(inputField);

panel.add(new JLabel("Vowel Count:"));

panel.add(outputField);

panel.add(countButton);

panel.add(resetButton);

panel.add(exitButton);

countButton.addActionListener(e -> {

String text = inputField.getText();

int vowelCount = 0;

for (char c :text.toLowerCase().toCharArray()){

if ("aeiou".indexOf(c) >= 0) {

vowelCount++;

}

}

outputField.setText(String.valueOf(vowelCount));

});

resetButton.addActionListener(e -> {

inputField.setText("");

outputField.setText("");

});

exitButton.addActionListener(e -> System.exit(0));

frame.add(panel);

frame.setVisible(true);

}

}

25. Building a To-Do Application using JavaFX. Create a GUI-based application where users can add tasks, mark tasks as complete, and delete tasks from the list

import javafx.application.Application;

import javafx.collections.FXCollections;

importjavafx.collections.ObservableList;

import javafx.scene.Scene;

import javafx.scene.control.\*;

importjavafx.scene.layout.VBox;

import javafx.stage.Stage;

public class ToDoApp extends Application {

private ObservableList<String> tasks;

@Override

public void start(Stage primaryStage) {

tasks = FXCollections.observableArrayList();

ListView<String> taskList = new ListView<>(tasks);

TextField taskInput = new TextField();

Button addButton = new Button("Add Task");

Button deleteButton = new Button("Delete Selected Task");

addButton.setOnAction(e -> {

String task = taskInput.getText();

if (!task.isEmpty()) {

tasks.add(task);

taskInput.clear();

}

});

deleteButton.setOnAction(e -> {

String selectedTask = taskList.getSelectionModel().getSelectedItem();

tasks.remove(selectedTask);

});

VBox layout = new VBox(10, taskInput, addButton,taskList, deleteButton);

layout.setStyle("-fx-padding: 10; -fx-spacing: 10;");

Scene scene = new Scene(layout, 300, 400);

primaryStage.setTitle("To-DoApplication");

primaryStage.setScene(scene);

primaryStage.show();

}

public static void main(String[] args){

launch(args);

}

}

26. Create a database of employees with the following fields: • Name • Code • Designation • Salary a) Write a Java program to create a GUI application that accepts employee data from TextFields and stores it in a database using JDBC. b) Write a JDBC program to retrieve all the records from the employee database.

import javax.swing.\*;

import java.awt.event.ActionEvent;

importjava.awt.event.ActionListener;

import java.sql.\*;

public class EmployeeDatabaseApp {

public static void main(String[] args){

JFrame frame = new JFrame("Employee Database");

frame.setSize(400, 300);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JTextField nameField = new JTextField(20);

JTextField codeField = new JTextField(20);

JTextField designationField = new JTextField(20);

JTextField salaryField = new JTextField(20);

JButton addButton = new JButton("Add Employee");

JButton viewButton = newJButton("View Employees");

JPanel panel = new JPanel();

panel.add(newJLabel("Name:"));

panel.add(nameField);

panel.add(new JLabel("Code:"));

panel.add(codeField);

panel.add(newJLabel("Designation:"));

panel.add(designationField);

panel.add(new JLabel("Salary:"));

panel.add(salaryField);

panel.add(addButton);

panel.add(viewButton);

addButton.addActionListener(e -> {

try (Connection conn =

DriverManager.getConnection("jdbc:mysql://localhost:3306/employee\_db","root",

"password")) {

String query = "INSERT INTO employees (name, code, designation, salary) VALUES

(?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, nameField.getText());

stmt.setString(2, codeField.getText());

stmt.setString(3, designationField.getText());

stmt.setDouble(4,Double.parseDouble(salaryField.getText()));

stmt.executeUpdate();

JOptionPane.showMessageDialog(frame, "Employee added successfully.");

} catch (Exception ex){

JOptionPane.showMessageDialog(frame, "Error: " + ex.getMessage());

}

});

viewButton.addActionListener(e -> {

try (Connection conn =

DriverManager.getConnection("jdbc:mysql://localhost:3306/employee\_db","root",

"password")) {

String query = "SELECT \* FROM employees";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query);

StringBuilder results = new StringBuilder("Employees:

");

");

while (rs.next()){

results.append("Name: ").append(rs.getString("name"))

.append(", Code:").append(rs.getString("code"))

.append(", Designation: ").append(rs.getString("designation"))

.append(", Salary: ").append(rs.getDouble("salary"))

.append("

}

JOptionPane.showMessageDialog(frame, results.toString());

} catch (Exception ex){

JOptionPane.showMessageDialog(frame, "Error: " + ex.getMessage());

}

});

frame.add(panel);

frame.setVisible(true);

}

}