PRAGRAM 1:

package program1;

import java.util.Scanner;

interface Personal {

void personalDetails();

}

interface Official {

void officialDetails();

}

abstract class Person implements Personal, Official {

protected String name;

protected String address;

protected double phone;

public Person(String name, String address, double phone) {

this.name = name;

this.address = address;

this.phone = phone;

}

}

class Employee extends Person {

private int empId;

private String company;

private String designation;

public Employee(String name, String address, double phone, int empId, String company, String designation) {

super(name, address, phone);

this.empId = empId;

this.company = company;

this.designation = designation;

}

*@Override*

public void personalDetails() {

System.***out***.println("--------------Employee details----------------");

System.***out***.println("Employee name: " + name);

System.***out***.println("Address: " + address);

System.***out***.println("Phone number: " + phone);

}

*@Override*

public void officialDetails() {

System.***out***.println("Staff ID: " + empId);

System.***out***.println("Company: " + company);

System.***out***.println("Designation: " + designation);

}

}

public class lab1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.***in***);

System.***out***.println("Employee Details");

System.***out***.print("Name: ");

String empName = sc.nextLine();

System.***out***.print("Address: ");

String empAddr = sc.nextLine();

System.***out***.print("Phone Number: ");

double empNum = sc.nextDouble();

sc.nextLine(); // Consume the newline

System.***out***.print("ID: ");

int empId = sc.nextInt();

sc.nextLine(); // Consume the newline

System.***out***.print("Company: ");

String empCom = sc.nextLine();

System.***out***.print("Designation: ");

String empDes = sc.nextLine();

Employee emp = new Employee(empName, empAddr, empNum, empId, empCom, empDes);

emp.personalDetails();

emp.officialDetails();

sc.close();

}

}

PROGAM 2:

SQUARE

package program2;

public class Square { // Conventionally, class names should start with an uppercase letter

private double side;

public Square(double side) {

this.side = side;

}

public double getArea() {

return side \* side;

}

public double getPerimeter() {

return 4 \* side;

}

}

TRIANGLE

package program2;

public class Triangle {

private double base;

private double height;

public Triangle(double base, double height) {

this.base = base;

this.height = height;

}

public double getArea() {

return 0.5 \* base \* height;

}

public double getPerimeter() {

// Assuming this is an equilateral triangle with all sides equal to 'base'

return 3 \* base;

}

}

CIRCLE

package program2;

public class Circle {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

public double getArea() {

return Math.***PI*** \* radius \* radius;

}

public double getCircumference() {

return 2 \* Math.***PI*** \* radius;

}

}

MAIN

package program2;

import java.util.Scanner;

import program2.Square;

import program2.Triangle;

import program2.Circle;

public class lab2 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

int choice;

do {

System.***out***.println("Menu:");

System.***out***.println("1. Calculate Square");

System.***out***.println("2. Calculate Triangle");

System.***out***.println("3. Calculate Circle");

System.***out***.println("0. Exit");

System.***out***.print("Enter your choice: ");

choice = scanner.nextInt();

switch (choice) {

case 1:

System.***out***.print("Enter side length of the square: ");

double side = scanner.nextDouble();

Square square = new Square(side);

System.***out***.println("Square area: " + square.getArea());

System.***out***.println("Square perimeter: " + square.getPerimeter());

break;

case 2:

System.***out***.print("Enter base length of the triangle: ");

double base = scanner.nextDouble();

System.***out***.print("Enter height of the triangle: ");

double height = scanner.nextDouble();

Triangle triangle = new Triangle(base, height);

System.***out***.println("Triangle area: " + triangle.getArea());

System.***out***.println("Triangle perimeter: " + triangle.getPerimeter());

break;

case 3:

System.***out***.print("Enter radius of the circle: ");

double radius = scanner.nextDouble();

Circle circle = new Circle(radius);

System.***out***.println("Circle area: " + circle.getArea());

System.***out***.println("Circle circumference: " + circle.getCircumference());

break;

case 0:

System.***out***.println("Exiting...");

break;

default:

System.***out***.println("Invalid choice. Please enter a valid option.");

break;

}

System.***out***.println();

} while (choice != 0);

scanner.close();

}

}

PROGRAM 3.1:

package program3;

import java.util.Scanner;

public class lab3 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

Printer p = new LaserPrinter("Laser Printer", 23);

Printer ps = new Dotmatrix("Dot Matrix Printer", 334);

while (true) {

System.***out***.println("Select an option:");

System.***out***.println("1. Print Laser Printer");

System.***out***.println("2. Print Dot Matrix Printer");

System.***out***.println("3. Exit");

System.***out***.println("Enter your choice");

int choice = scanner.nextInt();

switch (choice) {

case 1:

p.addPrinter();

break;

case 2:

ps.addPrinter();

break;

case 3:

System.***out***.println("Exiting...");

scanner.close();

return;

default:

System.***out***.println("Invalid choice.");

}}

}

}

abstract class Printer {

String name;

int model;

public Printer(String name, int model) {

this.name = name;

this.model = model;

}

abstract void addPrinter();

}

class LaserPrinter extends Printer {

public LaserPrinter(String name, int age) {

super(name, age);

}

void addPrinter() {

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

System.***out***.println("model: " + this.model);

}

}

class Dotmatrix extends Printer {

public Dotmatrix(String name, int age) {

super(name, age);

}

void addPrinter() {

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

System.***out***.println("model: " + this.model);

}

}

PRAGRAM 3.2:

package program3;

import java.util.Scanner;

abstract class Vehical {

public abstract void drive();

}

interface Drivable {

void drive();

}

class Car extends Vehical implements Drivable {

public void drive() {

System.***out***.println("Car is driven");

}

}

class Moterbike extends Vehical implements Drivable {

public void drive() {

System.***out***.println("Moterbike is driven");

}

}

public class lab33 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.***in***);

Vehical v1 = new Car();

Vehical v2 = new Moterbike();

Drivable d1 = new Car();

Drivable d2 = new Moterbike();

while (true) {

System.***out***.println("Choose a vehical to drive:");

System.***out***.println("1. Car");

System.***out***.println("2. Moterbike");

System.***out***.println("3. Exit");

System.***out***.print("Enter your choice: ");

int choice = sc.nextInt();

switch (choice) {

case 1:

v1.drive();

break;

case 2:

v2.drive();

break;

case 3:

System.***out***.println("Exiting program...");

sc.close();

return;

default:

System.***out***.println("Invalid choice");

}

}

}

}

PROGRAM 4:

package program4;

import java.util.Scanner;

interface MediaPlayer {

void play(String audioType, String fileName);

}

interface AdvancedMediaPlayer {

void playVlc(String fileName);

void playMp4(String fileName);

}

class MediaAdapter implements MediaPlayer {

AdvancedMediaPlayer advancedMusicPlayer;

public MediaAdapter(String audioType) {

if (audioType.equalsIgnoreCase("vlc")) {

advancedMusicPlayer = new VlcPlayer();

} else if (audioType.equalsIgnoreCase("mp4")) {

advancedMusicPlayer = new Mp4Player();

}

}

*@Override*

public void play(String audioType, String fileName) {

if (audioType.equalsIgnoreCase("vlc")) {

advancedMusicPlayer.playVlc(fileName);

} else if (audioType.equalsIgnoreCase("mp4")) {

advancedMusicPlayer.playMp4(fileName);

}

}

}

class VlcPlayer implements AdvancedMediaPlayer {

*@Override*

public void playVlc(String fileName) {

System.***out***.println("Playing vlc file. Name: " + fileName);

}

*@Override*

public void playMp4(String fileName) {

// Do nothing

}

}

class Mp4Player implements AdvancedMediaPlayer {

*@Override*

public void playVlc(String fileName) {

// Do nothing

}

*@Override*

public void playMp4(String fileName) {

System.***out***.println("Playing mp4 file. Name: " + fileName);

}

}

class AudioPlayer implements MediaPlayer {

MediaAdapter mediaAdapter;

*@Override*

public void play(String audioType, String fileName) {

if (audioType.equalsIgnoreCase("mp3")) {

System.***out***.println("Playing mp3 file. Name: " + fileName);

} else if (audioType.equalsIgnoreCase("vlc") || audioType.equalsIgnoreCase("mp4")) {

mediaAdapter = new MediaAdapter(audioType);

mediaAdapter.play(audioType, fileName);

} else {

System.***out***.println("Invalid media. " + audioType + " format not supported");

}

}

}

public class lab4 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

MediaPlayer audioPlayer = new AudioPlayer();

while (true) {

System.***out***.println("Media Player Menu:");

System.***out***.println("1. Play MP3");

System.***out***.println("2. Play MP4");

System.***out***.println("3. Play VLC");

System.***out***.println("4. Exit");

System.***out***.print("Choose an option: ");

int choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

if (choice == 4) {

System.***out***.println("Exiting Media Player. Goodbye!");

break;

}

System.***out***.print("Enter the file name: ");

String fileName = scanner.nextLine();

switch (choice) {

case 1:

audioPlayer.play("mp3", fileName);

break;

case 2:

audioPlayer.play("mp4", fileName);

break;

case 3:

audioPlayer.play("vlc", fileName);

break;

default:

System.***out***.println("Invalid choice. Please choose a valid option.");

}

}

scanner.close();

}

}

PROGRAM 5:

package program5;

class Thread1 extends Thread {

public void run() {

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

// Check if the thread has been interrupted

if (Thread.*currentThread*().isInterrupted()) {

System.***out***.println("Thread 1 interrupted");

return; // Exit the loop and end the thread

}

System.***out***.println("Thread 1: " + i);

try {

Thread.*sleep*(2000);

} catch (InterruptedException e) {

System.***out***.println("Thread 1 interrupted during sleep");

return; // Exit the thread

}

}

}

}

class Thread2 extends Thread {

public void run() {

for (int i = 10; i <= 20; i++) {

// Check if the thread has been interrupted

if (Thread.*currentThread*().isInterrupted()) {

System.***out***.println("Thread 2 interrupted");

return; // Exit the loop and end the thread

}

System.***out***.println("Thread 2: " + i);

try {

Thread.*sleep*(2000);

} catch (InterruptedException e) {

System.***out***.println("Thread 2 interrupted during sleep");

return; // Exit the thread

}

}

}

}

public class lab5 {

public static void main(String[] args) throws Exception {

Thread1 t1 = new Thread1();

Thread2 t2 = new Thread2();

t1.start(); // Start Thread1

t1.interrupt(); // Interrupt Thread1

t2.start(); // Start Thread2

// Interrupt only Thread1

//Thread.sleep(1000); // Adjusted delay to give Thread1 some time to start

//t1.interrupt(); // Interrupt Thread1

// Wait for threads to finish if needed

t1.join();

t2.join();

}

}

PROGRAM 6:

package program6;

public class lab6 {

private int material;

private boolean available = false;

// Method to get the material

public synchronized int get() {

while (!available) {

try {

wait(); // Wait until material is available

} catch (InterruptedException e) {

Thread.*currentThread*().interrupt(); // Handle interrupt

}

}

available = false;

notifyAll(); // Notify all waiting threads

return material;

}

// Method to put the material

public synchronized void put(int value) {

while (available) {

try {

wait(); // Wait if material is already available

} catch (InterruptedException e) {

Thread.*currentThread*().interrupt(); // Handle interrupt

}

}

material = value;

available = true;

notifyAll(); // Notify all waiting threads

}

public static class Producer extends Thread {

private lab6 shop; // Changed to lab6 to match the class name

private int number;

public Producer(lab6 shop, int number) {

this.shop = shop;

this.number = number;

}

*@Override*

public void run() {

for (int i = 0; i < 10; i++) { // Produce 10 items

shop.put(i); // Put item into the shop

System.***out***.println("Producer " + this.number + " put: " + i);

try {

*sleep*((int) (Math.*random*() \* 100)); // Random sleep

} catch (InterruptedException e) {

Thread.*currentThread*().interrupt(); // Handle interrupt

}

}

}

}

public static class Consumer extends Thread {

private lab6 shop; // Changed to lab6 to match the class name

private int number;

public Consumer(lab6 shop, int number) {

this.shop = shop;

this.number = number;

}

*@Override*

public void run() {

for (int i = 0; i < 10; i++) { // Consume 10 items

int value = shop.get(); // Get item from the shop

System.***out***.println("Consumer " + this.number + " got: " + value);

try {

*sleep*((int) (Math.*random*() \* 100)); // Random sleep

} catch (InterruptedException e) {

Thread.*currentThread*().interrupt(); // Handle interrupt

}

}

}

}

public static void main(String[] args) {

lab6 shop = new lab6(); // Instantiate lab6 object

Producer producer = new Producer(shop, 1); // Create a Producer

Consumer consumer = new Consumer(shop, 1); // Create a Consumer

producer.start(); // Start the Producer thread

consumer.start(); // Start the Consumer thread

}

}

PROGRAM 7:

package program7;

class IllegalArgumentException extends Exception{

public IllegalArgumentException(String msg) {

super(msg);

System.***out***.println(msg);

}

}

class SingletonBank{

private static SingletonBank *instance* = new SingletonBank();

private double balance;

private SingletonBank() {

this.balance = 0.0;

}

public static synchronized SingletonBank getInstance() {

return *instance*;

}

public synchronized void credit(double amount) throws IllegalArgumentException {

if (amount < 0) {

throw new IllegalArgumentException("Amount must be positive");

}

balance += amount;

System.***out***.println("Credited " + amount + ", new balance: " + balance);

}

public synchronized void debit(double amount) throws IllegalArgumentException {

if (amount < 0) {

throw new IllegalArgumentException("Amount must be positive");

}

if (balance < amount) {

throw new IllegalArgumentException("Insufficient funds");

}

balance -= amount;

System.***out***.println("Debited " + amount + ", new balance: " + balance);

}

public synchronized double getBalance() {

return this.balance;

}

}

class CreditThread extends Thread {

private final SingletonBank account;

private final double amount;

CreditThread(SingletonBank account, double amount) {

this.account = account;

this.amount = amount;

}

*@Override*

public void run() {

try {

account.credit(amount);

} catch (IllegalArgumentException e) {

}

}

}

class DebitThread extends Thread {

private final SingletonBank account;

private final double amount;

DebitThread(SingletonBank account, double amount) {

this.account = account;

this.amount = amount;

}

*@Override*

public void run() {

try {

account.debit(amount);

} catch (IllegalArgumentException e) {

// **TODO** Auto-generated catch block

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

}

}

}

public class lab7 {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

SingletonBank account = SingletonBank.*getInstance*();

Thread t1 = new CreditThread(account,1000);

Thread t2 = new DebitThread(account,500);

Thread t3 = new CreditThread(account,1000);

Thread t4 = new DebitThread(account,500);

t1.start();

t2.start();

t3.start();

t4.start();

}

}

PROGRAM 8.1:

package program8;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

// Step 1: Create a class that accepts integer and double values from user and converts it into a List format

class UpperBoundWildcard {

private List<Number> numbers;

public UpperBoundWildcard() {

numbers = new ArrayList<>();

}

public void addNumber(Number number) {

numbers.add(number);

}

public List<Number> getNumbers() {

return numbers;

}

}

// Step 2: Create sum and sort methods that accept the generic type list

class MathOperations {

public static double sum(List<? extends Number> numbers) {

double sum = 0;

for (Number number : numbers) {

sum += number.doubleValue();

}

return sum;

}

public static void sort(List<? extends Number> numbers) {

Collections.*sort*(numbers, (a, b) -> Double.*compare*(a.doubleValue(), b.doubleValue()));

}

}

// Step 3: Design a menu-driven program to accept and call for the sum and sort of numbers

public class lab8 {

public static void main(String[] args) {

UpperBoundWildcard wildcard = new UpperBoundWildcard();

Scanner scanner = new Scanner(System.***in***);

while (true) {

System.***out***.println("Menu:");

System.***out***.println("1. Add Integer");

System.***out***.println("2. Add Double");

System.***out***.println("3. Sum");

System.***out***.println("4. Sort");

System.***out***.println("5. Exit");

System.***out***.println("Enter your choice");

int choice = scanner.nextInt();

switch (choice) {

case 1:

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

wildcard.addNumber(scanner.nextInt());

break;

case 2:

System.***out***.println("Enter a double:");

wildcard.addNumber(scanner.nextDouble());

break;

case 3:

System.***out***.println("Sum: " + MathOperations.*sum*(wildcard.getNumbers()));

break;

case 4:

MathOperations.*sort*(wildcard.getNumbers());

System.***out***.println("Sorted numbers: " + wildcard.getNumbers());

break;

case 5:

System.*exit*(0);

break;

default:

System.***out***.println("Invalid choice. Please try again.");

}

}

}

}

PROGRAM 8.2:

package program8;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

class UpperBoundWildcards {

private List<Number> numbers;

private List<String> strings;

public UpperBoundWildcards() {

numbers = new ArrayList<>();

strings = new ArrayList<>();

}

public void addNumber(Number number) {

numbers.add(number);

}

public void addString(String string) {

strings.add(string);

}

public List<Number> getNumbers() {

return numbers;

}

public List<String> getStrings() {

return strings;

}

}

class MathOperationss {

public static double sum(List<? extends Number> numbers) {

double sum = 0;

for (Number number : numbers) {

sum += number.doubleValue();

}

return sum;

}

public static void sort(List<? extends Number> numbers) {

Collections.*sort*(numbers, (a, b) -> Double.*compare*(a.doubleValue(), b.doubleValue()));

}

public static void sortStrings(List<String> strings) {

Collections.*sort*(strings);

}

}

public class lab88 {

public static void main(String[] args) {

UpperBoundWildcards wildcard = new UpperBoundWildcards();

Scanner scanner = new Scanner(System.***in***);

while (true) {

System.***out***.println("Menu:");

System.***out***.println("1. Add Integer");

System.***out***.println("2. Add Double");

System.***out***.println("3. Add Float");

System.***out***.println("4. Add String");

System.***out***.println("5. Sum of Numbers");

System.***out***.println("6. Sort Numbers");

System.***out***.println("7. Sort Strings");

System.***out***.println("8. Exit");

System.***out***.println("Enter your choice");

int choice;

try {

choice = Integer.*parseInt*(scanner.nextLine());

} catch (NumberFormatException e) {

System.***out***.println("Invalid choice. Please enter a number.");

continue;

}

switch (choice) {

case 1:

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

try {

wildcard.addNumber(scanner.nextInt());

scanner.nextLine(); // Consume newline

} catch (Exception e) {

System.***out***.println("Invalid input. Please enter an integer.");

scanner.nextLine(); // Clear invalid input

}

break;

case 2:

System.***out***.println("Enter a double:");

try {

wildcard.addNumber(scanner.nextDouble());

scanner.nextLine(); // Consume newline

} catch (Exception e) {

System.***out***.println("Invalid input. Please enter a double.");

scanner.nextLine(); // Clear invalid input

}

break;

case 3:

System.***out***.println("Enter a float:");

try {

wildcard.addNumber(scanner.nextFloat());

scanner.nextLine(); // Consume newline

} catch (Exception e) {

System.***out***.println("Invalid input. Please enter a float.");

scanner.nextLine(); // Clear invalid input

}

break;

case 4:

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

wildcard.addString(scanner.nextLine());

break;

case 5:

System.***out***.println("Sum: " + MathOperationss.*sum*(wildcard.getNumbers()));

break;

case 6:

MathOperationss.*sort*(wildcard.getNumbers());

System.***out***.println("Sorted numbers: " + wildcard.getNumbers());

break;

case 7:

MathOperationss.*sortStrings*(wildcard.getStrings());

System.***out***.println("Sorted strings: " + wildcard.getStrings());

break;

case 8:

System.***out***.println("Exiting...");

scanner.close();

System.*exit*(0);

break;

default:

System.***out***.println("Invalid choice. Please try again.");

}

}

}

}