2. Write a program in Java to demonstrate sleep() and wait() ?

Source code :

package smpili;

public class sleep {

public static void main(String[] args) {

Object lock = new Object();

// Thread 1

Thread t1 = new Thread(() -> {

synchronized (lock) {

System.***out***.println("Thread 1 acquired lock and is waiting for 5 seconds...");

try {

Thread.*sleep*(5000); // Sleep for 5 seconds

} catch (InterruptedException e) {

e.printStackTrace();

}

System.***out***.println("Thread 1 finished waiting.");

// lock.notify(); // Notify the waiting thread

}

});

// Thread 2

Thread t2 = new Thread(() -> {

synchronized (lock) {

System.***out***.println("Thread 2 acquired lock and is waiting...");

/\* try {

lock.wait(); // Wait for notification

} catch (InterruptedException e) {

e.printStackTrace();

}\*/

System.***out***.println("Thread 2 finished waiting.");

}

});

// Start the threads

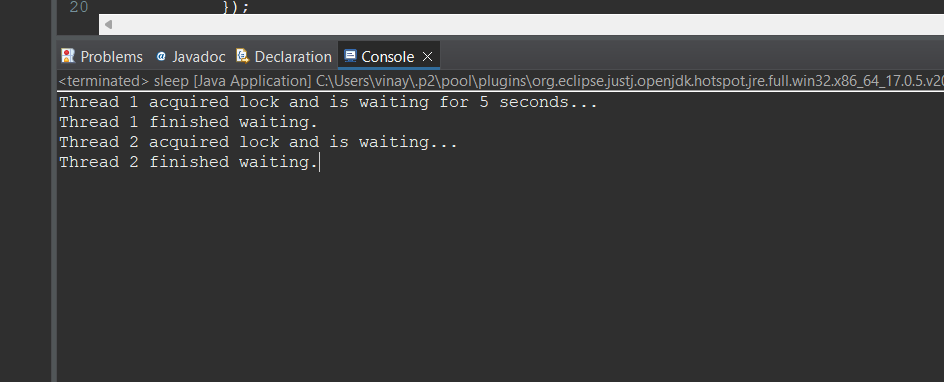
t1.start();

t2.start();

}

}

Output:



3.Write a program in Java to demonstrate synchronization ?

Source code:

package smpili;

public class synronization1 {

private static int *counter* = 0;

public synchronized static void incrementCounter() {

*counter*++; // Increment the counter in a synchronized method

}

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

Thread t1 = new Thread(() -> {

for (int i = 0; i < 100000; i++) {

*incrementCounter*();

}

});

Thread t2 = new Thread(() -> {

for (int i = 0; i < 100000; i++) {

*incrementCounter*();

}

});

t1.start();

t2.start();

t1.join();

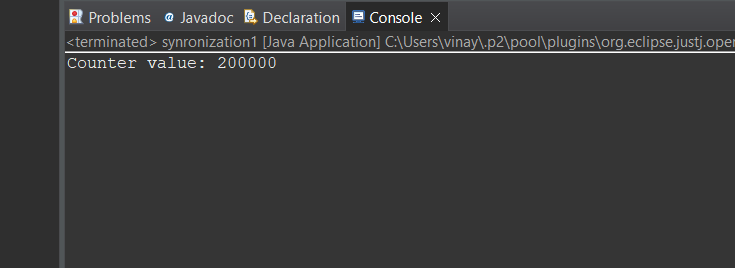
t2.join();

System.***out***.println("Counter value: " + *counter*);

}

}

OUTPUT:

**

4.Write a program in Java to demonstrate try and catch ?

Source code:

package smpili;

import java.util.Scanner;

public class trycatch4 {

//public class trycatch {

//import java.util.Scanner;

// public class TryCatchDemo {

public static void main(String[] args) {

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

int a, b;

try {

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

a = scanner.nextInt();

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

b = scanner.nextInt();

int result = a / b;

System.***out***.println("Result: " + result);

} catch (ArithmeticException e) {

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

} catch (Exception e) {

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

} finally {

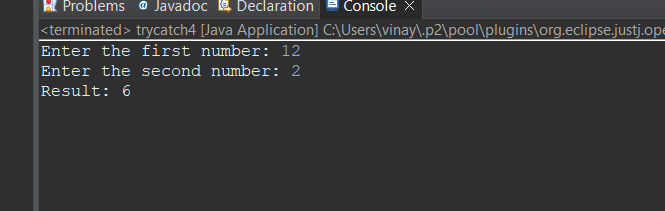
scanner.close();

}

}

}

Output:

**

*5.* Writing a program in Java to throws, throw, finally, and custom exceptions in Java?

*Source code:*

package smpili;

public class finally5 {

import java.util.Scanner;

class CustomException extends Exception {

CustomException(String message) {

super(message);

}

//public class ExceptionDemo {

public static void main(String[] args) {

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

try {

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

int a = scanner.nextInt();

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

int b = scanner.nextInt();

if (b == 0) {

throw new CustomException("Second number cannot be zero");

}

int result = a / b;

System.***out***.println("Result: " + result);

} catch (CustomException e) {

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

} catch (ArithmeticException e) {

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

} catch (Exception e) {

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

} finally {

System.***out***.println("Finally block is always executed");

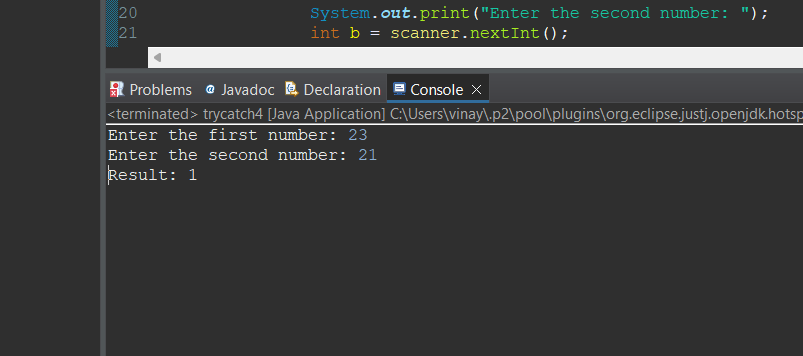
scanner.close();

}

}

}

*OUTPUT:*

**

6.Write a program in Java to demonstrate exception handling?

Source Code:

import java.util.Scanner;

public class ExceptionHandling {

//import java.util.Scanner;

/\*public class ExceptionHandlingDemo {\*/

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

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

int a = scanner.nextInt();

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

int b = scanner.nextInt();

int result = a / b;

System.out.println("Result: " + result);

} catch (ArithmeticException e) {

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

} catch (Exception e) {

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

} finally {

System.out.println("Finally block is always executed");

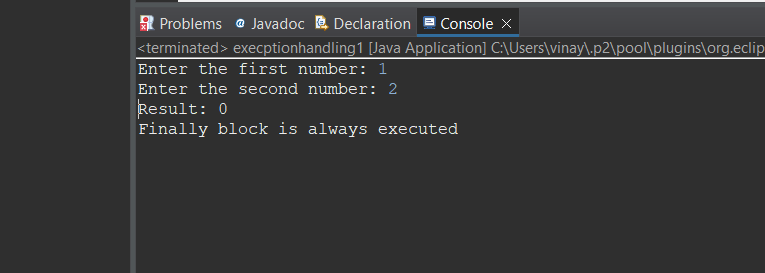
scanner.close();

}

}

}

OUTPUT:

**

*7.* Writing a program in Java to create, read, update, and delete operations on the files in Java?

Source Code:

package smpili;

import java.io.File;

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.Scanner;

public class updatedelete1 {

public static void main(String[] args) {

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

while (true) {

System.***out***.println("1. Create file");

System.***out***.println("2. Read file");

System.***out***.println("3. Update file");

System.***out***.println("4. Delete file");

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

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

int choice = scanner.nextInt();

switch (choice) {

case 1:

*createFile*();

break;

case 2:

*readFile*();

break;

case 3:

*updateFile*();

break;

case 4:

*deleteFile*();

break;

case 5:

scanner.close();

System.*exit*(0);

break;

default:

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

}

}

}

public static void createFile() {

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

try {

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

String fileName = scanner.nextLine();

File file = new File(fileName);

if (file.createNewFile()) {

System.***out***.println("File created successfully");

} else {

System.***out***.println("File already exists");

}

} catch (IOException e) {

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

}

}

public static void readFile() {

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

try {

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

String fileName = scanner.nextLine();

Path filePath = Paths.*get*(fileName);

String fileContent = new String(Files.*readAllBytes*(filePath));

System.***out***.println("File content: " + fileContent);

} catch (IOException e) {

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

}

}

public static void updateFile() {

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

try {

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

String fileName = scanner.nextLine();

Path filePath = Paths.*get*(fileName);

if (Files.*exists*(filePath)) {

System.***out***.print("Enter the new content: ");

String newContent = scanner.nextLine();

Files.*write*(filePath, newContent.getBytes());

System.***out***.println("File updated successfully");

} else {

System.***out***.println("File does not exist");

}

} catch (IOException e) {

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

}

}

public static void deleteFile() {

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

try {

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

String fileName = scanner.nextLine();

Path filePath = Paths.*get*(fileName);

if (Files.*exists*(filePath)) {

Files.*delete*(filePath);

System.***out***.println("File deleted successfully");

} else {

System.***out***.println("File does not exist");

}

} catch (IOException e) {

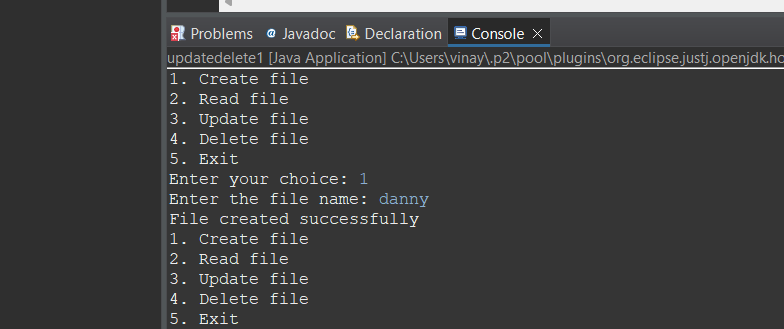
System.***out***.println("Error: " + e.getMessage());

}

}

}

Ouput;

**

*8.* Write a program in Java to demonstrate the uses of classes, objects, and the object-oriented..?

Source code:

package smpili;

public class car8 {

/\*public class car8 {\*/

private String make;

private String model;

private int year;

public car8(String make, String model, int year) {

this.make = make;

this.model = model;

this.year = year;

}

public String getMake() {

return make;

}

public String getModel() {

return model;

}

public int getYear() {

return year;

}

public void startEngine() {

System.***out***.println("Starting the engine...");

}

public void stopEngine() {

System.***out***.println("Stopping the engine...");

}

*@SuppressWarnings*("null")

public static void main(String[] args) {

car8 myCar1 = new car8("Honda", "Civic", 2021);

//car8 myCar11 = null;

System.***out***.println("Make: " + myCar1.getMake());

System.***out***.println("Model: " + myCar1.getModel());

System.***out***.println("Year: " + myCar1.getYear());

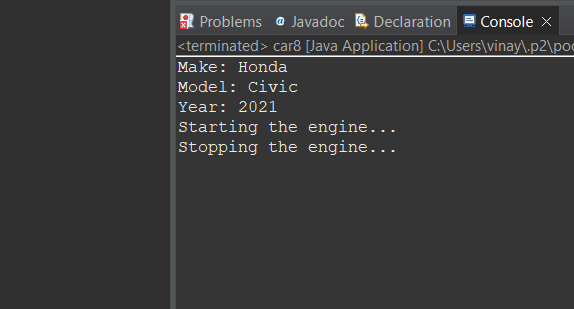
myCar1.startEngine();

myCar1.stopEngine();

}

}

*Output:*

**

*9.* Write a program in Java to resolve the diamond problem using OOPs’ concepts.?

*Source code:*

package smpili;

//public class diamond9 {

// Define an interface A with a default method

interface A {

default void print() {

System.***out***.println("Interface A");

}

}

// Define an interface B that extends A with its own default method

interface B extends A {

default void print() {

System.***out***.println("Interface B");

}

}

// Define an interface C that extends A with its own default method

interface C extends A {

default void print() {

System.***out***.println("Interface C");

}

}

// Define a class D that implements B and C

class D implements B, C {

// Override the conflicting default method from interfaces B and C

*@Override*

public void print() {

System.***out***.println("Class D");

}

}

// Define a main class to test the program

public class diamond9 {

public static void main(String[] args) {

D d = new D();

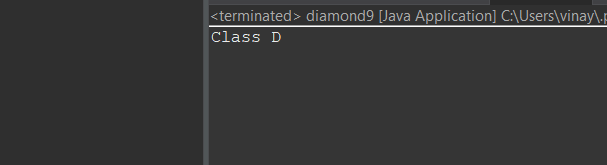
d.print(); // Output: Class D

}

}

*Outout:*

we create an object of class **D** and call its **print** method. The output will be **Class D**, which shows that the **print** method from class **D** was called, resolving the diamond problem.

**