Lab7

Aditya kamble

(1)-Write the programme to open a text file named input 2, and copy its contents to an output text file output 2.

**package** lab7;

**import** java.io.\*;

**public** **class** que1 {

**public** **static** **void** main(String[] args) {

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

String inputFile = "input2.txt";

String outputFile = "output2.txt";

FileReader reader = **null**;

FileWriter writer = **null**;

**try** {

reader = **new** FileReader(inputFile);

writer = **new** FileWriter(outputFile);

**int** character;

**while** ((character = reader.read()) != -1) {

writer.write(character);

}

System.***out***.println("Contents of " + inputFile + " copied to " + outputFile + " successfully.");

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (reader != **null**) {

reader.close();

}

**if** (writer != **null**) {

writer.close();

}

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

}

-

(file is not being read)

(2)- Write the programme to show multithreading for the string “multi threads”. Show the resulting output.

-

**package** lab7;

**public** **class** que2 {

**public** **static** **void** main(String[] args) {

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

String inputString = "multi threads";

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

Thread thread = **new** Thread(**new** PrintCharTask(inputString.charAt(i)));

thread.start(); // Start the thread

}

}

**private** **static** **class** PrintCharTask **implements** Runnable {

**private** **char** character;

**public** PrintCharTask(**char** character) {

**this**.character = character;

}

**public** **void** run() {

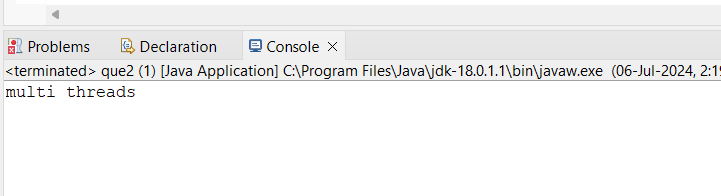
System.***out***.print(character);

}

}

}

Output-



(3)- Implement a Java program that creates a thread using the Runnable interface. The thread should print numbers from 1 to 10 with a delay of 1 second between each number.

-

**package** lab7;

**class** NumberPrinter **implements** Runnable {

**public** **void** run() {

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

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

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**public** **class** que3 {

**public** **static** **void** main(String[] args) {

Runnable numberPrinter = **new** NumberPrinter();

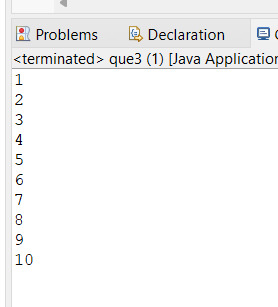
Thread thread = **new** Thread(numberPrinter);

thread.start();

}

}

Output-



(4)-Write a Java program that creates and starts three threads. Each thread should print its name and count from 1 to 5 with a delay of 500 milliseconds between each count.

-

**package** lab7;

**public** **class** que4 {

**public** **static** **void** main(String[] args) {

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

**class** CountingThread **extends** Thread {

**private** String threadName;

**public** CountingThread(String threadName) {

**this**.threadName = threadName;

}

**public** **void** run() {

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

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

**try** {

Thread.*sleep*(500);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**public** **class** Main {

**public** **static** **void** main(String[] args) {

thread1.start();

thread2.start();

thread3.start();

}

}

}

}

- Thread 1: 1

Thread 2: 1

Thread 3: 1

Thread 1: 2

Thread 2: 2

Thread 3: 2

Thread 1: 3

Thread 2: 3

Thread 3: 3

Thread 1: 4

Thread 2: 4

Thread 3: 4

Thread 1: 5

Thread 2: 5

Thread 3: 5

(5)- Create a Java program that demonstrates thread priorities. Create three threads with different priorities and observe the order in which they execute.

-

**package** lab7;

**class** PriorityThread **extends** Thread {

**public** PriorityThread(String name, **int** priority) {

**super**(name);

setPriority(priority);

}

**public** **void** run() {

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

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

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}**public** **class** Main {

**public** **static** **void** main(String[] args) {

PriorityThread thread1 = **new** PriorityThread("Thread 1", Thread.***MIN\_PRIORITY***);

PriorityThread thread2 = **new** PriorityThread("Thread 2", Thread.***NORM\_PRIORITY***);

PriorityThread thread3 = **new** PriorityThread("Thread 3", Thread.***MAX\_PRIORITY***);

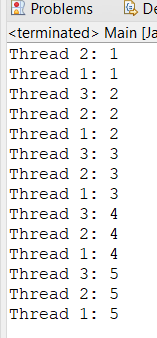
thread1.start();

thread2.start();

thread3.start();

}

}

Output- 

(6)-Write a Java program that creates a deadlock scenario with two threads and two resources.

-

**package** lab7;

**public** **class** que6 {

**private** **static** **final** Object ***resource1*** = **new** Object();

**private** **static** **final** Object ***resource2*** = **new** Object();

**public** **static** **void** main(String[] args) {

Thread thread1 = **new** Thread(() -> {

**synchronized** (***resource1***) {

System.***out***.println("Thread 1: Locked resource1");

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

**synchronized** (***resource2***) {

System.***out***.println("Thread 1: Locked resource2");

}

}

});

Thread thread2 = **new** Thread(() -> {

**synchronized** (***resource2***) {

System.***out***.println("Thread 2: Locked resource2");

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

**synchronized** (***resource1***) {

System.***out***.println("Thread 2: Locked resource1");

}

}

});

thread1.start();

thread2.start();

}

}

Output-

