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.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** CopyFile {

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

FileReader fr = **null**;

FileWriter fw = **null**;

**try** {

// Open the input file

fr = **new** FileReader("input2.txt");

// Open the output file

fw = **new** FileWriter("output2.txt");

**int** c;

// Read and write one character at a time

**while** ((c = fr.read()) != -1) {

fw.write(c);

}

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

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

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

fr.close();

}

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

fw.close();

}

} **catch** (IOException ex) {

ex.printStackTrace();

}

}

}

}

OutPut:

File copied successfully.

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

**package** lab7;

**public** **class** MultiThreadExample {

// The string to be printed

**static** String *inputString* = "multi threads";

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

// Create and start a thread for each character in the string

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

**int** index = i; // To use in the lambda expression

Thread thread = **new** Thread(() -> *printChar*(index));

thread.start();

}

}

// Method to print the character at the given index

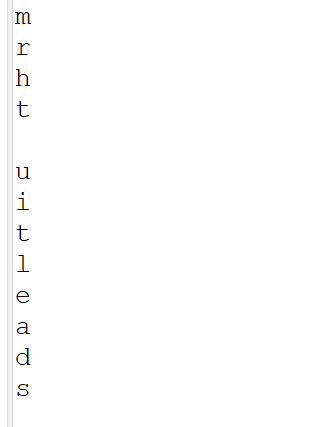
**public** **static** **void** printChar(**int** index) {

System.***out***.println(*inputString*.charAt(index));

}

}

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;

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

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

// Create an instance of the NumberPrinter class

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

// Create a new thread with the Runnable instance

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

// Start the thread

thread.start();

}

@Override

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

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

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

**try** {

// Sleep for 1 second

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// Handle the interrupted exception

e.printStackTrace();

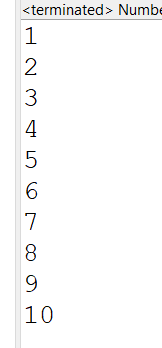
}

}

}

}

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;

**class** CountPrinter **implements** Runnable {

**private** String threadName;

**public** CountPrinter(String threadName) {

**this**.threadName = threadName;

}

@Override

**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** MultiThreadCount {

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

// Create instances of the CountPrinter class

CountPrinter printer1 = **new** CountPrinter("Thread 1");

CountPrinter printer2 = **new** CountPrinter("Thread 2");

CountPrinter printer3 = **new** CountPrinter("Thread 3");

// Create threads with the CountPrinter instances

Thread thread1 = **new** Thread(printer1);

Thread thread2 = **new** Thread(printer2);

Thread thread3 = **new** Thread(printer3);

// Start the threads

thread1.start();

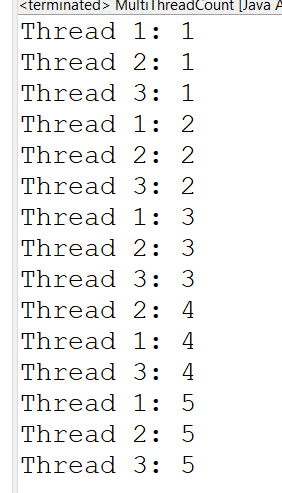
thread2.start();

thread3.start();

}

}

OutPut:



**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 **implements** Runnable {

**private** String threadName;

**public** PriorityThread(String threadName) {

**this**.threadName = threadName;

}

@Override

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

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

System.***out***.println(threadName + " - Priority: " + Thread.*currentThread*().getPriority() + " - Count: " + i);

**try** {

Thread.*sleep*(500);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**public** **class** ThreadPriorityDemo {

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

// Create instances of the PriorityThread class

PriorityThread pt1 = **new** PriorityThread("Thread 1");

PriorityThread pt2 = **new** PriorityThread("Thread 2");

PriorityThread pt3 = **new** PriorityThread("Thread 3");

// Create threads with the PriorityThread instances

Thread thread1 = **new** Thread(pt1);

Thread thread2 = **new** Thread(pt2);

Thread thread3 = **new** Thread(pt3);

// Set different priorities for each thread

thread1.setPriority(Thread.***MIN\_PRIORITY***); // Priority 1

thread2.setPriority(Thread.***NORM\_PRIORITY***); // Priority 5

thread3.setPriority(Thread.***MAX\_PRIORITY***); // Priority 10

// Start the threads

thread1.start();

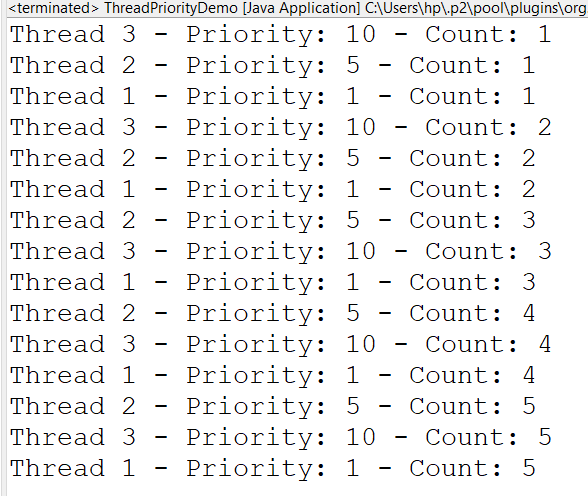
thread2.start();

thread3.start();

}

}

OutPut:



**6.**Write a Java program that creates a deadlock scenario with two threads and two resources.

**package** lab7;

**public** **class** DeadlockExample {

**static** **class** Resource {

**private** **final** String name;

**public** Resource(String name) {

**this**.name = name;

}

**public** String getName() {

**return** name;

}

}

**static** **class** Task **implements** Runnable {

**private** **final** Resource resource1;

**private** **final** Resource resource2;

**public** Task(Resource resource1, Resource resource2) {

**this**.resource1 = resource1;

**this**.resource2 = resource2;

}

@Override

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

**synchronized** (resource1) {

System.***out***.println(Thread.*currentThread*().getName() + " locked " + resource1.getName());

**try** {

// Adding delay to make deadlock visible

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

System.***out***.println(Thread.*currentThread*().getName() + " waiting for " + resource2.getName());

**synchronized** (resource2) {

System.***out***.println(Thread.*currentThread*().getName() + " locked " + resource2.getName());

}

}

}

}

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

Resource resourceA = **new** Resource("Resource A");

Resource resourceB = **new** Resource("Resource B");

Thread thread1 = **new** Thread(**new** Task(resourceA, resourceB), "Thread 1");

Thread thread2 = **new** Thread(**new** Task(resourceB, resourceA), "Thread 2");

thread1.start();

thread2.start();

}

}

OutPut:

