**Name: Abinash Satapathy**

**Reg. no.: 16BCE0081**

**Slot: L33 + L34**

**Subject: Java Programming (CSE1007) Lab**

**Multi-threading**

1. **Demonstrate multithreading by creating two threads, one for printing the odd numbers and the other for printing even numbers with in a given range of your choice.**

class TaskEvenOdd implements Runnable {

private int max;

private Printer print;

private boolean isEvenNumber;

// standard constructors

@Override

public void run() {

int number = isEvenNumber ? 2 : 1;

while (number <= max) {

if (isEvenNumber) {

print.printEven(number);

} else {

print.printOdd(number);

}

number += 2;

}

}

}

class Printer {

private volatile boolean isOdd;

synchronized void printEven(int number) {

while (!isOdd) {

try {

wait();

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

}

System.out.println(Thread.currentThread().getName() + ":" + number);

isOdd = false;

notify();

}

synchronized void printOdd(int number) {

while (isOdd) {

try {

wait();

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

}

System.out.println(Thread.currentThread().getName() + ":" + number);

isOdd = true;

notify();

}

}

public class evenodd {

public static void main(String[] args) {

Printer print = new Printer();

Thread t1 = new Thread(new TaskEvenOdd(print, 10, false),"Odd");

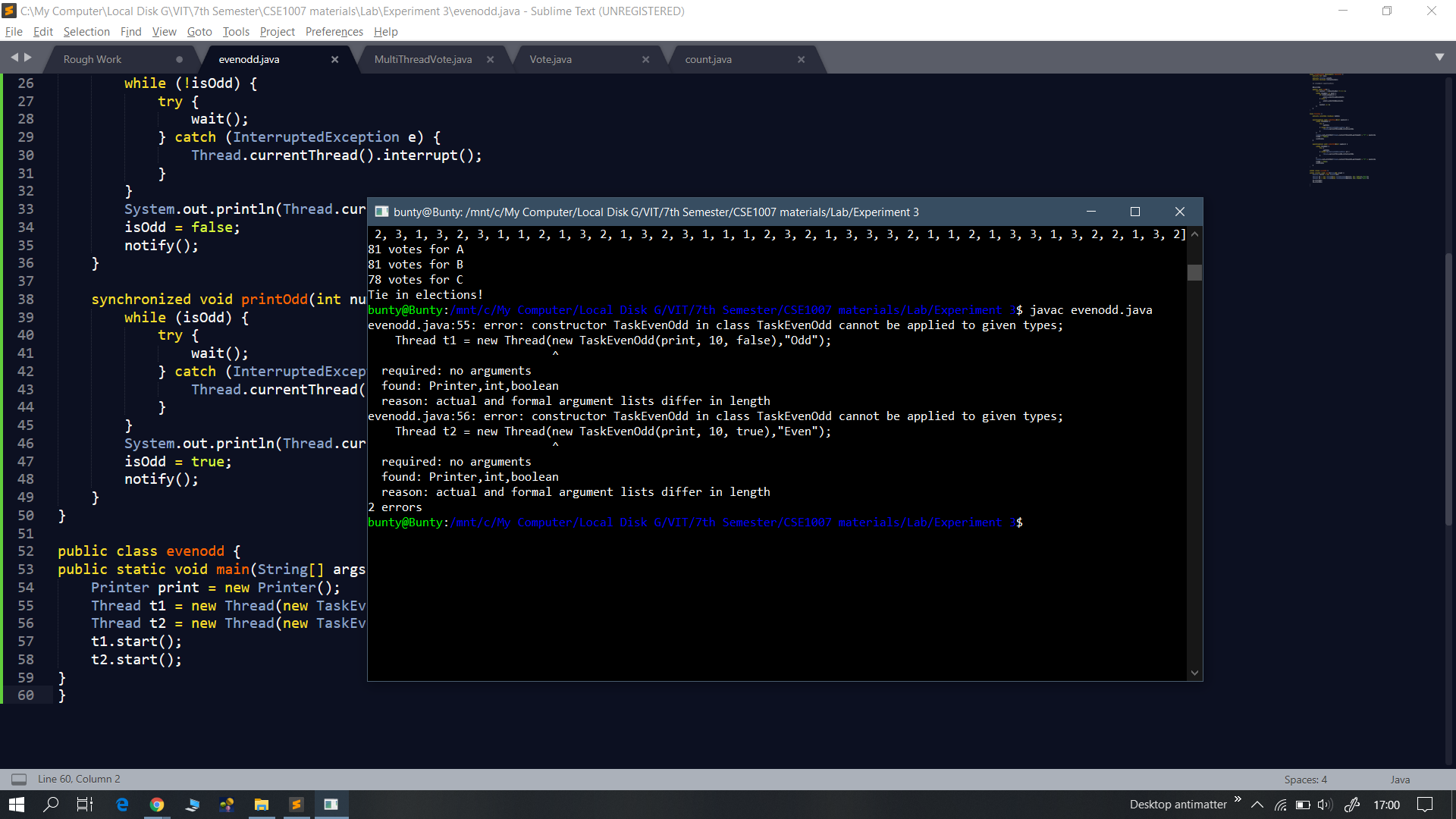
Thread t2 = new Thread(new TaskEvenOdd(print, 10, true),"Even");

t1.start();

t2.start();

}

}



1. **Write a program to demonstrate the knowledge of students in multithreading. Eg., Three students A, B and C of B.Tech- II year contest for the PR election. With the total strength of 240 students in II year, simulate the vote casting by generating 240 random numbers (1 for student A, 2 for B and 3 for C) and store them in an array. Create four threads to equally share the task of counting the number of votes cast for all the three candidates. Use synchronized method or synchronized block to update the three count variables. The main thread should receive the final vote count for all three contestants and hence decide the PR based on the values received.**

package elections;

import java.util.Vector;

public class count extends Thread{

Vector vec;

int k, i;

public int count = 0;

public count(int k, Vector vec){

this.k = k;

this.vec = vec;

}

public void run(){

for(i = 0; i < vec.capacity(); i++){

if(vec.elementAt(i).equals(k)) // check if elements match

count++;

}

}

}

package elections;

import java.util.Random;

import java.util.Vector;

public class Vote extends Thread{

Random rand = new Random(); // generating a random number

int max = 750; // max wait time for thread

int min = 100; // min wait time for thread

int v, s;

Vector vec;

public Vote(int v, Vector vec)

{

this.v = v;

this.vec = vec;

}

public void run() {

try

{

// while voting print id

while(vec.size() < 240) { // ensure size of vote vector is below 240

System.out.println(this.getName() + " is Voting");

vec.add(v);

s = rand.nextInt((max - min) + 1) + min;

System.out.println(this.getName() + " is sleeping for " + s);

Thread.sleep(s); // create random delay between threads

}

}

catch(InterruptedException e)

{

System.out.println("Voting Exception: " + e);

}

}

}

import elections.Vote;

import elections.count;

import java.util.Vector;

public class MultiThreadVote {

public static void main(String[] args) {

Vector votevec = new Vector(240); // creating a vote array for 240 votes

Vote a = new Vote(1, votevec);

a.start();

Vote b = new Vote(2, votevec);

b.start();

Vote c = new Vote(3, votevec);

c.start();

try{

a.join();

b.join();

c.join();

System.out.println("Voting has ended!");

}catch(Exception e){System.out.println(e);}

count ac = new count(1, votevec);

count bc = new count(2, votevec);

count cc = new count(3, votevec);

ac.start();

bc.start();

cc.start();

try{

ac.join();

bc.join();

cc.join();

System.out.println("Counting has ended!");

}catch(Exception e){System.out.println(e);}

int av = ac.count;

int bv = bc.count;

int cv = cc.count;

System.out.println("elections.Vote Vector:" + "\n" + votevec);

System.out.println(av + " votes for A");

System.out.println(bv + " votes for B");

System.out.println(cv + " votes for C");

if(av >= bv && av >= cv){

if(av == bv || av == cv)

System.out.println("Tie in elections!");

else

System.out.println("A has won the elections!");

}

else if(bv >= av && bv >= cv){

if(av == bv || bv == cv)

System.out.println("Tie in elections!");

else

System.out.println("B has won the elections!");

}

else if(cv >= av && cv >= bv){

if(cv == bv || cv == av)

System.out.println("Tie in elections!");

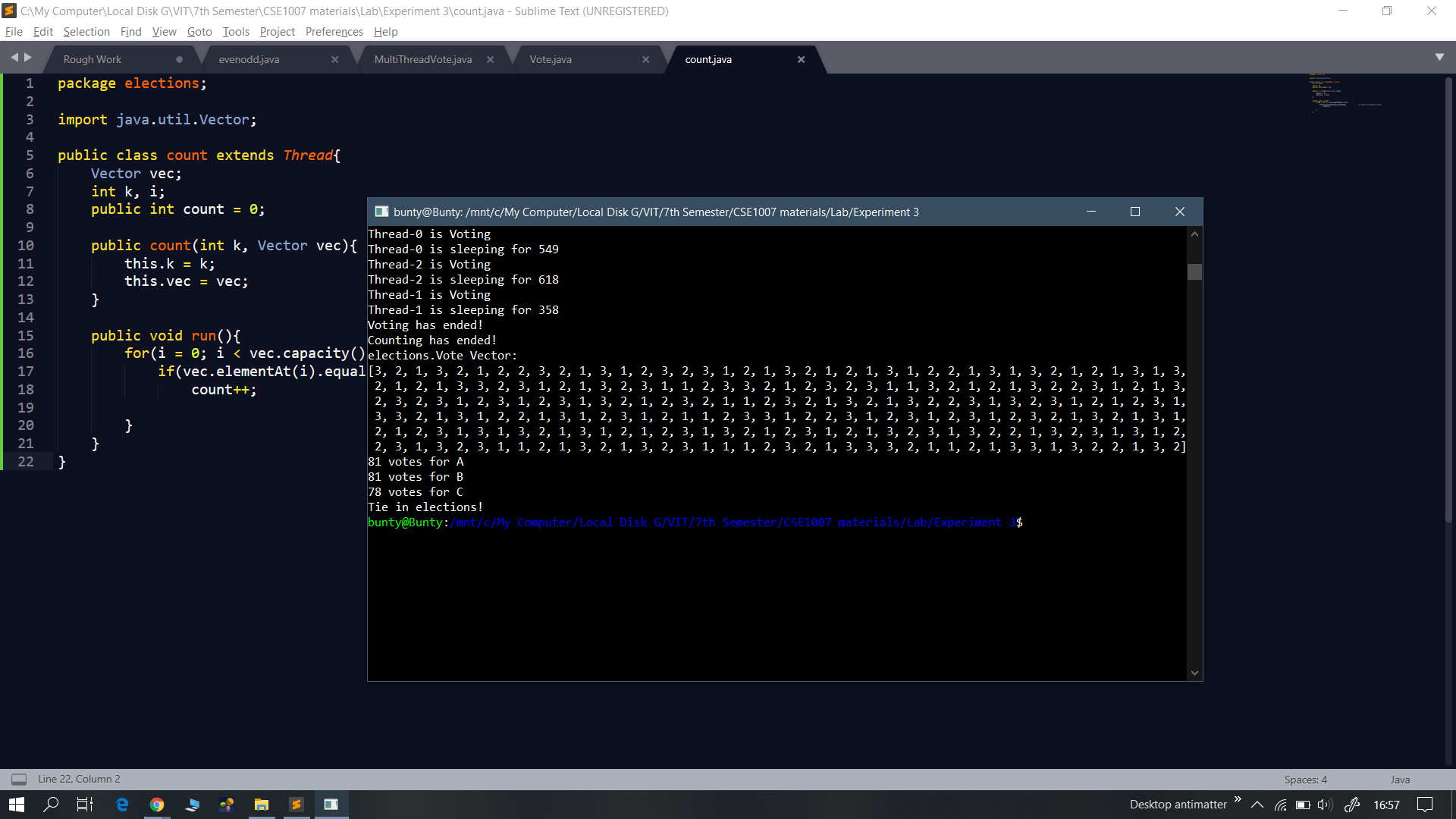
else

System.out.println("C has won the elections!");

}

}

}



**File Handling**

1. **Java Program to Replace First Letter of Every Word with Capital Letter.**

class q1

{

public static void main(String[] args) throws IOException

{

FileReader fr = new FileReader("/home/matlab/file.txt");

String s="";

int i;

while((i = fr.read())!= -1)

s+=Character.toString((char)i);

System.out.println(s);

String[] arr = s.split(" ");

String fin = "";

for(String j:arr)

{

char c = j.charAt(0);

String sub = j.substring(1,j.length());

if(Character.isLowerCase(c))

c = Character.toUpperCase(c);

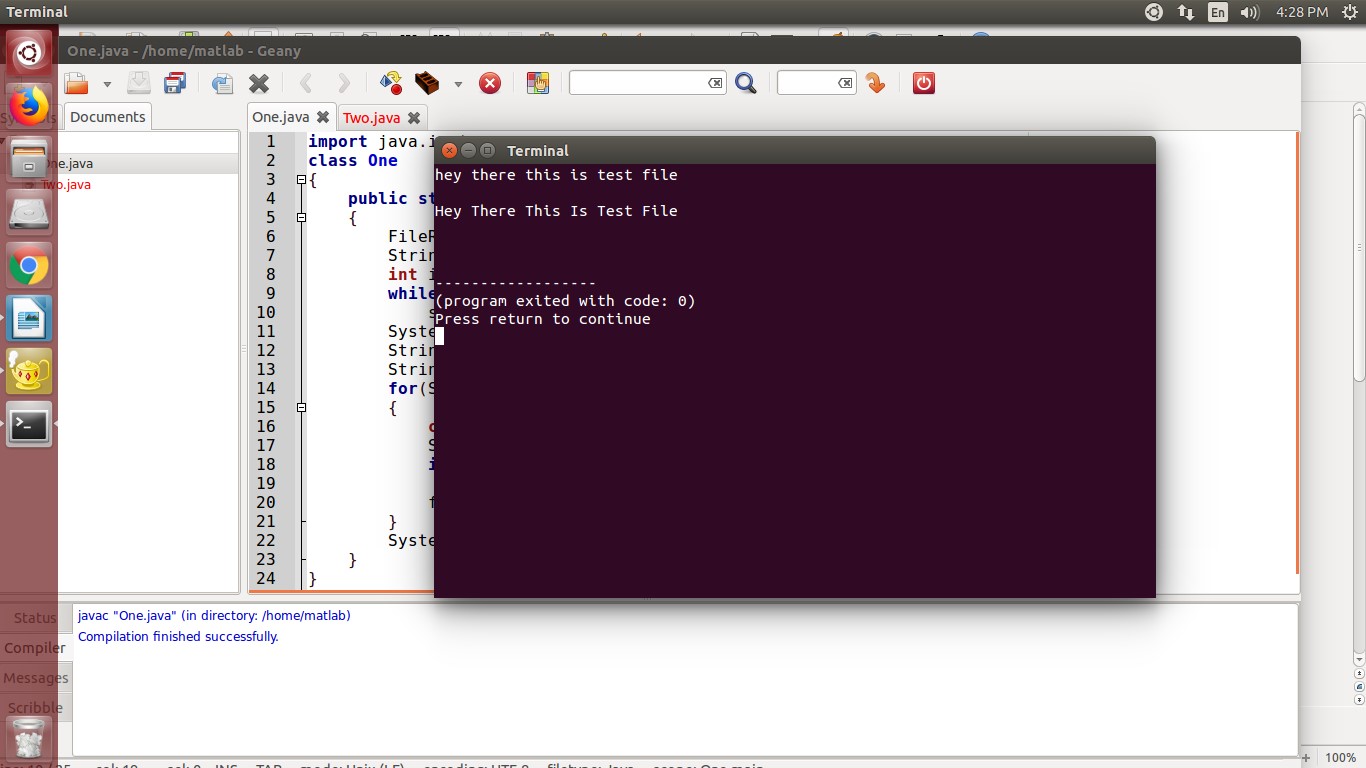
fin+= (c + sub + " ");

}

System.out.println(fin);

}

}



1. **Java Program to Reverse the Contents of a File and Print it.**

import java.io.\*;

class q2

{

public static void main(String[] args) throws IOException

{

FileReader fr = new FileReader("/home/matlab/file.txt");

String s="";

int i;

while((i = fr.read())!= -1)

s+=Character.toString((char)i);

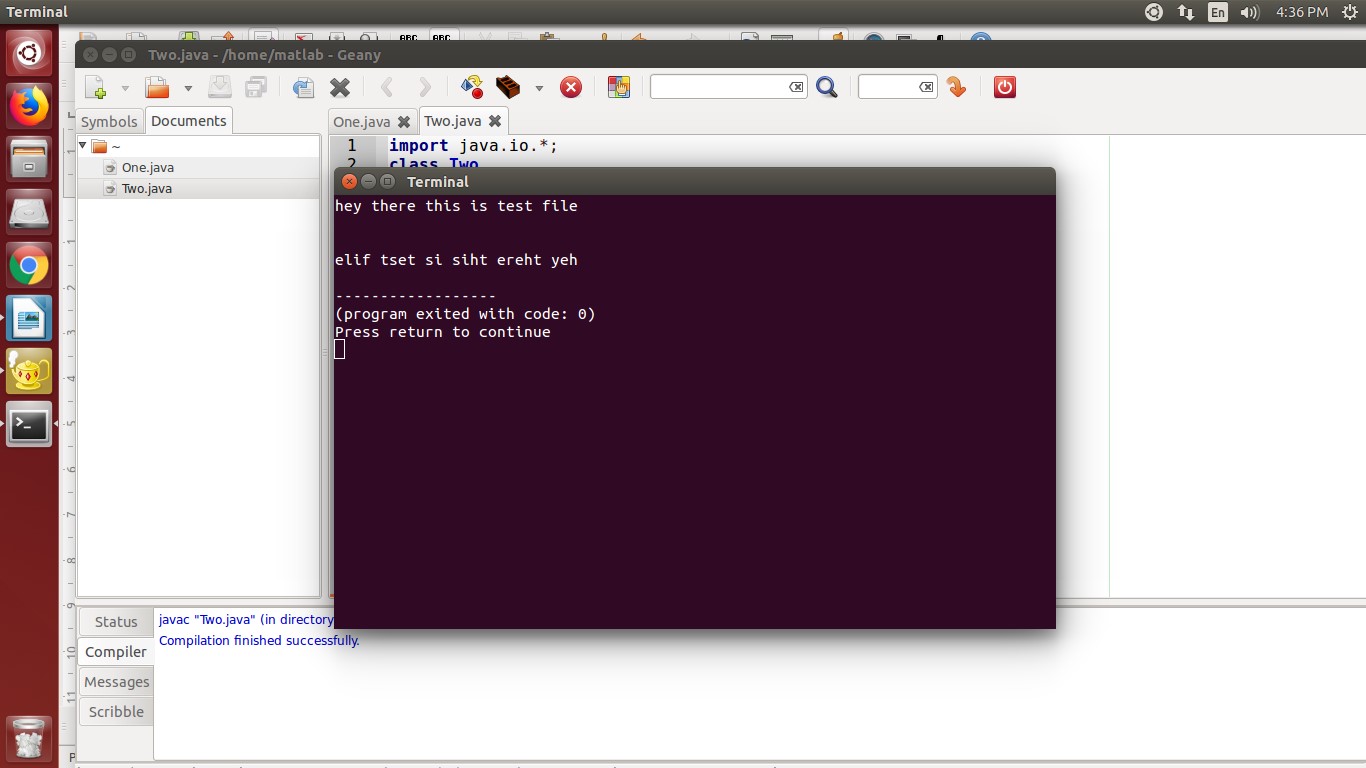
System.out.println(s);

for(int j=s.length()-1;j>=0;j--)

System.out.print(s.charAt(j));

}

}



1. **Java Program to Update Details of Employee Using Files.**

import java.io.\*;

import java.util.Scanner;

class q3

{

public static void main(String[] args) throws IOException

{

Scanner sc = new Scanner(System.in);

FileWriter fw = new FileWriter("/home/matlab/file.txt");

fw.write("Name: ABC\n");

fw.write("ID: 123456\n");

fw.write("Dept: XYZ\n");

fw.close();

FileReader fr = new FileReader("/home/matlab/file.txt");

String s="";

int i;

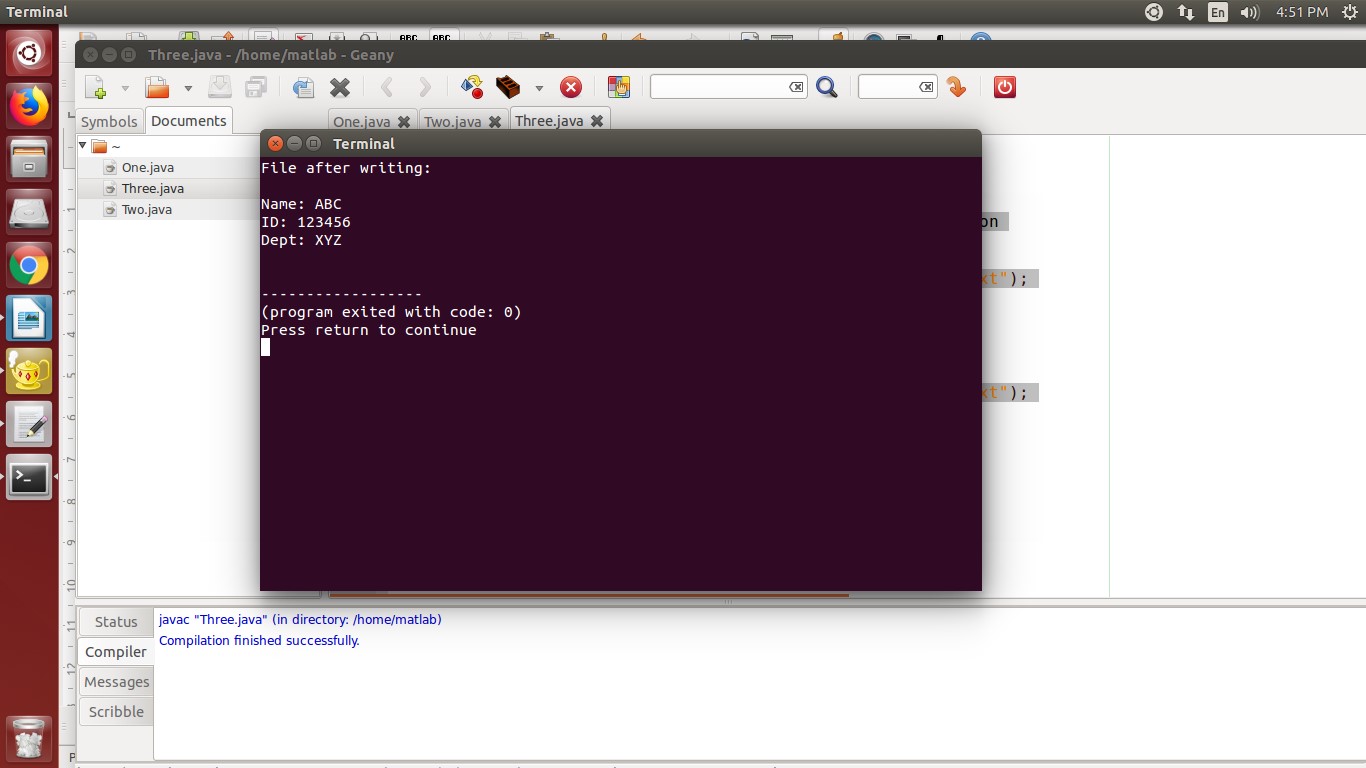
System.out.println("File after writing: \n");

while((i = fr.read())!= -1)

System.out.print((char)i);

}

}



1. **Java Program to Convert the Content of File to LowerCase**

import java.io.\*;

class q4

{

public static void main(String[] args) throws IOException

{

FileReader fr = new FileReader("/home/matlab/file.txt");

String s="";

int i;

System.out.println("File content before writing:");

while((i = fr.read())!= -1)

s+=Character.toString((char)i);

System.out.println(s);

String fin = "";

for(int j=0;j<s.length();j++)

{

fin+=Character.toLowerCase(s.charAt(j));

}

FileWriter fw = new FileWriter("/home/matlab/file.txt");

fw.write(fin);

fw.close();

FileReader fr2 = new FileReader("/home/matlab/file.txt");

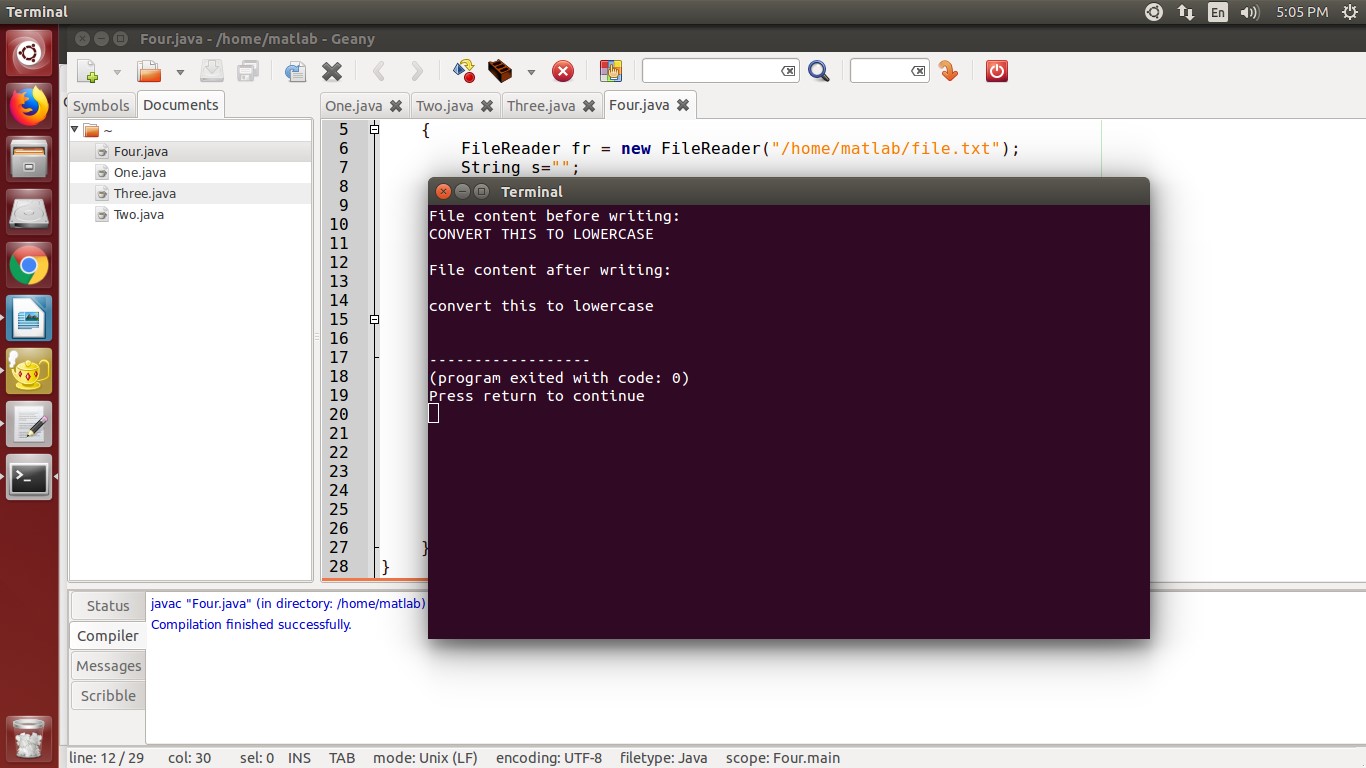
System.out.println("File content after writing: \n");

while((i = fr2.read())!= -1)

System.out.print((char)i);

}

}



1. **Java Program to Create and Count Number of Characters in a File.**

import java.io.\*;

class Five

{

public static void main(String[] args) throws IOException

{

int i,count=0;

String writingString = "Hello World";

File f = new File("/home/matlab/NewFile.txt");

FileWriter fw = new FileWriter(f);

fw.write(writingString);

fw.close();

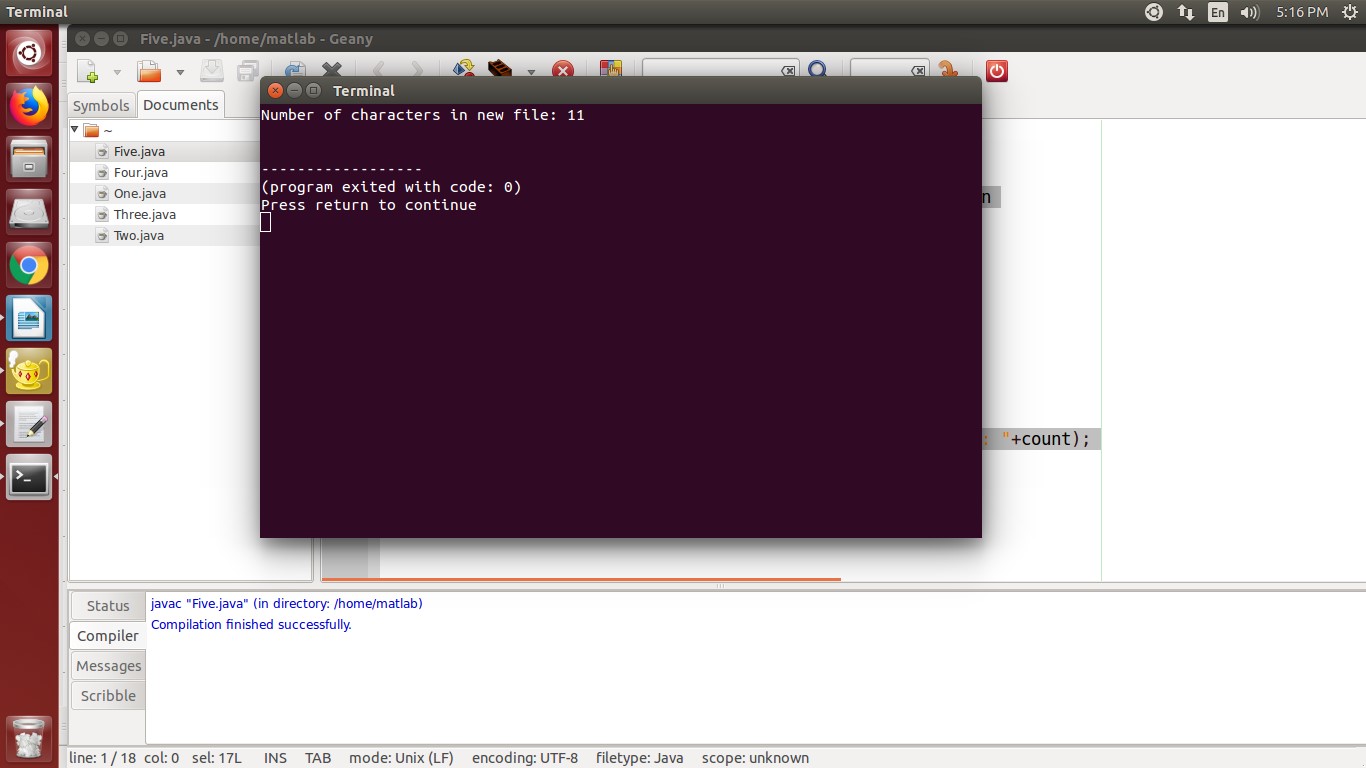
FileReader fr = new FileReader(f);

while((i = fr.read())!= -1)

count++;

System.out.println("Number of characters in new file: "+count);

}



1. **Java Program to Join Lines of Two given Files and Store them in a New file**

import java.io.\*;

public class FileMerge

{

public static void main(String[] args) throws IOException

{

// PrintWriter object for file3.txt

PrintWriter pw = new PrintWriter("file3.txt");

// BufferedReader object for file1.txt

BufferedReader br1 = new BufferedReader(new FileReader("file1.txt"));

BufferedReader br2 = new BufferedReader(new FileReader("file2.txt"));

String line1 = br1.readLine();

String line2 = br2.readLine();

// loop to copy lines of

// file1.txt and file2.txt

// to file3.txt alternatively

while (line1 != null || line2 !=null)

{

if(line1 != null)

{

pw.println(line1);

line1 = br1.readLine();

}

if(line2 != null)

{

pw.println(line2);

line2 = br2.readLine();

}

}

pw.flush();

// closing resources

br1.close();

br2.close();

pw.close();

System.out.println("Merged file1.txt and file2.txt

alternatively into file3.txt");

}

}

1. **Java Program to Collect Statistics of a Source File like Total Lines, Total no. of Blank Lines, Total no. of Lines Ending with Semicolon**

// Java program to count the

// number of characters in a file

import java.io.\*;

public class q7

{

public static void main(String[] args) throws IOException

{

File file = new File("C:\\input.txt");

FileInputStream fileStream = new FileInputStream(file);

InputStreamReader input = new InputStreamReader(fileStream);

BufferedReader reader = new BufferedReader(input);

String line;

// Initializing counters

int countWord = 0;

int sentenceCount = 0;

int characterCount = 0;

int paragraphCount = 1;

int whitespaceCount = 0;

// Reading line by line from the

// file until a null is returned

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

{

if(line.equals(""))

{

paragraphCount++;

}

if(!(line.equals("")))

{

characterCount += line.length();

// \\s+ is the space delimiter in java

String[] wordList = line.split("\\s+");

countWord += wordList.length;

whitespaceCount += countWord -1;

// [!?.:]+ is the sentence delimiter in java

String[] sentenceList = line.split("[!?.:]+");

sentenceCount += sentenceList.length;

}

}

System.out.println("Total word count = " + countWord);

System.out.println("Total number of sentences = " + sentenceCount);

System.out.println("Total number of characters = " + characterCount);

System.out.println("Number of paragraphs = " + paragraphCount);

System.out.println("Total number of whitespaces = " + whitespaceCount);

}

}

