# 

# **Home Assignment Number 4**

This assignment covers topics related to 1. “Exception Handling” 2. “Filing” 3. “Recursion”

**Q1.** (***ArrayIndexOutOfBoundsException***) Write a program that meets the following requirements:

■ Creates an array with **100** randomly chosen integers.

■ Prompts the user to enter the index of the array, then displays the corresponding element value. If the specified index is out of bounds, display the message **Out of Bounds**.

**Q2.** (*Remove text*) Write a program that removes all the occurrences of a specified string from a text file. For example, invoking **java MyProgram Javed filename** removes the string **Javed** from the specified file. Your program should get the arguments from the command line.

**Q3.** (*Count characters, words, and lines in a file*) Write a program that will count the number of characters, words, and lines in a file. Words are separated by whitespace characters. The file name should be passed as a command-line

argument.

**Q4.** (*Compute greatest common divisor using recursion*) The **gcd(m, n)** can also be defined recursively as follows:

■ If **m % n** is **0**, **gcd(m, n)** is **n**.

■ Otherwise, **gcd(m, n)** is **gcd(n, m % n)**.

Write a recursive method to find the GCD. Write a test program that prompts the

user to enter two integers and displays their GCD.

Q5. (*Print the characters in a string reversely*) Write a recursive method that displays a string reversely on the console using the following header:

**public static void** reverseDisplay(String value)

For example, **reverseDisplay("abcd")** displays **dcba**. Write a test program that prompts the user to enter a string and displays its reversal.

**Question1.** (**ArrayIndexOutOfBoundsException**) Write a program that meets the following requirements:

■ Creates an array with **100** randomly chosen integers.

■ Prompts the user to enter the index of the array, then displays the corresponding element value. If the specified index is out of bounds, display the message **Out of Bounds**.

**Source Code:**

package Ruslan\_7kd;

import java.util.\*;

public class Question\_One\_ArrayIndexOutOfBoundEx {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);//Scanner object for input

int[] integerList = getRandomIntegers(100);//caller method to get random integers

System.out.print("Choose index from list of 100 indexes\n--> Enter an index: ");

try{

int index = input.nextInt();

System.out.printf("The element at index %d is %d.",index,integerList[index]);

}catch(InputMismatchException e){//catches exception if input other than int is given

System.out.println("Integer is required");

}

catch(ArrayIndexOutOfBoundsException x){//Out of bound exception is handled here

System.out.println("Out of bound index");

}

}

//Method to return list random integers

public static int[] getRandomIntegers(int a){

int[] list = new int[a];

Random random = new Random();//random class

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

list[i] = random.nextInt(a+1);

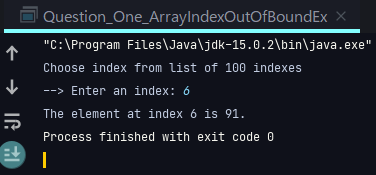
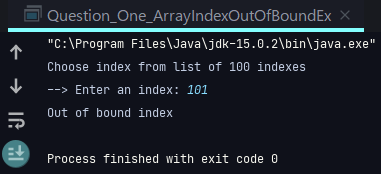
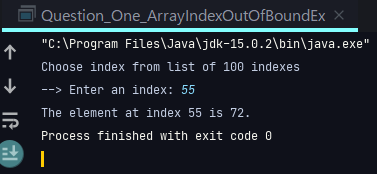
}

return list;

}

}

**OUTPUT**

**Question2.** (*Remove text*) Write a program that removes all the occurrences of a specified string from a text file. For example, invoking **java MyProgram Javed filename** removes the string **Javed** from the specified file. Your program should get the arguments from the command line.

**Source Code**

package Ruslan\_7kd;

import java.util.\*;

import java.io.\*;

public class MyProgram {

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

//input should be from command line

//java MyProgram Javed filename

if(args.length == 0){//in case no argument is passed

System.out.println("Kindly pass an arguments");

System.exit(0);

}

File file = new File(args[1]);

if(!(file.exists())){//in case file doesn't exists

System.out.println("File doesn't exists");

System.exit(1);

}

String removedText = "";

Scanner sc = new Scanner(file);//Reader object

boolean x = false;

while(sc.hasNextLine()){//iterator

String line = sc.nextLine();

if(line.contains(args[0])){

//replaceAll method to replace text

removedText += line.replaceAll( args[0],"")+"\n";

x = true;

}

else removedText += line+"\n";

}

FileWriter writer = new FileWriter(file);

writer.write(removedText);

writer.close();

if(x) System.out.printf("Action has been perform\nThe text %s has been removed",args[0]);

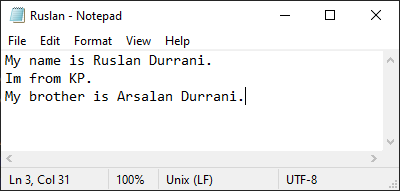
else System.out.println("No such word found.\nTry other word");

}

}

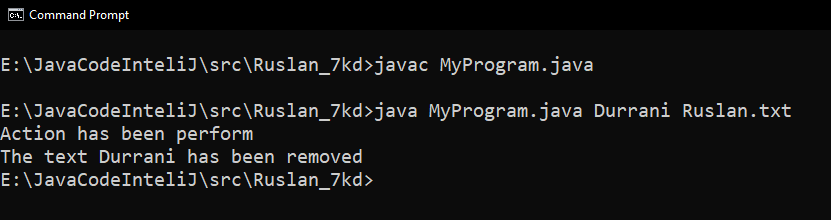
**OUTPUT**

**Text File To Be Read**

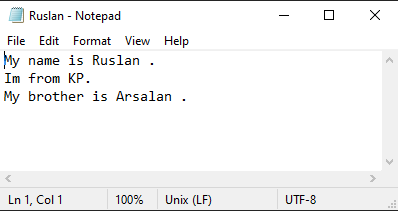


**Command Line Performance**

**Command pattern** *(java MyProgram Javed filename)*



**Updated Text File**



**Question3.** (*Count characters, words, and lines in a file*) Write a program that will count the number of characters, words, and lines in a file. Words are separated by whitespace characters. The file name should be passed as a command-line

argument.

**Source Code:**

package Ruslan\_7kd;

import java.io.\*;

import java.util.\*;

public class CharactersWordsLines {

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

//Instruction will be given from cmd

//File name as parameters from cmd

if(args.length == 0) {

System.out.println("Kindly provide Parameters");

System.exit(1);

}

File myfile = new File(args[0]);

if(!(myfile.exists())){//in case file doesn't exits

System.out.println("File doesn't exists");

System.exit(0);

}

int characterCounter = 0;//character counter

int wordCounter = 0;//word counter

int lineCounter = 0;//Line counter

Scanner reader = new Scanner(myfile);//File Reader

while(reader.hasNextLine()){//Iterator

String line = reader.nextLine();

String[] totalWords = line.split(" ");//Total words Array

String[] totalCharacter = line.split("");//Total characters Array

lineCounter ++;//increment

wordCounter += totalWords.length;//increment

characterCounter += totalCharacter.length;//whiteSpaces characters included

}

//Displaying

System.out.printf("Number of lines are: %d.\n",lineCounter);

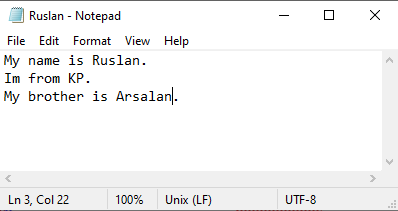
System.out.printf("Number of words are: %d.\n",wordCounter);

System.out.printf("Number of characters are: %d.\n",characterCounter);

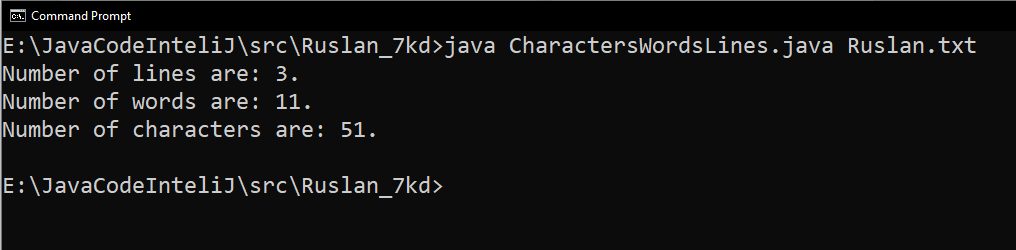
}

}

**OUTPUT** *(words lines characters counting)*



**Command Line Output**



*(White spaces are considered as characters)*

**Question4.** (*Compute greatest common divisor using recursion*) The **gcd(m, n)** can also be defined recursively as follows:

■ If **m % n** is **0**, **gcd(m, n)** is **n**.

■ Otherwise, **gcd(m, n)** is **gcd(n, m % n)**.

**Source Code:**

package Ruslan\_7kd;

import java.util.\*;

public class Gcd\_Recursion {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.println("\*\*\*\* GCD \*\*\*\*");

System.out.print("Enter two numbers: ");

int a = input.nextInt();

int b = input.nextInt();

System.out.printf("The GCD for %d and %d is : %d",a,b,gcd(a,b));

}

//Recursive Method

private static int gcd(int m, int n) {

// Base case

if (m % n == 0) return n;

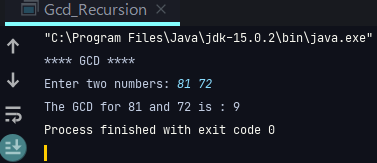
// Recursive call

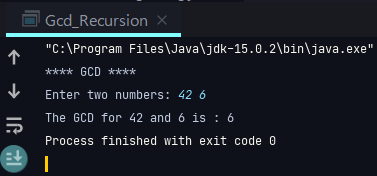
else return gcd(n, m % n);

}

}

**OUTPUT**





**Question5**. (*Print the characters in a string reversely*) Write a recursive method that displays a string reversely on the console using the following header:

**public static void** reverseDisplay(String value)

For example, **reverseDisplay("abcd")** displays **dcba**. Write a test program that prompts the user to enter a string and displays its reversal.

**Source Code**

package Ruslan\_7kd;

import java.util.\*;

public class Question\_5\_reversingString {

public static void main(String[] args) {

//Scanner obj for input

Scanner input = new Scanner(System.in);

System.out.print("Enter a string: ");

String string = input.nextLine();

//Caller method for recursion.

System.out.print("Reversed string: ");

reverseDisplay(string);

}

public static void reverseDisplay(String value){

//Base Case

if(value.length()==0){

return ;

}

else{

System.out.print(value.charAt(value.length()-1));

//Recursive call

reverseDisplay(value.substring(0,value.length()-1));

}

}

}

**OUTPUT**

