Batch: A3 Roll No.: 1911034

Experiment / assignment / tutorial No. 10

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE :File Handling in Java

AIM: Write a program that ask the user to enter the name of the file, and then asks the user to enter a character. The program should count and display the number of times that the specified character appears in the file

Expected OUTCOME of Experiment:

CO4: Explore the interface, exceptions, multithreading, packages.

Books/ Journals/ Websites referred:

1. Ralph Bravaco , Shai Simoson , "Java Programing From the Group Up" Tata McGraw-Hill.

2.Grady Booch, Object Oriented Analysis and Design .

Pre Lab/ Prior Concepts:

File Handling In Java:

File Handling in Java permits us to create, read, update, and delete the files, which are stored on the local file system. There are two types of File handling in Java – FileWriter,
and FileReader, which can perform all the file operations in Java Program.

Types of File Handling in Java
FileWriter and FileReader classes are very frequently used to write and read data from

text files (they are character stream classes).

Java FileWriter:
FileWriter in Java is very useful in creating a file writing character
This class inherits from the OutputStream class.
The constructors of the class FileWriter usually assume that the byte-buffer size an default character encoding is acceptable.
To declare them by oneself we need to construct OutputStreamWriter on a FileOutputStream. Java FileWriter is meant for writing streams of characters.

- **FileWriter(File file) –** This constructor constructs a FileWriter object when a file object is given.
- **FileWriter (File file, boolean append) –** Constructs a FileWriter object.
- **FileWriter (FileDescriptor fd) –** Constructs a FileWriter object associated using a file descriptor.
- **public void write (int c) throws IOException –** Writes a single character.
- **FileWriter (String fileName) – C**onstructs a FileWriter object when a file name is given.
- **public void write (char [] stir) throws IOException –** Writes an array of characters.
- **public void write(String str)throws IOException –** Writes a ***string in Java***.
- **FileWriter (String fileName, Boolean append) –** Constructs a FileWriter object when a file name is given with a Boolean to decide whether it append or not.
- **public void write(String str, int off, int len)throws IOException –** Writes a portion of a string.

Java FileReader:

FileReader (File Handling in Java) uses for reading the data, which are in the form of characters, and it is done from a 'text' file. This class inherits from the InputStreamReader Class.

The constructors of this class are assuming that the default character encoding and the default byte are appropriate. To confirm these values by your own, construct an InputStreamReader on a FileInputStream.

Java FileReader uses for particularly reading streams of character. For reading streams of raw bytes, FileInputStream can use.

- **FileReader(File file) –** This constructor creates a FileReader only when there is File to read from.

- **FileReader(FileDescripter fd) –** Creates a new FileReader when there is a FileDescripter from which it can read from.
- **FileReader(String fileName) –** Creates a new FileReader.

Methods:

- **public int read () throws IOException –** This method reads a single character and also blocks one until another one is available, i.e. an input/output error occurs.
- **public int read(char[] cbuff) throws IOException –** Reads characters into an array. It will block until a character is available.
- **public abstract int read(char[] buff, int off, int len) which throws an IOException –** Use to read characters into a portion of an array. It will block the process until the input is available or an error occurs in input and output, or the stream end reach.

Class Diagram:

public class Exp10
Method : public static void main(String args[])
Variables : char n; int c int i, int count=0; String name;

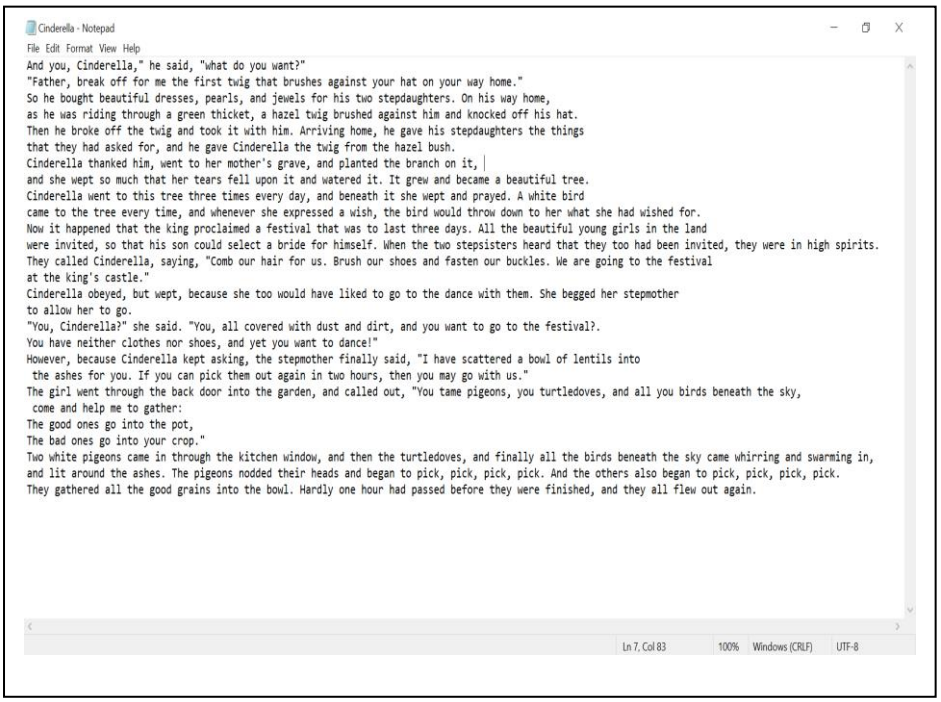
Implementation details:

Code :

```
import java.util.*;
import java.io.*;

public class Exp10
{

public static void main(String args[])
{
char n;
int c,i, count=0;
String name;
Scanner sc = new Scanner(System.in);
System.out.println("Enter the name of the file");
name = sc.next();
try
{
File obj = new File(name + ".txt");
if(!obj.exists()){
      throw new FileNotFoundException ("File Not found");
    }
```

```
else
{

Scanner obj1=new Scanner(obj);
System.out.println("Enter the character whose count you want to find");
n = sc.next().charAt(0);
while(obj1.hasNextLine())
{
String line = obj1.nextLine(); /*as obj1 belongs to scanner class we are
using the hasnextLine method of scanner class.*/
for(i=0;i<line.length();i++)
{
if(line.charAt(i)==n)
{
count = count+1;
}
}
}
obj1.close();
System.out.println("The character "+n+" is found "+count+" number of
times");
}

}
catch(FileNotFoundException e)
{
System.out.println("Some error occured");
e.printStackTrace();
}
}
}
```

Text file used in executing the above code :



Output:

Case when the file is found :

```
C:\Users\arvin\Desktop>javac Exp10.java

C:\Users\arvin\Desktop>java Exp10
Enter the name of the file
Cinderella
Enter the character whose count you want to find
t
The character t is found 187 number of times

C:\Users\arvin\Desktop>java Exp10
Enter the name of the file
Cinderella
Enter the character whose count you want to find
f
The character f is found 31 number of times
```

When the filename specified does not exist (execution of catch block occurs)

```
C:\Users\arvin\Desktop>java Exp10
Enter the name of the file
Aditi
Some error occured
java.io.FileNotFoundException: File Not found
        at Exp10.main(Exp10.java:19)

C:\Users\arvin\Desktop>
```

Conclusion: In this experiment , we have learnt the use of various classes in InputStream as well as OutputStream , for writing and reading bytes of data , as well as the classes FileReader and FileWriter which allow us to read and write character data

into files. We have also been able to execute a program for the implementation of the same.

Post Lab Descriptive Questions

Q.1 How do you write to a file using FileWriter class?

The Java FileWriter Class inherits the output stream class , it is used for writing streams of characters onto files.
The FileWriter object creates the output file , if it is not present already.

Example of using FileWriter class is as follows :

```
import java.io.*;
public class Main {

   public static void main(String args[])throws IOException {
      File infile = new File("Hello.txt");

      // creates the file
      infile.createNewFile();

      // creates a FileWriter Object
      FileWriter w = new FileWriter(infile);

      // Writes the content to the file
      w.write("Java is an Object Oriented Programming Language");
      w.flush();
      w.close();

      // Creates a FileReader Object
      FileReader fr = new FileReader(infile);
      char [] a = new char[50];
      fr.read(a);   // reads the content to the array

      for(char c : a)
         System.out.print(c);   // prints the characters one by one
      fr.close();
   }
}
```

Output of the above program will be as follows :

```
: javac -classpath .:/run_dir/junit-4.12.jar:target/depende
ncy/* -d . Main.java
: java -classpath .:/run_dir/junit-4.12.jar:target/depender
cy/* Main
Java is an Object Oriented Programming Language:
```

Q.2 What is the use of PrintWriter class?

The **Java.io.PrintWriter** class prints formatted representations of objects to a text-output stream.

Following are the fields for **Java.io.PrintWriter** class −

- **protected Writer out** − This is the character-output stream of this PrintWriter.
- **protected Object lock** − This is the object used to synchronize operations on this stream.

Date: **Signature of faculty in-charge**