LAB : FILE PROCESSING AND EXCEPTION HANDLING

1. OBJECTIVE

The objective of this Lab is to practise on file input/output processing and exception handling.

2. LABORATORY

This lab is conducted in the Computing Lab 2.

3. EQUIPMENT

Hardware: Workstation/PC running under the *Linux/Windows* environment.
Software: text editor, jGRASP
Java 2 Platform

4. INTRODUCTION

Java provides a set of input/output streams to store and retrieve data from files. A typical file processing task normally involves reading data from some input file, performing computation using the data, and writing the results of computation to one or more output files. A database is a collection of data organized in a way that allows efficient access to the data. Databases are sometimes organized as a collection of records, with each record containing all the related data items for one entity (such as an employee, department, etc.). We will practise on writing file I/O processing in this lab.

5. EXPERIMENT

Assume you have a file named "names.txt" which contains single names separated by newline characters and no blank lines. For example:

names.txt

```
Alex
Beckham
Giggs
Blanc
Neville
Scholls
Keane
McDonald
KennyRoger
BurgerKing
AandW
PHut
```

Write a Java program (with the necessary I/O exception handling code) which will sort the names into two files based on their length. Names with 5 letters or less should be written to a file named "small.txt". Names with more than 5 letters should be written to a file named "big.txt". Maintain the relative order of the names as they appear in "names.txt". Each name should be followed by a newline character. Neither file should have any blank line. Given the above "names.txt" file, your program should produce the following two files:

small.txt **large.txt**

Alex Giggs Blanc Keane AandW PHut

Beckham Neville Scholls McDonald KennyRoger BurgerKing

Create the "names.txt" file in your directory. For the names in "names.txt", you may assume that:

- Each name is less than 20 characters (not counting the newline character).
- Each line contains a valid name (all characters, no space).
- Each name is terminated by a newline character.
- There is at least one name with 5 or less than 5 characters.
- There is at least one name with more than 5 characetrs.
- There is no blank space or blank line at the end of the file.

Important:
Remember to do all the programming inside the sub-directory *lab8* and name the source code of this program as **Lab8p.java** and name the compiled class code as **Lab8p.class**.

Test Data:
Use the "names.txt" as your test data.

Expected outputs: as shown from the sample data.