File Access

- Reading: Savitch, Chapter 9

Objectives

- To learn textfile accessing

What is a file?

- A file is a sequence of data elements residing in a secondary storage (eg. hard disks, cds, tapes etc).
  - If the data elements are characters, the file is a text file.
  - If the data elements are binary, the file is a binary file.

Example

HelloWorld.java, HelloWorld.class and winword.exe

HelloWorld.java -- a text file.

Winword.exe -- a binary file.

Why use files?

- To save information.

Example

When running a JAVA program, we can save the output as a file, so that we can access the output later.We can also save the input as a file, so that we don't have to key in the input each time when we run the program.

File Accessing

- File accessing refers to
  - reading data from a file (file reading), or
  - writing data to a file (file writing).

- Classes defined in *java.io* can be used to handle file accessing.

Text file accessing

Text file accessing involves
the following steps

(1) Define a FileReader/FileWriter object for reading/writing

Example

```
FileReader in = new FileReader ("source.txt");

FileWriter out = new FileWriter ("dest.txt");
```

(2) Use read()/write() method to read/write a character.

Example

```
char c = (char) in.read();
out.write(c);
```

(3) Close the file when writing is completed.

Example

```
out.close();
```

Note:

- read() reads char by char and returns the unicode of each character (which is an integer). Type cast is needed to convert the returned value to a char.

- read() returns –1 when it reaches the end of the file.

Example

```
//FileReaderWriterTest.java
//This program reads from source.txt char
//by char and writes to dest.txt

import java.io.*;
public class FileReaderWriterTest
{
    public static void main(String [ ] args)
        throws IOException  {
```

```
int content;

 // create the FileReader object
FileReader in = new FileReader ("source.txt");

 //  create the FileWriter object
FileWriter out = new FileWriter ("dest.txt");

content = in.read();
```

```
        while (content != -1)  {
            out.write((char)content);
            content = in.read();
        }
    out.close();
    }
}
```

Use readLine() and println()

readLine()/println() can be used to read/write line by line in text file accessing.

- readLine() is a method defined in the BufferedReader class.
- println() is a method of the PrintWriter class.

Example

```
//FileToUppercaseFile.java
//reads from a file line by line and echoes
//the contents to another file in uppercase

import java.io.*;
public class FileToUppercaseFile  {
   public static void main(String[] args)
      throws IOException  {
```

```
BufferedReader in = new BufferedReader
        (new FileReader (“source.txt”));
PrintWriter out = new PrintWriter (new
        BufferedWriter(new FileWriter
        (“dest.txt”)));

    String inputLine;

    inputLine = in.readLine();
```

```
while (inputLine != null)  {
        out.println(inputLine.toUpperCase());
        inputLine = in.readLine();
    }
    out.close();
   }
}
```

Class Exercise

(1) Modify the previous example so that the output will not go to *dest.txt*, but to the monitor.

(2) Count how many ‘s’ in *source.txt*.

Hint: use **indexOf**(int ch, int fromIndex)