# Assignment 4

**Due: Wednesday, March 1 by 11:59 PM**

## Objective

This assignment should give you practice with accessing the file system using the Java File class as well as the basics of using input/output streams. It will also give you more practice with command line arguments and some practice formatting dates.

## Task

Write a file viewer that allows you to view file information, view directory structure, read text files, and copy files. Your main program should be run from the file:

- `FileViewer.java`

You should not **need** any other files or classes, but you may create them if you wish.

**Each file should have a comment including your name at the top of the file. Each file should also have appropriate comments throughout.**

---

## Requirements

1. **Command Line Options:** Your program should accept the following command line options:

   - "-i" (for (i)nformation) with an optional file or directory as a 2nd parameter. If no 2nd parameter is passed, default to the current directory "."
   - "-v" (for (v)iew) with a file to view as the 2nd parameter
   - "-c" (for (c)opy) with a sourcefile as the 2nd parameter and a destination file as the 3rd parameter
   - If no parameters are passed, then the program should default to displaying information on the current directory (as if the user had passed "-i .")

   If the command is invalid usage (illegal options), print a usage message, as below, then exit the program:

```
Usage: java -jar hw4.jar [-i [<file>|<directory>]|-v <file>|-c <sourceFile> <destFile>]
```

   Since that text has to be small to fit on the line, I'll break it up into multiple lines here (make sure you print it all on one line in your code):

```
Usage: java -jar hw4.jar
[-i [<file>|<directory>]|-v <file>|-c <sourceFile> <destFile>]
```

2. **Information:** When the user requests information using the "-i" option, they may pass a 2nd parameter indicating the file or directory they wish information on. This is handled in different ways, depending on whether it is a file or directory. If it is a file:

   - Print the absolute path to the file
   - Indicate whether the file can be executed
   - Print out the file size in bytes
   - Print out the last modified date of the file
   - Your output should match mine exactly (see the example output for the exact format)

   However, if it is a directory:

   - Print a heading indicating that you are printing size followed by filename
   - Print each file/directory contained in the directory being viewed
   - For each file that you are printing, print the size, followed by the filename
   - For directories, print the size as *
   - Sort the files by filesize (lowest to highest)
   - See example output for details

   If the 2nd parameter is not a regular file or directory, print the error message:
   "Error: Invalid File"

3. **View:** If the user requests to view a file, the 2nd parameter should be a regular file. If the file is not found, print an appropriate error message. You should then print the contents of the file, as text, to the screen. Remember to handle all exceptions you encounter.

4. **Copy:** If the user requests to copy a file, the 2nd parameter should be the name of a regular file (the source file) and the 3rd parameter should be the name of the file to copy to (the destination file). If the destination file already exists, do not make the copy and print an appropriate error message. If the source file is not valid, print an appropriate error message. Then you should copy the information from the source file to the destination file, handling any exceptions encountered appropriately. After this, print a message to the user stating that the copy was successful. Please note that you should be able to copy binary files with this method, not just text files.

5. **Other Requirements:**

   - Make sure you handle all exceptions that could be thrown when dealing with the file I/O
   - When printing error messages to the user, you do not have to match my format exactly (although it is preferred)
   - For all other output, match my format and text exactly

6. **Extra Credit:** You may earn extra credit on this assignment by allowing the user to specify an additional command-line option: "-d" for compare (d)ifferences in files. This option should take 2 more parameters, both of which should be files. It should compare the two files and print a message indicating whether they are the same. As in the other options, handle exceptions. Please note that this changes the usage message. The new usage message is as follows (Please note this is on multiple lines, but your actual message should all print on the same line)

```
Usage: java -jar hw4.jar [-i [<file>|<directory>]|
-v <file>|-c <sourceFile> <destFile>|-d <file1> <file2>]
```

7. **Hints:**

   - Remember that file sizes are returned via the `length()` function and are of type `long` (which can be converted to type `Long` using a class method of class `Long` - This can be very useful when attempting to sort)
   - You can print the last modified date using the Java `Date` class along with a `SimpleDateFormat` class (this is a good place to start)
   - You could also print the last modified date using `printf`. Look at the Formatter class for a detailed listing of the date/time flags. Remember to use the "<" flag as well to keep using the same parameter (so you don't have to pass the date in multiple times). As an example:

     ```
     System.out.printf("%b %<d %<Y", date)
     ```

     would print out

     ```
     Jan 31 2016
     ```

     if the variable `date` was a long integer representing the date Jan 31, 2016.
   - The display option only has to handle text files, all other options should handle binary files as well (so use the appropriate streams).
   - You may use `System.exit(0)` to exit the program at any time
   - Make sure you handle all invalid commands (including too many command-line parameters)

**Some Helpful Classes/Methods**

- `String.equals(Object)`
- `String.charAt(int)`
- `File class`
- `Arrays.sort(T[], Comparator)`
- `Comparator Interface`
- `Date(long)`
- `DateFormat.format(Date)`
- `Long.valueOf(long)`

---

## Sample Output

(This example has the extra credit implemented)
(commands are underlined)
(these commands were all issued in order in the same directory)

```
java -jar hw4.jar

Size  Filename

26    manifest.mf
767   FileViewer$1.class
1018  aTextFile
1018  aCopyOfATextFile
*     sol
6261  FileViewer.class
6683  hw4.jar
7140  FileViewer.java

java -jar hw4.jar -i ~/cop3252/

Size  Filename

*     another
*     classExamples
*     hw
*     exams
*     lec
*     cpdj
```

java -jar hw4.jar -i aTextFile

```
File Path: /home/grads/thrasher/cop3252/hw/hw4/aTextFile
Is executable? false
Size: 1018 bytes
Last Modified Date:  Wed Nov 23 15:36:21
```

java -jar hw4.jar -v aTextFile

```
"Jabberwocky"

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

He took his vorpal sword in hand:
Long time the manxome foe he sought
So rested he by the Tumtum tree,
And stood awhile in thought.

And as in uffish thought he stood,
The Jabberwock, with eyes of flame,
Came whiffling through the tulgey wood,
And burbled as it came!

One, two! One, two! and through and through
The vorpal blade went snicker-snack!
He left it dead, and with its head
He went galumphing back.

"And hast thou slain the Jabberwock?
Come to my arms, my beamish boy!
O frabjous day! Callooh! Callay!"
He chortled in his joy.

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.
```

```
-- from Through the Looking-Glass, and What Alice Found There (1871)

java -jar hw4.jar -c aTextFile aCopyOfATextFile
File aCopyOfATextFile already exists.

rm aCopyOfATextFile

java -jar hw4.jar -c aTextFile aCopyOfATextFile
File aTextFile was copied to file aCopyOfATextFile successfully.

java -jar hw4.jar -d aTextFile aCopyOfATextFile
The two files aTextFile and aCopyOfATextFile are the same.

java -jar hw4.jar -d aTextFile FileViewer.class
The two files aTextFile and FileViewer.class are not the same.

java -jar hw4.jar -q
Usage: java -jar hw4.jar [-i [<file>|<directory>]|-v <file>|-c <sourceFile> <des
```

---

## Submitting

Pack all of your files (class files **and** source code) into a **fully runnable** JAR file called `hw4.jar`. The main program that the jar file should execute should be in class `FileViewer`. I should be able to run the main() method from your file with the command:

```
java -jar hw4.jar
```

Submit your jar file via the Blackboard submission link for assignment 4.