Rochester Institute of Technology
National Technical Institute for the Deaf
Information and Computing Studies Department

**Name:** ____________________

# NACA.161 Programming Fundamentals II
# In-Class Exercise #25 – File IO - Reading

## *Overview*

This exercise is designed to let you start reading rile using the various IO classes.

## *Connect to a File*

1) Download the file called **in.txt** from the myCourses content area.

2) Look at this file in a text editor to see what you're trying to read.

3) Create a class called **Reading** that contains a main method. We will use this class to do this exercise.

4) Write the code to prompt the user to enter the name of the file. Compile and fix all errors before continuing.

5) Create an object that allows you to obtain information about a file,

   What IO class did you use?

   FileReader

   What package is it in?

   java.io.*;

   Compile and fix all errors before continuing.

6) Write the code that will display a message that indicates whether or not the file exists. Make sure you include the name of the file in your message.

What method did you use?

try catch

7) Run the program and enter **in.txt** for the filename

Does it exist? Yes

8) Run the program and enter the file name **bad.txt**.

Does it exist? No

Test your program and fix all errors before continuing.

9) Use your file object to obtain the following information about this file and indicate which method you called:

a) Name of the file: in.txt

Method used: getName()

b) Size of the file (in bytes) : 141

Method used: .length()

c) Is the file readable? Yes

Method used: in.exists();

d) The absolute path of the file: ...\Progra Fintmtl II\ICE\Riley_ICE 25\ in.txt

Method used: .getPathName()

10) We will now attempt to read the file.

What IO class do you use to read a file?

FileReader fr = new FileReader(f);

11) Which class can you use to read the data in the file character by character?

fr. read() ;

Inside your if-statement that indicates that the user-input file exists, open the file for reading using the File class as an argument to the constructor of the class above.

Write the line of code to open the file:

if (f. exists() == true)

12) Compile and run the program.

Did it compile? no

What error did you get?

NullPointerException

13) Add a try/catch statement around this line and only catch the specific exception type indicated in the compile error message. Print the string returned by the **toString** method if the exception occurs.

What exception must be caught?

Null Pointer Exception

What is the code you used to print the return value from the **toString** method?

(char)

14) Now add a while-loop to your program to read every character of the file.

What method will get the next character from the file?

new String

What data type is returned?

Char

How can you detect when you have reached the end of the file?

When result is -1

What is the condition in your while-loop?

While (i != -1) [illegible]

15) Compile the code.

You should receive the following errors:

a) unreported exception java.io.IOException: must be caught or declared to be thrown

b) variable in might not have been initialized

What do you need to do to fix the first error?

put them in try block

What do you need to do to fix the second error?

int i = 0

16) Fix all the errors before continuing.

17) Inside the while loop add code that displays each character by doing the following:
   a) Use an int variable to store each return from the read( ) method.
   b) Display this int without any typecasting.
   Compile and run the code

18) If you run the code as described above you should see something like:

```
84104105115321051153210810511010132481310841041051153210511532108105 1
1010132491310841041051153210511532108105110101325013108410410511532 10
51153210810511010132511310841041051153210511532108105110101325213108 4
10410511532105115321081051101013253131084104105115321051153210810511 0
10132541310841041051153210511532108105110101325513108410410511532105 1
1532108105110101325613108410410511532105115321081051101
```

Not exactly the same output as a text editor.

The problem is that the **read()** method reads a character at a time but puts it into an int variable. This was done in order to return a special value (-1) to indicate when you have reached the end of the file.

Unfortunately, the **System.out.println** method doesn't know anything about all this and it thinks the variable is an int (it doesn't know that it is really a character stored inside an integer). So it prints each character as a number. Which is NOT what we want.

This problem is "simple" to fix. Just cast the int variable to a char before you print it. This way you are telling the **System.out.println** method to print characters, not numbers.

19) Fix the code as described above, recompile, and run your program. If all went well you should see exactly what you saw when you looked at it in the text editor.

20) Now add code to close the file and add any required exception handling code.

21) What integer values of the two characters that are used to indicate the end of the line?

10 + 13 [illegible]

Now add code to count the number of characters that are not used to indicate "end of line".

Signoff

When you complete all of the steps successfully and answer all of the questions, contact your instructor to check if your application(s) executes correctly and to review your code. We will initial the line below

_________Successful execution of code

If you do not finish the program during the class period, contact your instructor to check to review your code and initial below.

_________Code not completed during lab time

You may then submit your work at the start of next class. You may not use the work period of the next class to complete this assignment. If you do not have a signature, then you cannot receive any points for this assignment.