Lesson 36: Processing File Input with *Scanner*

We are going to illustrate the use of the *Scanner* class with lines of text that we input from a file. First, we will look at how to process **numbers** that are embedded in the text that makes up the various lines of an ASCII text file.

Suppose we consider text files with the following properties:

1. They will have an unknown number of lines of text.
2. Each line of text consists of an unknown number of integers separated by spaces.

Following is an example of the contents of such a file (*NumData.in* stored in your standard *temp_Name* folder):

```
12 10 3 5
18 1 5 92 6 8
2 9 3 22 4 11 7
```

Adapting to unpredictability:

When we write our program, we want to remember that this file has elements of unpredictability. There are an **unpredictable number of lines of text**. Furthermore, each line of text contains an **unpredictable number of integers**.

Here is our task. We are to input the lines of text and then print the sum of the numbers in each line. For example, the sum of the numbers in the first line is:

12 + 10 + 3 + 5 = 30

Similarly, the other lines of text yield:

18 + 1 + 5 + 92 + 6 + 8 = 130
2 + 9 + 3 + 22 + 4 + 11 + 7 = 58

We are required to process the data in such a way that the final printout appears as follows:

```
12 + 10 + 3 + 5 = 30
18 + 1 + 5 + 92 + 6 + 8 = 130
2 + 9 + 3 + 22 + 4 + 11 + 7 = 58
```

Begin the new class:

Let's begin our new *InputNumData* class as follows:

```
import java.io.*; //necessary for File and IOException
import java.util.*; //necessary for Scanner
public class InputNumData
{
        public static void main( String args[] ) throws IOException
```

```
        {
            Scanner sf = new Scanner(new
            File("C:\\temp_Name\\NumData.in"));
            int maxIndx = -1;
            //-1 so when we increment below, the first index is 0

            String text[] = new String[1000];
            //To be safe, declare more than we need

            while(sf.hasNext( ))
            {
                maxIndx++;
                text[maxIndx] = sf.nextLine( );
                //System.out.println(text[maxIndx]); //Remove rem for
                testing
            }
            //maxIndx is now the highest index of text[], -1 if no text lines

            sf.close( );
            //We opened a file above, so close it when finished.
            // …process the text[] array…
        }
    }
```

Notice that the *while*-loop automatically adjusts to an unpredictable number of lines of text. We exit the loop with the lines of text stored in *text[ ]* and with *maxIndx* being the highest index.

Processing the text:

Our real job here is to fill in code in the area of **…process the *text[ ]* array…**. To do this, we will set up a loop to process the *text[ ]* elements. As the first line inside the loop, we will create another *Scanner* object where *text[j]* is the *String* to be tokenized (parsed).

```
for(int j =0; j <= maxIndx; j++)
{
    Scanner sc = new Scanner(text[j]);
    //Notice we create a new object each time through the loop
    • • •
}
```

Now, we will adjust to the unpredictable number of integers in each line of text by using a *while* loop and the *Scanner* method *hasNext( )* as follows:

```
String answer = “”; //We will accumulate the answer string here.
int sum; //accumulates sum of integers
for(int j =0; j <= maxIndx; j++)
{
```

```
            Scanner sc = new Scanner(text[j]);
            sum = 0; //important to set to 0; otherwise it will remember the last sum
            answer = “”; //otherwise it will remember last answer String

            while( sc.hasNext( ) ) //We could also have used hasNextInt( ) here
            {
                  int i = sc.nextInt( );
                  answer = answer + “ + ” + i;
                  sum = sum + i;
            }
            answer = answer + “ = ” + sum;
            System.out.println(answer);
      }
```

As we learned in Lesson 7, use of the *Scanner* class necessitates the import of *java.util.**.

The resulting printout:

```
+ 12 + 10 + 3 + 5 = 30
+ 18 + 1 + 5 + 92 + 6 + 8 = 130
+ 2 + 9 + 3 + 22 + 4 + 11 + 7 = 58
```