CSE21 : Lab #13 – 2D Arrays & File Output

Overview

This week we will read more than one file as input and create an output file also. For that purpose we will actually take in two matrices and multiply them. The result is then written to a file. Most of the code and logic you need to finish this lab is presented in lecture and also included in this lab assignment. Your job is to combine each portion to do the job correctly.

Getting started

After starting Eclipse, create a new project called Lab 21_13. Import MultOutput.java from the assignment page. You will also need InNum1.txt, InNum2_1.txt, InNum2_2.txt and InNum2_3.txt to test your program with. Note: if you are using Eclipse on Windows computer that put it in the Project directory (instead of /src).

We need to read in the matrix from two files. For that we don't need to write two different codes that are essentially the same except for the name of the file. For that we will be using a method that takes the filename as a parameter. First let's look at input files which have the following features:

- First Line contains matrix dimension information
  - Row
  - Column
- Matrix values are printed in corresponding rows and columns

```
3     3
1     2     3
3     2     1
1     3     2
```

This means 3x3 Matrix with the following values (first line):

1	2	3
3	2	1
1	3	2

We will introduce is the new keyword `null` which means an 'empty' object that the pointer can point to. We can also check if a pointer is valid by checking it against the object, `null`. Here is a sample method that can use as a basis for your lab:

```
1  public static int[][] proc(String filename) {
2     int [][] arr = null;
3     try {
4           Scanner sc = new Scanner ( new FileReader(filename) );
5           int row = sc.nextInt();
6           int column = sc.nextInt();
7           arr = new int[row][column];

8           for (int i = 0 ; i < row; i++)
9                 for (int j = 0 ; j < column; j++)
10                      arr[i][j] = sc.nextInt();
11          sc.close();
12    } catch ( NoSuchElementException e){
13          System.out.println(e);

14    } catch (FileNotFoundException e) {
15          System.out.println(e);
      }
16    return arr;
   }
```

Q1. What is the return type of the method, proc()? What line in the above code can you find this information?

Q2. What is line 2 doing? (Answer in terms of variable name, type and initial value)

Q3. Why do we need to use `try` in line 3?

Q4. What line(s) read in the dimension of the matrix?

Q5. What is line 7 doing?

Q6. Why is there no while-loop in this method and we can use a for-loop?

Q7. How many times does line 10 get executed? (answer in terms of # of rows and # of columns)

```
public static int[][] multiply(int[][] m1, int[][] m2) {
      int m1rows = m1.length;
      int m1cols = m1[0].length;
      int m2rows = m2.length;
      int m2cols = m2[0].length;
      int[][] result = new int[m1rows][m2cols];

      for (int i=0; i<m1rows; i++)
            for (int j=0; j<m2cols; j++)
                  for (int k=0; k<m1cols; k++)
                        result[i][j] += m1[i][k] * m2[k][j];
      return result;
}
```

Matrix multiplication is done with the above code. Your job is to combine the multiply with output to create multOutput(). This code example shows how to print a 2D array to a file called Result.txt. Note: you will need to modify it in order for it to do everything required in the lab. (It does not print out dimensions yet) You can combine the above "multiply" with this "output" all in multOutput() method so it is all part of the 3 nested for-loops.

```
String filename = "Result.txt";

try {
      FileWriter output = new FileWriter(filename);

      String ostr = "";
      for (int i = 0; i < arr2D.length; i++) {
            for (int j = 0; j < arr2D[0].length; j++) {
                  System.out.print(ostr = (arr2D[i][j] + "\t"));
                  output.write(ostr);
            }
            System.out.println();
            output.write("\r\n"); // Carriage return
      }
      output.close();
} catch (Exception e) {
      System.out.println(e);
}
```

Q8. What does `arr2D.length` mean?

Q9. What does `arr2D[0].length` mean?

Q10. What does `arr2D[i][j] + "\t"` create?

(Exercise) Fill-in MultOutput.java

You need to fill in two methods to complete this lab assignment. getInput() will create a matrix and fill in the values from the filename given in the parameter. multOutput will create an output file which you will write the resulting matrix into the file. Note that you can do the calculation at the same time as printing out the result to the console and the output file. You need to print out the resulting matrix's dimensions as the first thing (just like the format of the input files).

You will have 3 input combinations to test for this lab. Each output should be saved to a different file. They should be named Result1.txt, Result2.txt and Result3.txt.

Expected Output:

Put into → Result1.txt

```
Enter the First file name: InNum1.txt
Enter the Second file name: InNum2_1.txt
MATRIX MULTIPLY:
3     3
0     0     0
0     0     0
0     0     0
```

Put into → Result2.txt

```
Enter the First file name: InNum1.txt
Enter the Second file name: InNum2_2.txt
MATRIX MULTIPLY:
3     3
30    36    42
18    24    30
27    33    39
```

Put into → Result3.txt

```
Enter the First file name: InNum1.txt
Enter the Second file name: InNum2_3.txt
MATRIX MULTIPLY:
3     6
54    60    66    72    78    84
30    36    42    48    54    60
48    54    60    66    72    78
```

What to hand in

When you are done with this lab assignment, you are ready to submit your work. Make sure you have done the following ***before*** you press Submit:

- Include answers to questions (Q1-Q10)
- Attach filled in MultOutput.java
- Attach created Result1.txt, Result2.txt and Result3.txt
- List of Collaborators
