Lab 2 – Reflection and Results

Jason Gasparini
Jason.Gasparini@Marist.edu

October 4, 2023

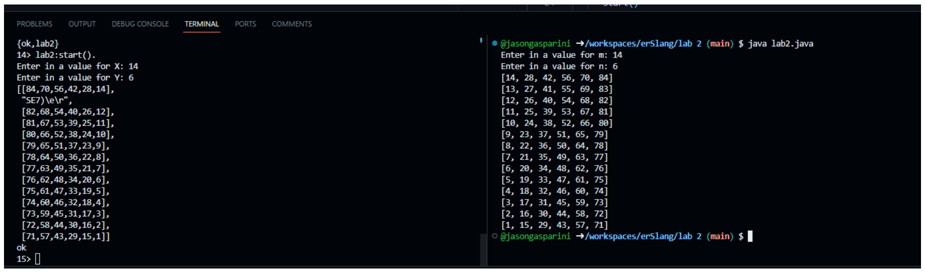
1 Reflection

1.1 Erlang vs. Java

With choosing to make use of the natural loops that are available to Java, coding this program between Erlang and Java required two different approaches. For a basic algorithm like this one, its complexity could be measured by how many total lines of code were needed. But, from my experience with this lab, that is not an accurate measurement of complexity. Because of the fact that I do not favor recursion over a basic for loop, the Erlang version was more complex yet it required many less lines to accomplish the same goal. The difficulty that I faced while coding the Erlang version can certainly be attributed to my inexperience with the syntax but also is because of the forced recursive nature. While this program in Erlang may be easier to code, I still think that it is harder to understand conceptually. The Erlang version of the code uses an

extra function for the input that I segmented away from the start() method for some clarity. The nature of the function is recursive so that if the input is not of the desired type it will continue to call itself so that the program does not have to be rerun. Once the input is gathered, start() calls a method listLoop() that recurses on itself to create the desired "X" number of lists. In each call of listLoop(), elementLoop() is another recursive function called to populate each list with "Y" number of elements. My Java implementation does not use any recursion. It uses two whiel loops for input with try-catch blocks in order to handle any bad inputs. After X and Y are defined, a for loop is initiated for each "X" number of lists and then begins an embedded for loop that will fill the lists with their elements. In order to print the list of lists, a for loop is needed again to print its contents. This is DIFFERENT than Erlang because within the Erlang program, I was able to print the list of lists with one line of code and no loop of recursion.

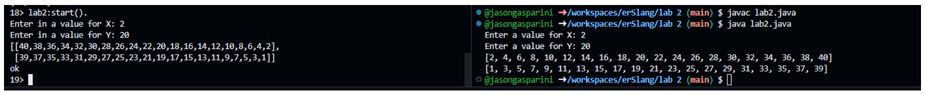
1.2 Results



```
{ok,lab2}
14> lab2:start().
Enter in a value for X: 14
Enter in a value for Y: 6
[[84,70,56,42,28,14],
 "SE7)\e\r",
 [82,68,54,40,26,12],
 [81,67,53,39,25,11],
 [80,66,52,38,24,10],
 [79,65,51,37,23,9],
 [78,64,50,36,22,8],
 [77,63,49,35,21,7],
 [76,62,48,34,20,6],
 [75,61,47,33,19,5],
 [74,60,46,32,18,4],
 [73,59,45,31,17,3],
 [72,58,44,30,16,2],
 [71,57,43,29,15,1]]
ok
15> 
```

```
@jasongasparini ➜/workspaces/erSlang/lab 2 (main) $ java lab2.java
Enter in a value for m: 14
Enter in a value for n: 6
[14, 28, 42, 56, 70, 84]
[13, 27, 41, 55, 69, 83]
[12, 26, 40, 54, 68, 82]
[11, 25, 39, 53, 67, 81]
[10, 24, 38, 52, 66, 80]
[9, 23, 37, 51, 65, 79]
[8, 22, 36, 50, 64, 78]
[7, 21, 35, 49, 63, 77]
[6, 20, 34, 48, 62, 76]
[5, 19, 33, 47, 61, 75]
[4, 18, 32, 46, 60, 74]
[3, 17, 31, 45, 59, 73]
[2, 16, 30, 44, 58, 72]
[1, 15, 29, 43, 57, 71]
@jasongasparini ➜/workspaces/erSlang/lab 2 (main) $ 
```

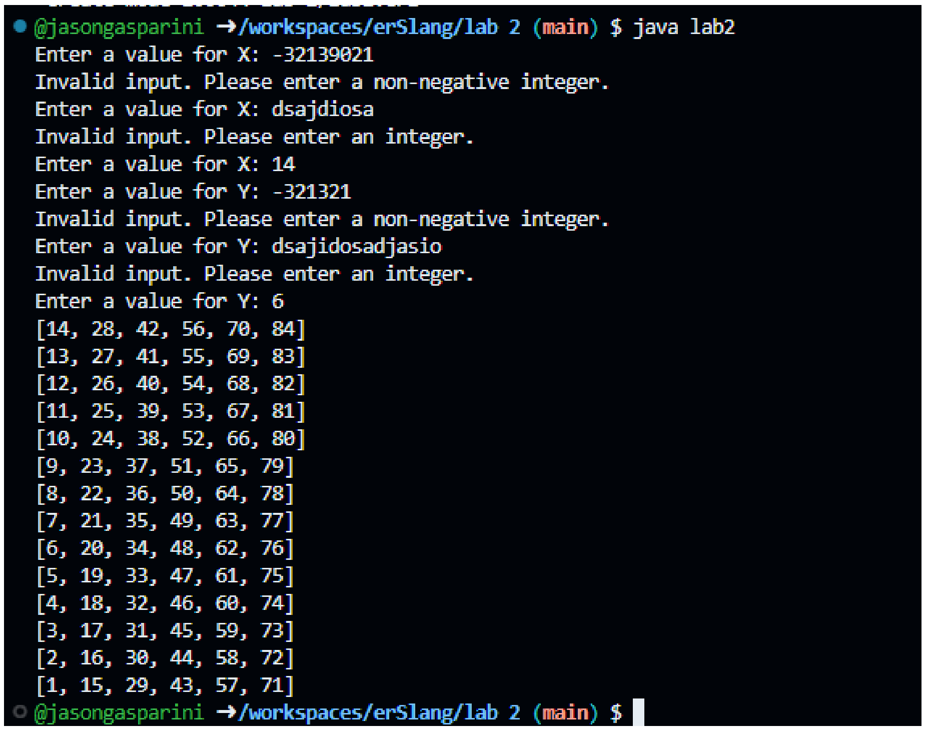
Figure 1: 1st successful run mirrored across



```
18> lab2:start().
Enter in a value for X: 2
Enter in a value for Y: 20
[[40,38,36,34,32,30,28,26,24,22,20,18,16,14,12,10,8,6,4,2],
 [39,37,35,33,31,29,27,25,23,21,19,17,15,13,11,9,7,5,3,1]]
ok
19> 
```

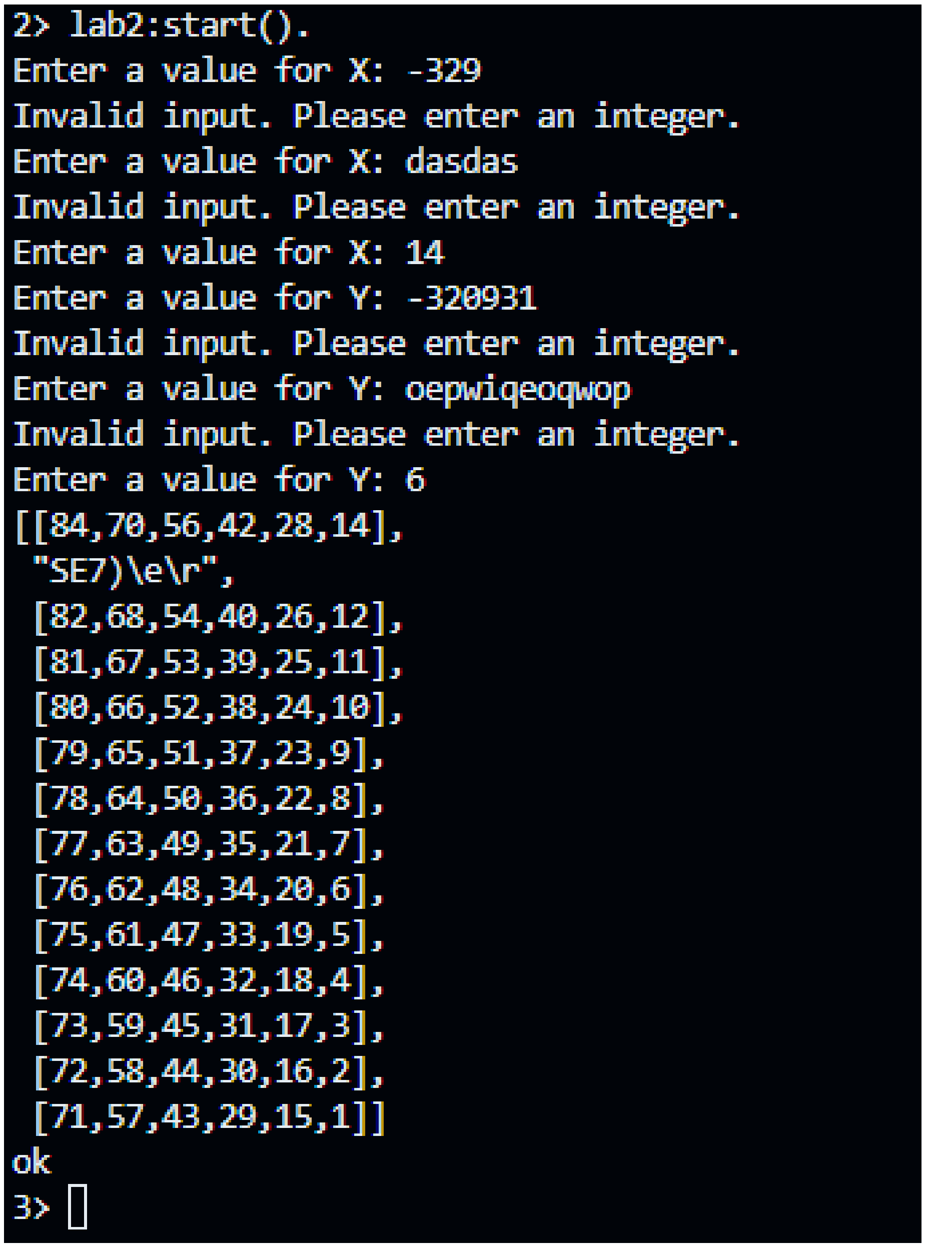
```
@jasongasparini ➜/workspaces/erSlang/lab 2 (main) $ javac lab2.java
@jasongasparini ➜/workspaces/erSlang/lab 2 (main) $ java lab2.java
Enter a value for X: 2
Enter a value for Y: 20
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40]
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39]
@jasongasparini ➜/workspaces/erSlang/lab 2 (main) $ 
```

Figure 2: 2nd successful run



```
● @jasongasparini ➜ /workspaces/erSlang/lab 2 (main) $ java lab2
  Enter a value for X: -32139021
  Invalid input. Please enter a non-negative integer.
  Enter a value for X: dsajdiosa
  Invalid input. Please enter an integer.
  Enter a value for X: 14
  Enter a value for Y: -321321
  Invalid input. Please enter a non-negative integer.
  Enter a value for Y: dsajidosadjasio
  Invalid input. Please enter an integer.
  Enter a value for Y: 6
  [14, 28, 42, 56, 70, 84]
  [13, 27, 41, 55, 69, 83]
  [12, 26, 40, 54, 68, 82]
  [11, 25, 39, 53, 67, 81]
  [10, 24, 38, 52, 66, 80]
  [9, 23, 37, 51, 65, 79]
  [8, 22, 36, 50, 64, 78]
  [7, 21, 35, 49, 63, 77]
  [6, 20, 34, 48, 62, 76]
  [5, 19, 33, 47, 61, 75]
  [4, 18, 32, 46, 60, 74]
  [3, 17, 31, 45, 59, 73]
  [2, 16, 30, 44, 58, 72]
  [1, 15, 29, 43, 57, 71]
○ @jasongasparini ➜ /workspaces/erSlang/lab 2 (main) $
```

Figure 3: Java run with error checking/unexpected inputs



```
2> lab2:start().
Enter a value for X: -329
Invalid input. Please enter an integer.
Enter a value for X: dasdas
Invalid input. Please enter an integer.
Enter a value for X: 14
Enter a value for Y: -320931
Invalid input. Please enter an integer.
Enter a value for Y: oepwiqeoqwop
Invalid input. Please enter an integer.
Enter a value for Y: 6
[[84,70,56,42,28,14],
 "SE7)\e\r",
 [82,68,54,40,26,12],
 [81,67,53,39,25,11],
 [80,66,52,38,24,10],
 [79,65,51,37,23,9],
 [78,64,50,36,22,8],
 [77,63,49,35,21,7],
 [76,62,48,34,20,6],
 [75,61,47,33,19,5],
 [74,60,46,32,18,4],
 [73,59,45,31,17,3],
 [72,58,44,30,16,2],
 [71,57,43,29,15,1]]
ok
3> 
```

Figure 4: Erlang run with error checking/unexpected inputs, List 13 has all printable ASCII characters