Programming Techniques - Assignment 1 - Question 2 - Report

List of Contents

My Links

- Link to Socials

Part 1

What is the Problem?

We need to make a code that will output the character `*` for several times. But we also have to decrement the number of `*` to print... Let me show you what I mean.

Suppose we have a variable `N` which will hard coded into the program code; `N` can be any **integer** number. Here is a little example if we set `N` to 5!

```
*****
****
***
**
*
```

This above ↑ will be the output.

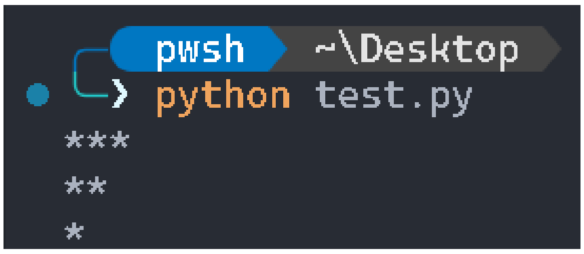
Description

We need to have 2 `for` loops. The first `for` loop will be responsible to print the number of lines (*row-wise*) and the second `for` loop will be responsible to output the number of `*` (*column-wise*).
We are also going to be changing the lines so that it does not print on a single line.

Python Code

```python
# DECLARE N: INTEGER
N = 3

# number of lines it will output ( row-wise )
# DECLARE x: INTEGER
for x in range(N):

    # number of '*' to output ( column-wise )
    # DECLARE y: INTEGER
    for y in range(N):
        # output the character on the same line
        # use `end=""` to allow '*' to stay on the same line
        print("*", end="", flush=True)

    # decrement N
    N -= 1
    # ouput a new line so that it whole "output" does not stay on 1 line
    print()
```

Output of Python Code

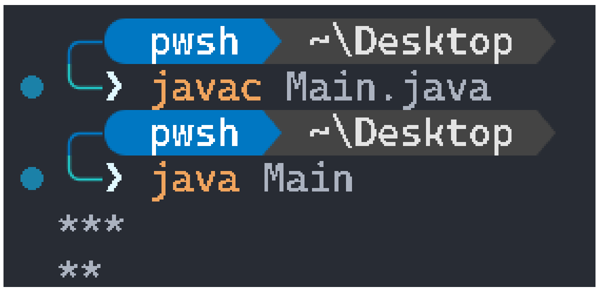


```
pwsh  ~\Desktop
❯ python test.py
***
**
*
```

Translating Python Code to Java

```java
public class Main {
    public static void main(String[] args) {

        // DECLARE N: INTEGER
        int N = 3;

        // number of lines it will output ( row-wise )
        // DECLARE x: INTEGER
        for(int x = 0; x < N; x++) {

            // number of '*' to output ( column-wise )
            // DECLARE y: INTEGER

            for(int y = 0; y < N; y++) {
                // output the character
                System.out.print("*");
            }

            // decrement N
            N -= 1;
            // output a new line; so that its not in the same line
            System.out.println();

        }
    }

}
```

Output of Translation



```
pwsh  ~\Desktop
❯ javac Main.java
pwsh  ~\Desktop
❯ java Main
***
**
```

Problems with Code Translation

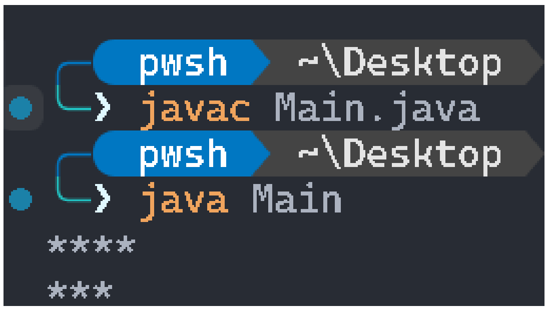
As we are decrementing `N` and we are using `N` for both `for` loops. This means that one line is being completely lost!

Test 1: Increase value of `N` without changing anything

Let's try changing the value of `N`. Even though we need to print:

```
N = 3
***
**
*
```

I am going to change the value of `N` to 4.



```
pwsh  ~\Desktop
> javac Main.java
pwsh  ~\Desktop
> java Main
****
***
```

We can see that this did not work!!!

Test 2: Change values in `for` loops

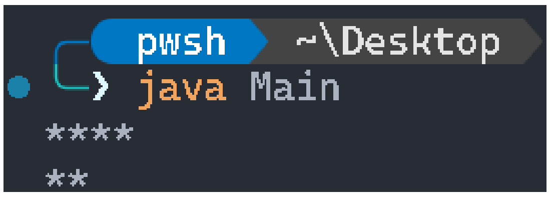
Instead of writing:

```
for(int x = 0; x < N; x++) {

    for(int y = 0; y < N; y++) {
        System.out.print("*");
```

I am going to be changing the initial value of `y` inside the loop

```
for(int x = 0; x < N; x++) {

    for(int y = x; y < N; y++) {
        System.out.print("*");
```

We can see that this is still not good!



```
pwsh  ~\Desktop
> java Main
****
**
```

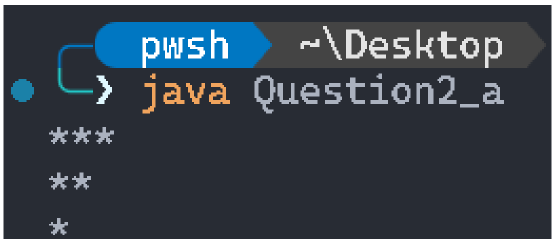
Solution - Part 1

This is because Java is similar to C when managing *memory*. What I mean is that; Java / C will need to have another variable that will store the number of lines (*row-wise*) to print and this variable should **not** change. That is, it needs to stay constant.

Corrected Java Code

```java
// Question 2 - Part (a)
// Author: S.Sunhaloo
// Date: 31 May @ 9:20

public class Question2_a {
    public static void main(String[] args) {

        // DECLARE ch_output: INTEGER
            // hold the number of character to print
        int ch_output = 3;
        // DECLARE N: INTEGER
            // hold the original amount of lines to print
        int N = ch_output;

        // number of lines to print / output ( row-wise )
        for(int i = 0; i < N; i++){

            // number of character to output ---> in single line ( column-wise )
            for(int j = 0; j < ch_output; j++){
                // output the character '*'
                System.out.print("*");
            }

            // decrement the number of character to print / output by 1
            ch_output -= 1;
            // add a println function to change the line
            // else will print in the same line as using print function
            System.out.println();
        }

    }
}
```

Output of Working Code



```
pwsh  ~\Desktop
> java Question2_a
***
**
*
```

Changing the Value of `N`

Warning

By changing the value of `N` in my code means to change the value of `ch_output`!

For fun, let's go ahead and change the value of `N` to 20!

Tip

If you did not know, we can use the command:

```
java Main > out.txt
```

This command will output the "*output*" of your program to the file `out.txt`

> If you just use the *Play* button in NetBeans; you will not be able to know this

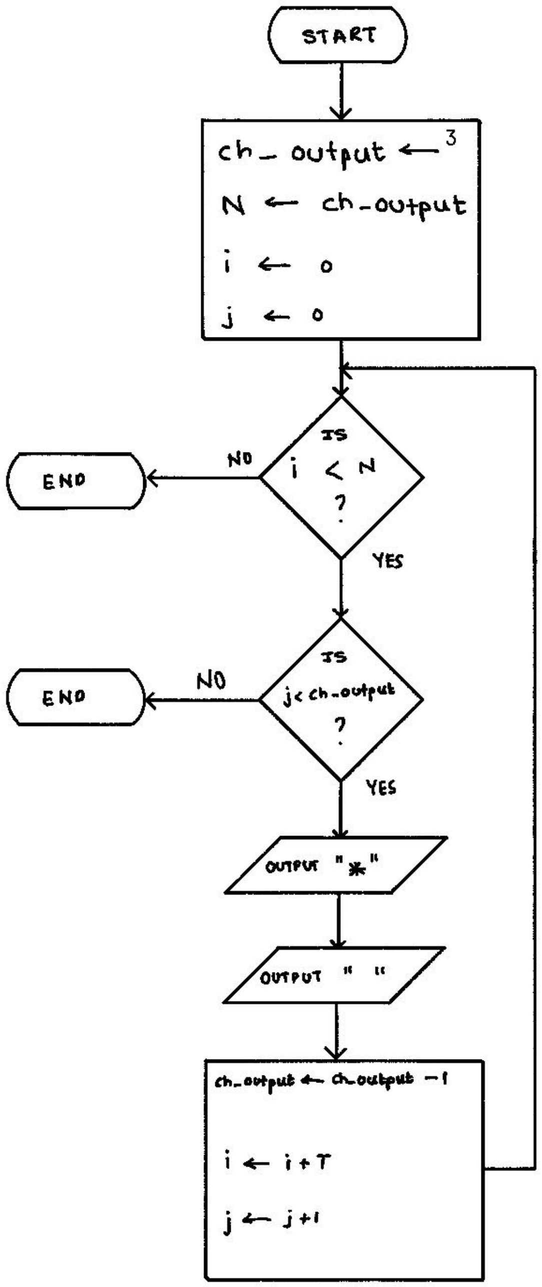
```
********************
*******************
******************
*****************
****************
***************
**************
*************
************
***********
**********
*********
********
*******
******
*****
****
***
**
*
```

Algorithm for Java Code - Part 1

> I have written in like we did back in HSC

```
DECLARE ch_output: INTEGER
DECLARE N: INTEGER
DECLARE i: INTEGER
DECLARE j: INTEGER

ch_output <--- 3
N <--- ch_output

FOR i <--- 0 TO N - 1 DO

    FOR j <--- 0 TO ch_output - 1 DO
        OUTPUT "*"
    ENDFOR

    ch_output <--- ch_output  - 1
    OUTPUT NEWLINE

ENDFOR
```

Flowchart for Java Code - Part 1



Part 2

What is the Problem Now?

We are now going to alter the program so that the output of our Java Code becomes:

```
***
 **
  *
```

Description

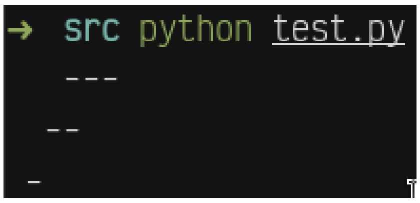
We are not going to add another `for` loop in the *range of* `i` so that we can output the correct number of `<Space>` and also output the character `*` in the correct position.

Lets start by writing a Python Code

Python Code

```python
# DECLARE N: INTEGER
N = 3

# number of lines it will output ( row-wise )
# DECLARE x: INTEGER
for x in range(N):

    # add the space
    # DECLARE i: INTEGER
    for i in range(N):
        # output a blank space
        print(" ", end="", flush=True)

    # number of '*' to output ( column-wise )
    # DECLARE y: INTEGER
    for y in range(N):
        # output the character on the same line
        # use `end=""` to allow '*' to stay on the same line
        print("-", end="", flush=True)

    # decrement N
    N -= 1
    # ouput a new line so that it whole "output" does not stay on 1 line
    print()
```

Output of Python Code

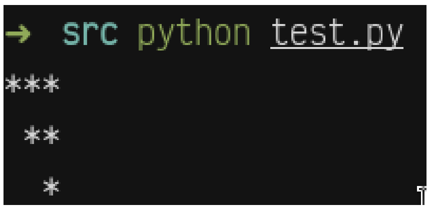


```
➜  src python test.py
   ---
  --
 -
```

Second Iteration of Python Code

```python
# DECLARE N: INTEGER
N = 3

# number of lines it will output ( row-wise )
# DECLARE x: INTEGER
for x in range(N):
    # add the space
    # DECLARE i: INTEGER
    for i in range(x): # output in the range of first outer loop
        # output a blank space
        print(" ", end="", flush=True)

    # number of '*' to output ( column-wise )
    # DECLARE y: INTEGER
    for y in range(N):
        # output the character on the same line
        # use `end=""` to allow '*' to stay on the same line
        print("-", end="", flush=True)

    # decrement N
    N -= 1
    # ouput a new line so that it whole "output" does not stay on 1 line
    print()
```

Output of Good Python Code (Second Iteration)



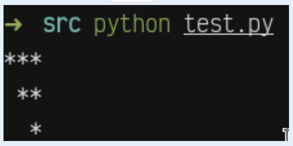
```
➜  src python test.py
***
 **
  *
```

Note

Why did we need to use "*in range of* `x`" to print the space?

This is because if we print in the *range of* `N`, we are going to be printing the `<Space>` before the `*` character.

- Consider it as printing the character `<Space>` like we are doing with the output of the character `*`
  Hence, we need to output in the *range of* `x`. This will follow the outer `for` loop and then output:



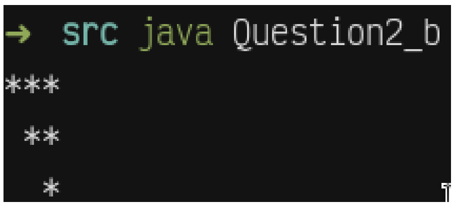
```
➜  src python test.py
***
 **
  *
```

Solution - Part 2

Corrected Java Code

```
// Question 2 - Part (b)
// Author: S.Sunhaloo
// Date: 31 May @ 9:40

public class Question2_b {
    public static void main(String[] args) {

        // DECLARE ch_output: INTEGER
            // hold the number of character to print
        int ch_output = 3;
        // DECLARE N: INTEGER
            // hold the original amount of lines to print
        int N = ch_output;

        // number of lines to print / output
        for(int i = 0; i < N; i ++){

            // move it to the other side / output <Space>
            for(int x = 0; x < i; x++){
                // as we are using print function, the <Space> character will be on the same li
                /*
                    in addition, we are using the same amount / same number of iteration / loop
                    ⟹ same number of <Space> output the same as `ch_output`
                */
                System.out.print(" ");
            }

            // number of character to output ---> in single line
            for(int j = 0; j < ch_output; j++){
                // output the character '*'
                System.out.print("*");
            }

            // decrement the number of character to print / output by 1
            ch_output -= 1;
            // add a println function to change the line ---> else will print in the same line
            System.out.println();
        }

    }
}
```

Output of Working Code



```
→ src java Question2_b
***
 **
  *
```

Changing the Value of `N`

> ⚠ Warning
>
> Again! By changing the value of `N` in my code means to change the value of `ch_output`!

Let's go ahead and change `N` to 20!
Hence, the output will be ↓:

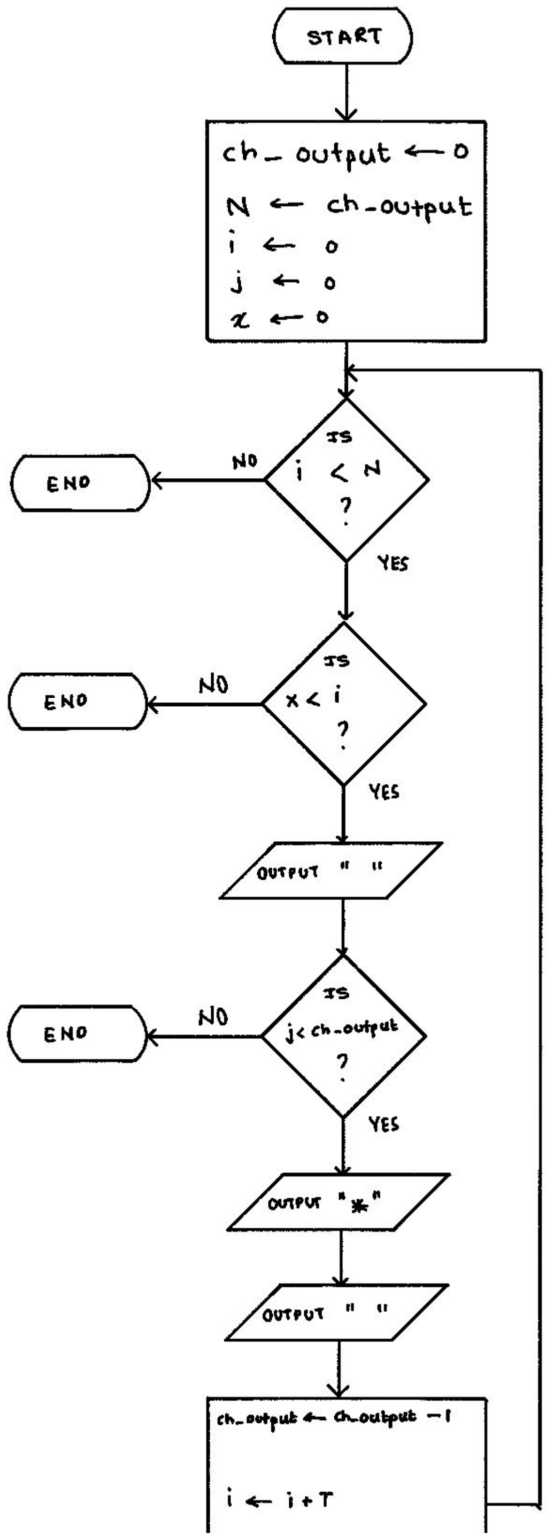
```
********************
 *******************
  ******************
   *****************
    ****************
     ***************
      **************
       *************
        ************
         ***********
          **********
           *********
            ********
             *******
              ******
               *****
                ****
                 ***
                  **
                   *
```

Algorithm for Java Code - Part 2

> I have written in like we did back in HSC

```
DECLARE ch_output: INTEGER
DECLARE N: INTEGER
DECLARE i: INTEGER
DECLARE j: INTEGER
DECLARE x: INTEGER

ch_output <--- 3
N <--- ch_output

FOR i <--- 0 TO N - 1 DO

    FOR x <--- 0 TO i DO
        OUTPUT " "
    ENDFOR
    FOR j <--- 0 TO ch_output - 1 DO
        OUTPUT "*"
    ENDFOR


    ch_output <--- ch_output  - 1
    OUTPUT NEWLINE

ENDFOR
```

Flowchart for Java Code - Part 2



Learning Curve

I learnt that **Python** is ≠ **Java**... Obviously.
The way the Python handles memory is extremely different from Java's (*or even C's*) way of handling memory.

Here is a little example I can give you while I was learning C, after already learning Python

In Python, we do not need to declare our variables. Hence, we can create an Array and then ask the user for the size of the array.
When I tried to do that in C... `ERROR`, because:

- We need to declare our variables / arrays
- We need already have the size of the array

Then, when it comes time to *create* our array... We can pass in the size into the array like so:

```c
#include <stdio.h>

int main() {

  // DECLARE array_size: INTEGER
  int array_size;
  // ask the user to enter the array size
  printf("\nPlease Enter the Size of Array: ");
  scanf("%d", &array_size);

  // create an array of size "array_size"
  // DECLARE ARRAY arr: INTEGER
  int arr[array_size];

  printf("\n");

  // ask the user to enter data into array
  for (int i = 0; i < array_size; i++) {
    printf("Please Enter A Value: ");
    scanf("%d", &arr[i]);
  }

  printf("\n");

  // display the array
  // could have also used `sizeof()` function ---> for display
  for (int i = 0; i < array_size; i++) {
    printf("Index: %d | Value = %d\n", i, arr[i]);
  }

  return 0;
}
```

and **NOT** this:

```c
#include <stdio.h>

int main() {

  // create an array of size "array_size"
  // DECLARE ARRAY arr: INTEGER
  int arr[array_size];

  // DECLARE array_size: INTEGER
  int array_size;
  // ask the user to enter the array size
  printf("\nPlease Enter the Size of Array: ");
  scanf("%d", &array_size);

  printf("\n");

  // ask the user to enter data into array
  for (int i = 0; i < array_size; i++) {
    printf("Please Enter A Value: ");
    scanf("%d", &arr[i]);
  }

  printf("\n");

  // display the array
  // could have also used `sizeof()` function ---> for display
  for (int i = 0; i < array_size; i++) {
    printf("Index: %d | Value = %d\n", i, arr[i]);
  }

  return 0;
}
```

Info

This is because the computer at this point in time does not know what `array_size` is because it has **not** been declared

---

Conclusion

Hence, we could say that we have successfully implemented the program to output the `*` characters the right way!

We are able to print:

```
***
**
*
```

```
***
 **
  *
```

In addition, we can change the value of `N`; in our case, we need to change the value of `ch_output`

Appraisal

I have learn about the "*memory allocation*" in Java compared to Python which is primarily used because of its ease of use.
In addition, learned about how to create space so that the `*` makes you think that it starts from Right to Left.
I personally think that this question / program is short, simple and sweet!

References

- My own Python Code

Socials

- Instagram: https://www.instagram.com/s.sunhaloo/
- YouTube: https://www.youtube.com/channel/UCMkQZsuW6eHMhdUObLPSpwg
- GitHub: https://www.github.com/Sunhaloo

S.Sunhaloo
Thank You!