Why am I writing this?

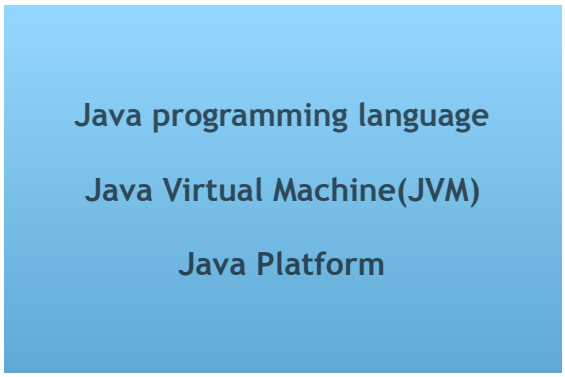
To be honest, when I think of Java, I always picture a very hard-to-learn language. That is possibly because people who write programs with Java say that 'it is a pain in the ass'. But I would like to test this statement myself. Simply because I do not like the idea that there is something 'I have never tried to learn' and 'I am scared to learn.' That is why I grabbed that book with 1224 pages and decided to take notes as I read page to page, chapter to chapter. So that if there exists someone like me with this irrational fear of JAVA, maybe we all can attempt to overcome it.

I expect my readers to have at least some beginner experience in other languages and familiarity with the concepts like classes, types(int, float, double), variables etc. If you don't, no worries! Whenever you see something that I don't explain in detail, that means it is time to create yet another tab in your browser window, and start searching.

Relax and enjoy the ride!

Chapter 1 - Introduction

What is java?



-> the language in which applications, applets and components are written.
-> portable machine language of a CPU architecture.

-> predefined set of Java classes that exist on every Java installation. (Java runtime environment)

Things you should know about Java:

1. **Write Once, Run Everywhere**

You have to write your application only once and you will be able to run it anywhere.

2. Security

The Java platform allow users to download untrusted code and run it on a secure environment. That is, untrusted code cannot infect the host system with a virus, cannot access the hard drive for reading and writing etc.

3. 'The network is the computer.'

4. Dynamic and Extensible

It is organized units called Classes. Classes are stored in files and loaded when needed. Which means that the program can dynamically extend itself by loading the classes it needs.

An example program:

```
public class Factorial {

     public static void main(String[] args) {
          int input = Integer.parseInt(args[0]);
          double result = factorial(input);
          System.out.println(result);
     }

     public static double factorial(int x) {
          if (x < 0)
               return 0.0;
          double fact = 1.0;
          while(x > 1) {
               fact = fact * x;
               x = x - 1;
          }
          return fact;
          }
     }
```

Analyzing the Program:

```
     public class Factorial {
```

We are defining a class named Factorial. The name of the class is also the name of the program. The file name indicates that the file contains a class named Factorial.

public: modifier. It means that the class is publicly available and can be used by anyone.

-> All Java programs are classes and you can use many classes instead of just one, depending on your needs.

-> Each class defines a unique object.

-> The Factorial class have two members, both of which are methods.

What is a method?

Don't panic. If you have programmed in other languages, you have probably coded methods before. However, you may have been calling them **functions, procedures or subroutines**.

A java program can call-or invoke- a method to **execute** the code in it.

Methods have parameters and return values.

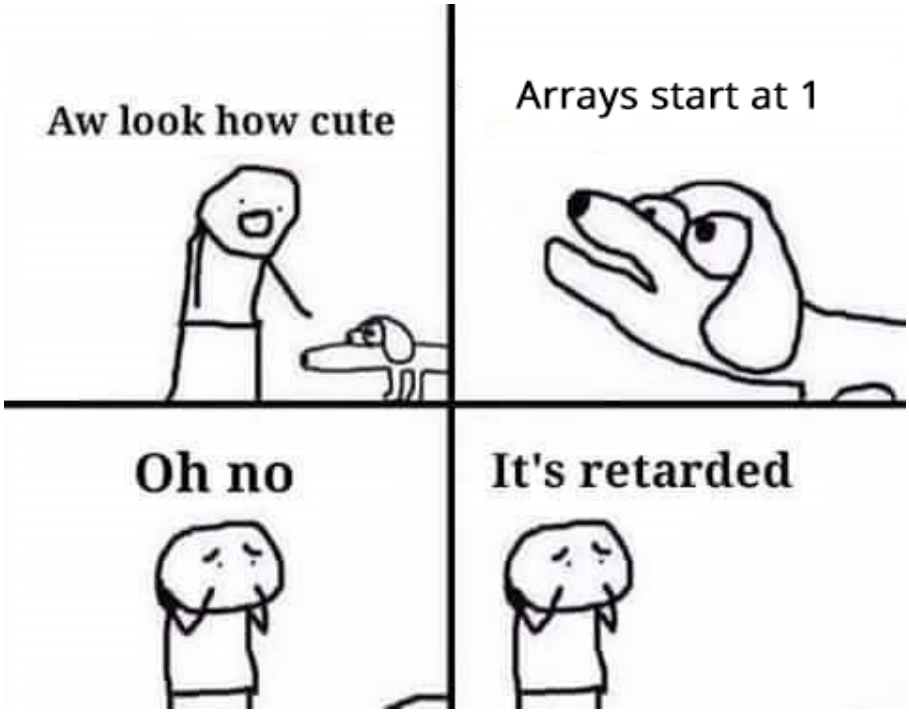
public	static	void	main	(String[]	args) {
Modifier	Motifier	Return type	Name of the method	Type of the parameter	Name of the parameter

public : publicly accessible.

void : The return type. In this case, it means that this method has '**no return value**'.

main : is a special name. Java interpreter reads the class you specify and looks for a method named **main**. Interpreter starts running the program at that method. When main() method finishes, the program is done. The method must be declared **public static void** exactly as shown above. Considering we will need to write this line in all of our Java programs, better write it down somewhere in your brain.

String[] : type of the parameter named **args**, which is an array of strings. Remember that arrays start at 0!



Note:
Java is a **strongly typed** language. Which means that you must declare the type of the variable and you can only refer to values of that type.

int input = Integer.parseInt(args[0]);

The input variable has the type int; therefore, it cannot refer to a floating-point number or a string.

The value **assigned** to our input variable is **Integer.parseInt(args[0]);** .
This is a method invocation.
This method 'parses an integer', that is, it converts a string representation of an integer to the integer itself. And as the argument, args[0] is given. We care only about the first integer in the args array, so we use **0** -zero- to refer to that string.

System.out.println(result);

This method causes Java interpreter to print out a value. In this case, the value of the result variable.

public static double factorial(int x) {

Unlike main() method, which had no return value(void), factorial method returns a value of type **double.**

```
if (x < 0)
  return 0.0;
```

Utilizing the if statement, we provide a control flow. We make sure that the value if x is not a negative number. If it is, then we return 0.0.

```
double fact = 1.0;
```

This variable with type double holds the value of the factorial as we compute it in the statements that follow.

```
while(x > 1) {
    fact = fact * x;
    x = x - 1;
```

Before the loop starts:
fact is 1.0 and x is 4. After one iteration:
fact is 4.0 and x is 3. After second iteration:
fact is 12.0 and x is 2. After third iteration:
fact is 24.0 and x is 1. The loop stops because x is no longer > 1.

We return the value of the fact and en fin.