Lecture Notes

INTRODUCTION

So, we started with Computational thinking. First, we introduced it to you as an approach or a thought process which all of us can use to solve not only coding problems but our day to day problems as well.

Decomposition

We then moved to first aspect of Computational thinking which is **Decomposition.** There we learnt how breaking down a problem to smaller easy to solve subproblems helps in reducing the complexity of the problem and we get a head start as to how to attack the problem. Like we discussed the example where the task was to set up an office. There we broke down the problem to the questions such as:

- What is the carpet are required for the office?
- How many people will be working in the office?
- How many of them will need computers?
- How much air conditioning is required for the office?

This is not exhaustive list but surely gave you an idea on how breaking down a problem to smaller ones helps a lot in solving them.

Pattern Recognition

Then we moved to the next aspect of Computational thinking which is **Pattern Recognition.** With this you realized how identifying similarities and differences among the subproblems helps us in identifying the information we need to work on. Also identifying the pattern was identified as a key step in reducing the time required to solve the problem. In this case the fingerprint sensors example was discussed. How fingerprint sensors work on patterns in various aspects of your fingerprints.

Abstraction

Next came **Abstraction.** Here we learnt that ignoring the information which is less relevant or irrelevant greatly reduces the complexity of the problem and make the problem at hand easier to solve. In this case we discussed an example of finding out the smallest distance between 2 cities. In this case the weather condition of the cities and the condition of roads between the cities was not important. In programming as well, we use functions which perform certain tasks. How they perform the task is not important to us. We just use them for their functionality.

Algorithm Design

Then we learnt the last and very important aspect of Computational thinking which is **Algorithm Design.** Here we discussed that once we know how to solve a problem, it's very important to write the solution in a step by step manner. These steps should be very clear and finite. Here we used the password

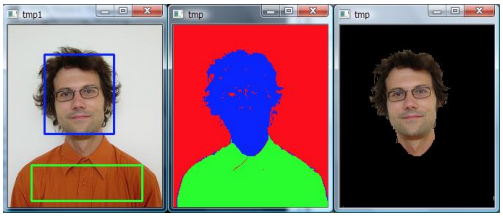
example to illustrate how it is written. The sequence of steps converted a website name to a unique password. Recipes as we discussed are great examples of algorithms as well.

All 4 steps in action

We then discussed a very important and interesting example. The problem was this
How does the camera app recognize if there is a human face in the camera frame? We discussed that the decision that the camera uses to figure out if there is a human face in the frame is actually really similar to the computational thinking process.

Decomposition

The first thing that the camera app does is decomposing the picture frame into segments of smaller things. For example, if the camera app sees a picture like this



It may decompose the picture into three small segments - that face of the man, the blue segment. The shirt that the man is wearing, the green segment. And the white background in the image, the red segment.

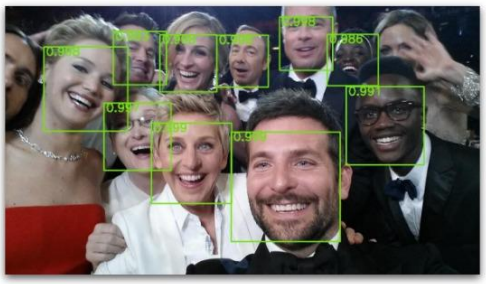
To identify which of the three segments is a face, which one is a shirt and so on abstraction and pattern recognition were found useful.

Abstraction

At first, this program is probably not very smart, and it may think that nothing is a human face, because it has never seen human faces before. Therefore, the smart computer scientists at Apple probably use abstraction to extract the most relevant characteristics of a human face and showed the human face recognition program those characteristics.

Pattern Recognition

But there are seven billion people in the world and seeing one face will not be enough for the program to learn the characteristics of human faces of different ethnicities and genders across the world. Therefore, as discussed, the computer scientists probably show the human face recognition program tens of thousands or maybe even hundreds of thousands of faces from the world so the program can use pattern recognition efficiently to identify the common characteristics, not just for one person but for every human ethnicity and gender in the world.



Algorithm Design

Once the human face recognition program has used abstraction and pattern recognition to figure out the common characteristics of human faces, it is then able to use algorithm design and create a set of rules that can detect whenever a human face pops up in the picture frame of your smartphone's-camera app. And that said these are the four main steps. The same four steps as computational thinking that your camera app uses to find human faces in its picture frame.

Java Development Kit

Then we moved on to learn how does a computer understands a code written in a Programming language and what tools does it need to do so. That prompted us to download and learn about **Java Development Kit (JDK).** We learnt that JDK contains **Java Runtime Environment** and **Java Compiler**. We discussed that Java Compiler converts the code written in Java to an intermediate byte code that can be understood by the Java Virtual Machine, which will then convert the byte to machine language. This machine language is the only thing that your computer understands everything in. Java Virtual Machine is contained by Java Runtime Environment. We downloaded version 8 of JDK from the Oracle website and installed it. So JDK as discussed needs to be there on your computer for any Java code to run on your system.

Command Line

Then came **Command Line**. There we learnt how do we execute and compile the code on command line. We learnt this by first knowing what is command line. How do we open it? (By searching for Command Prompt in Start Menu). We learnt about cd command which helped us navigating between directories. cd<space><name of the directory> was the command we used for this.

The next important thing we learnt here was how to open a file using command line. We just wrote the file name with its extension and hit enter. The file opened in the default program. But your current location should be the same directory which contains the required file.

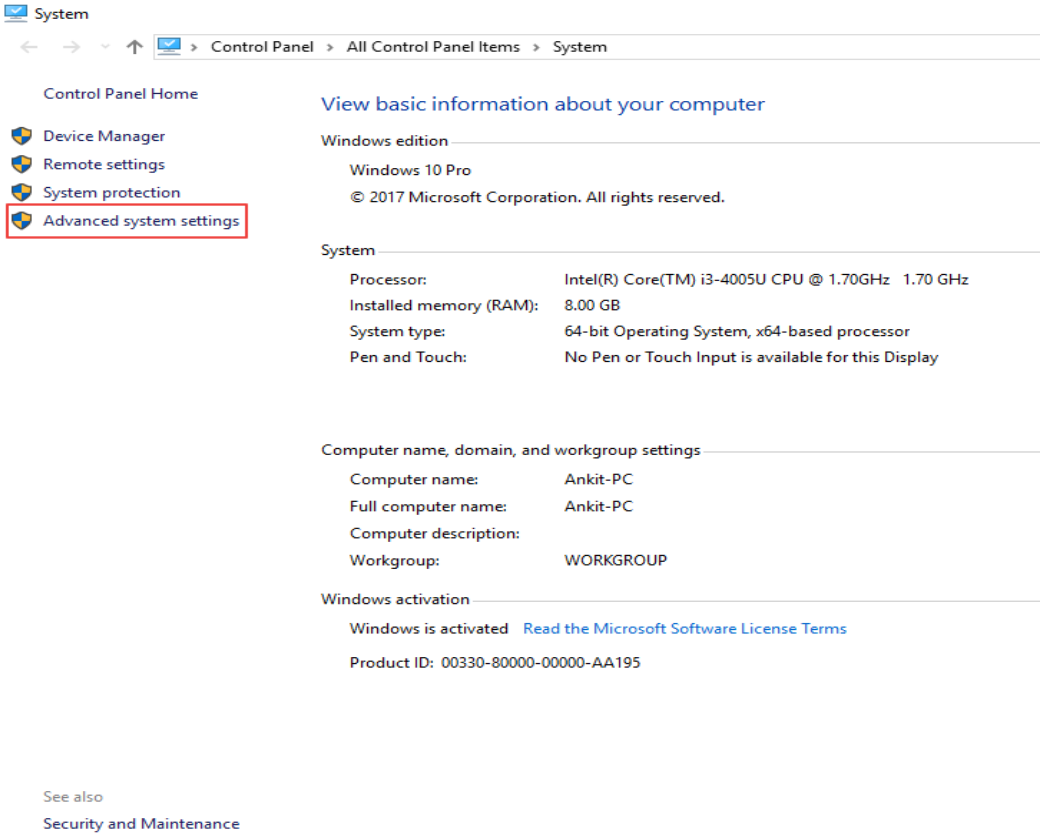
And if the file you were trying to open was not in your current location directory we added the file's location to the path variable so that command line could access it without changing your current location. You were required to restart your command line for this action to be effective.

All this is required to be done for Windows only and not for Mac

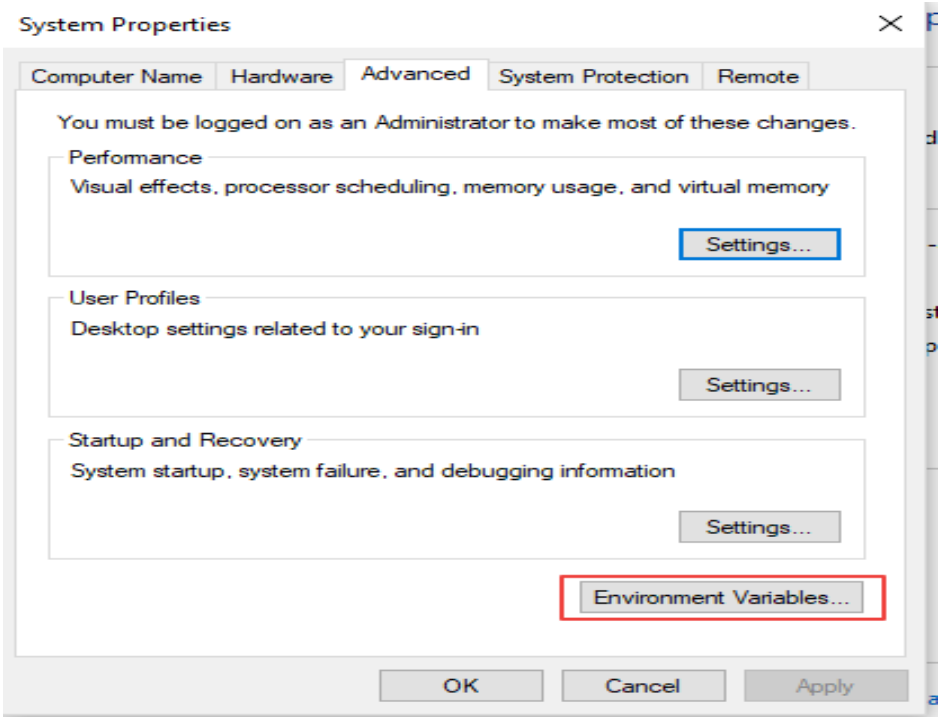
We then moved to writing Hello World code. We wrote the code on a text editor (Notepad in the case of windows)

There you were introduced to **System.out.println()** command which prints anything written inside the parentheses with in double quotes. We learnt how Notepad accessed Java Compiler through Command Line **(by writing javac<space>filename.java).** And once the code was compiled we ran it on Command Line itself **(by writing java<space>filename).** Before this we added Java Compiler to the path variable so that we don't need to navigate to its location while using it. We did that with the help of following steps.

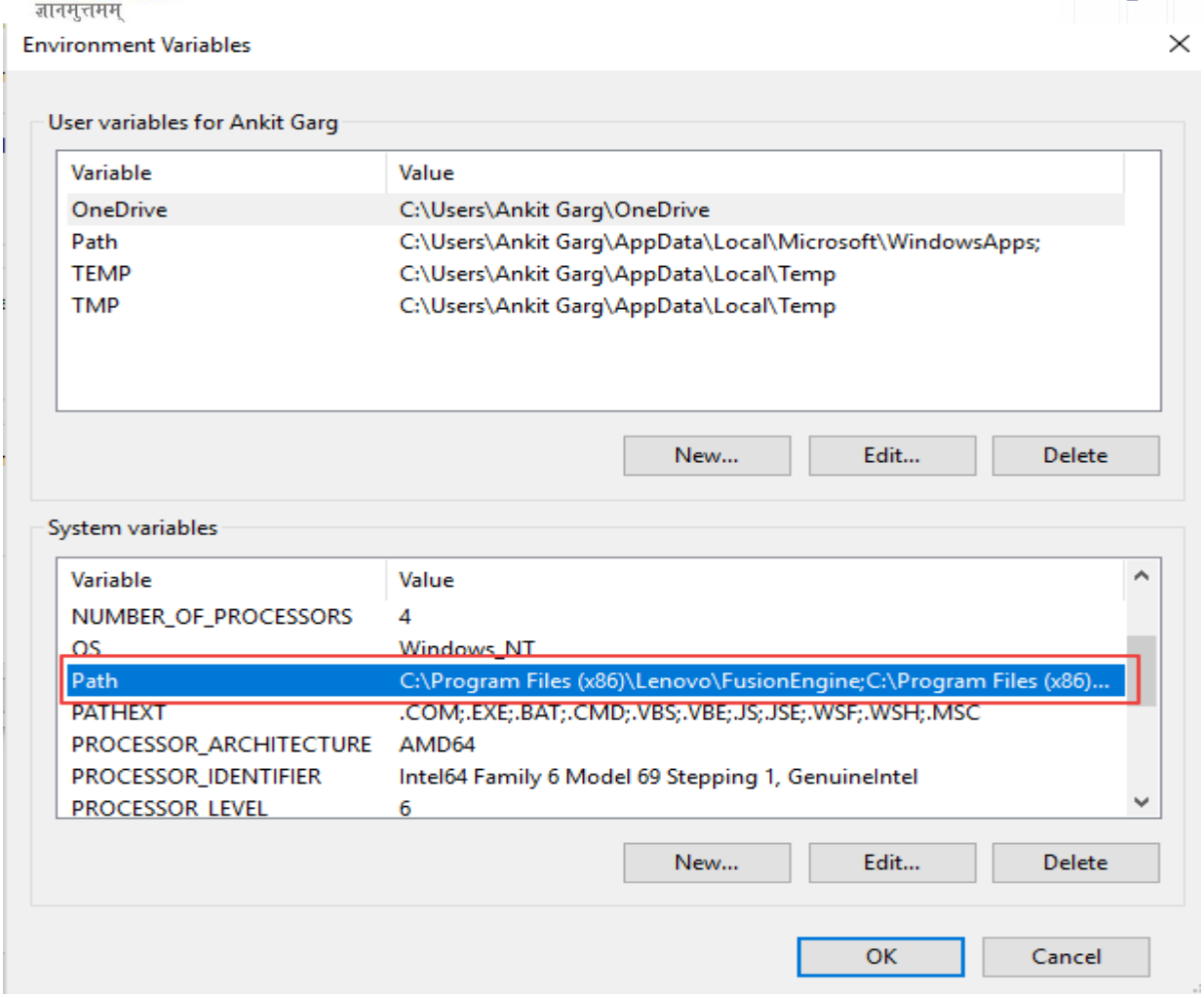
1. Go to Advanced Settings in System Properties.



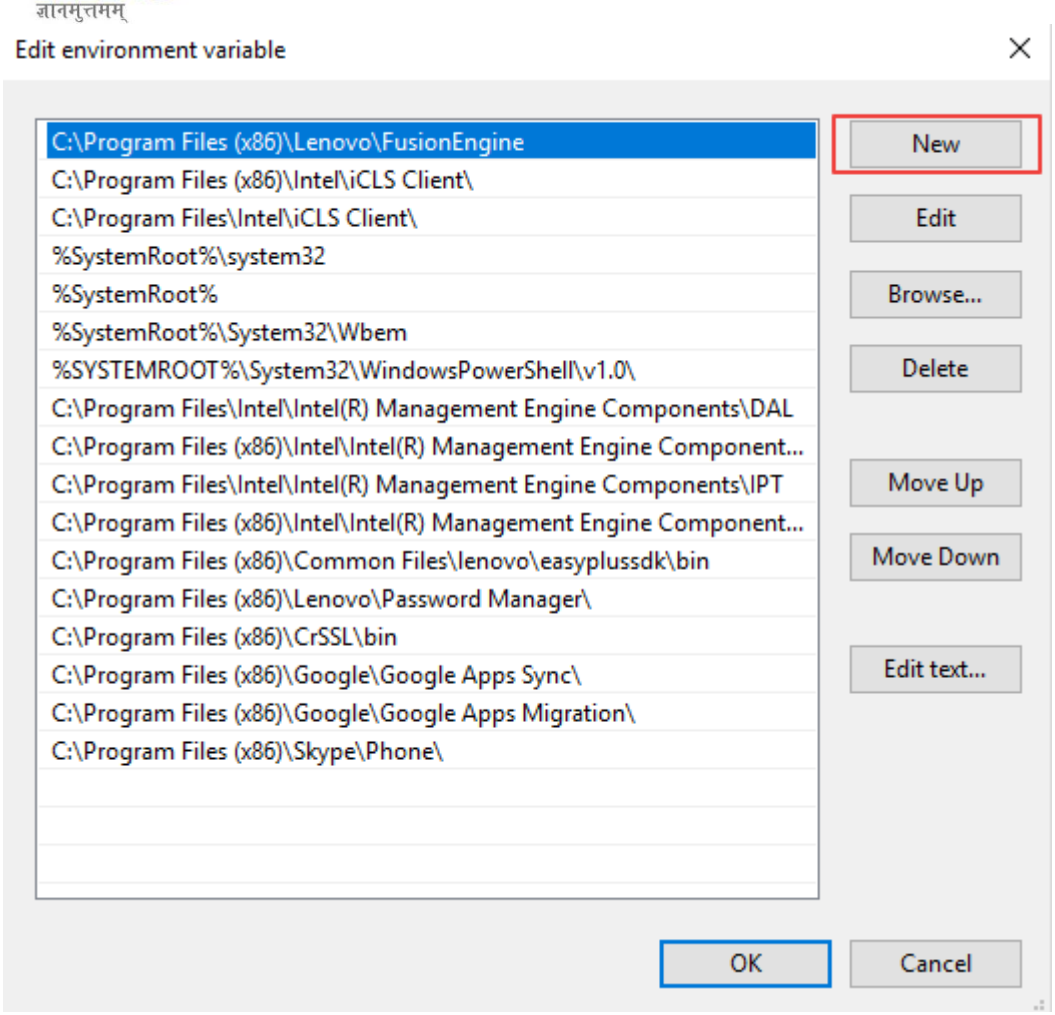
2. Click on Environment Variables



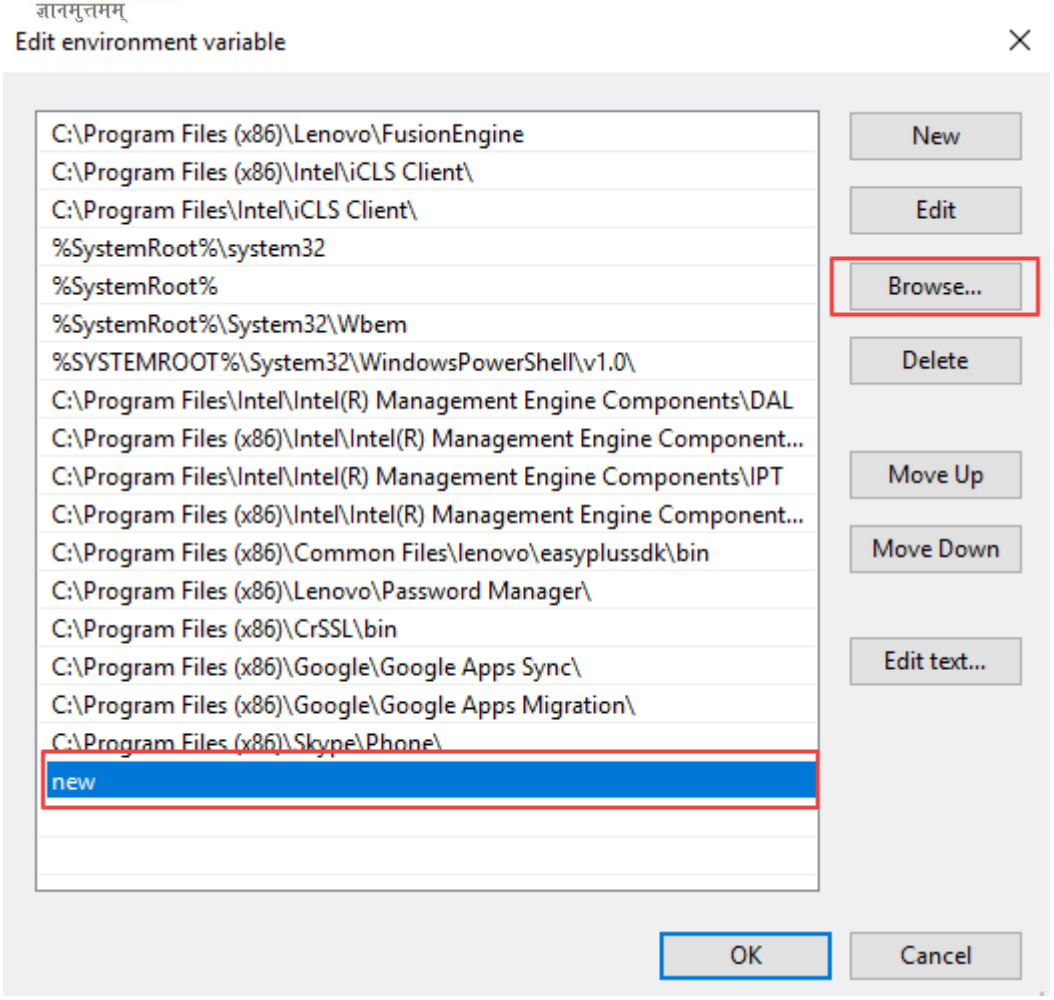
3. Click on Edit after selecting the Path or double click on Path

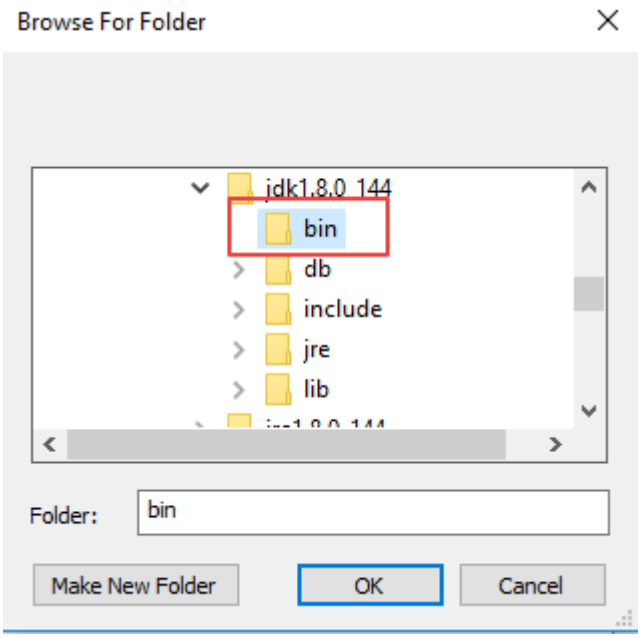


4. In the Dialog Box that appears, click on New

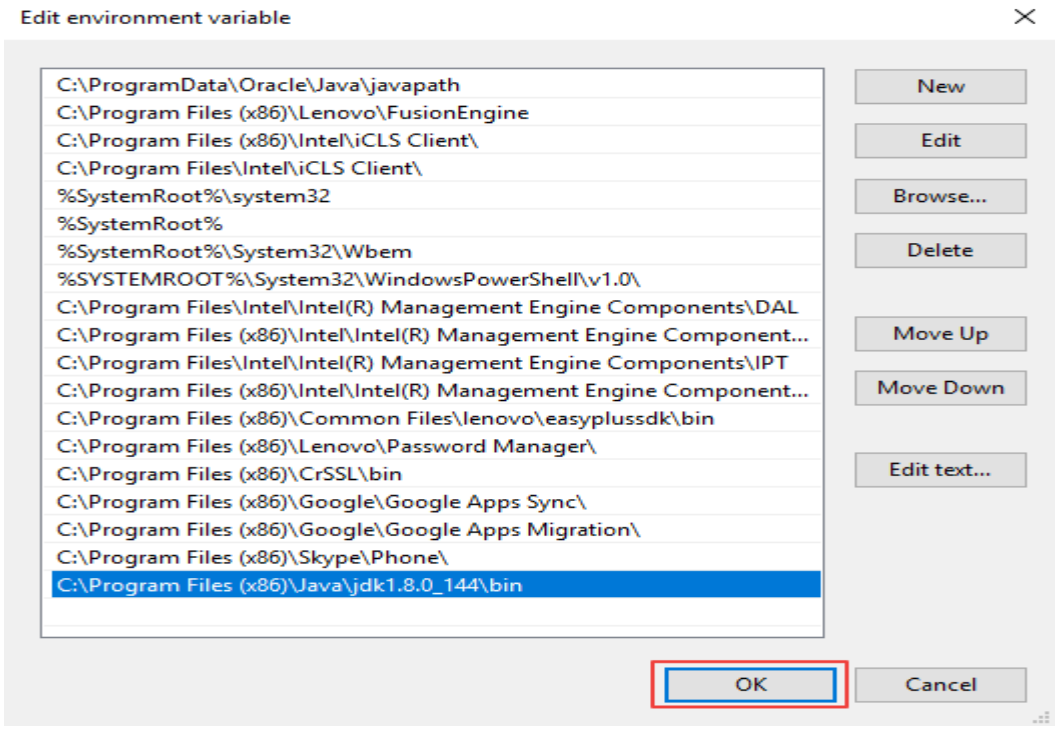


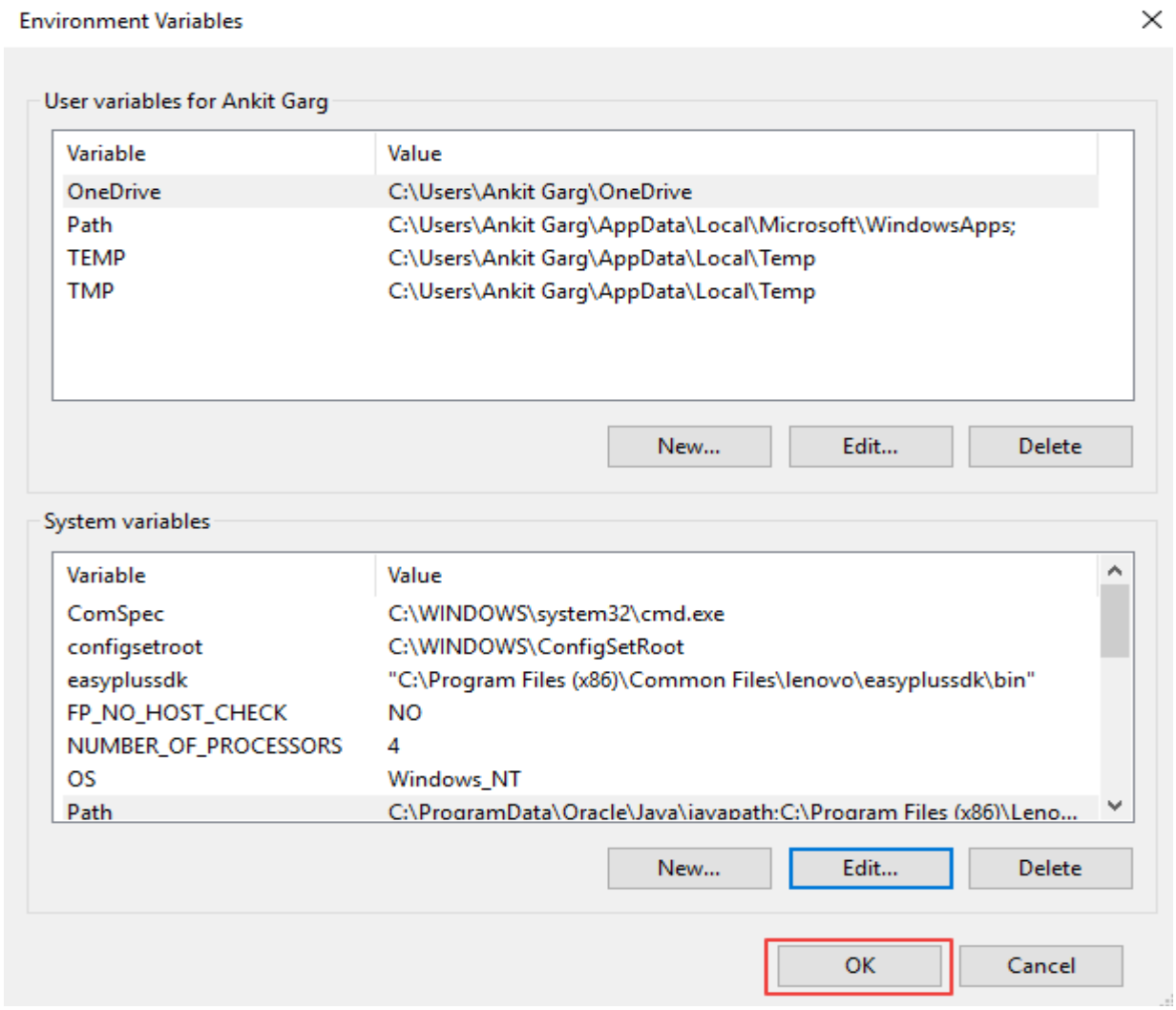
5. And finally Browse to C/Programs(x86)/Java/JDK/bin.





6. Click OK on the current Window and the previous Window as well.


Edit environment variable
C:\ProgramData\Oracle\Java\javapath
C:\Program Files (x86)\Lenovo\FusionEngine
C:\Program Files (x86)\Intel\iCLS Client\
C:\Program Files\Intel\iCLS Client\
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
C:\Program Files\Intel\Intel(R) Management Engine Components\DAL
C:\Program Files (x86)\Intel\Intel(R) Management Engine Component...
C:\Program Files\Intel\Intel(R) Management Engine Components\IPT
C:\Program Files (x86)\Intel\Intel(R) Management Engine Component...
C:\Program Files (x86)\Common Files\lenovo\easyplussdk\bin
C:\Program Files (x86)\Lenovo\Password Manager\
C:\Program Files (x86)\CrSSL\bin
C:\Program Files (x86)\Google\Google Apps Sync\
C:\Program Files (x86)\Google\Google Apps Migration\
C:\Program Files (x86)\Skype\Phone\
C:\Program Files (x86)\Java\jdk1.8.0_144\bin
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel


Environment Variables
User variables for Ankit Garg
Variable
Value
OneDrive
C:\Users\Ankit Garg\OneDrive
Path
C:\Users\Ankit Garg\AppData\Local\Microsoft\WindowsApps;
TEMP
C:\Users\Ankit Garg\AppData\Local\Temp
TMP
C:\Users\Ankit Garg\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
ComSpec
C:\WINDOWS\system32\cmd.exe
configsetroot
C:\WINDOWS\ConfigSetRoot
easyplussdk
"C:\Program Files (x86)\Common Files\lenovo\easyplussdk\bin"
FP_NO_HOST_CHECK
NO
NUMBER_OF_PROCESSORS
4
OS
Windows_NT
Path
New...
Edit...
Delete
OK
Cancel

7. Restart your command line and you are done.

IDE

But then when we changed the code on Notepad we had to recompile it on command line for the changes to be visible in the output. Also, only once we ran the code we got to know about the errors. This prompted us to use an Integrated Development Environment called Intellij, which will compile and execute the code for us without needing us to use the command line. Therefore, making our lives as programmers a bit easier.

We downloaded and installed the Ultimate version of Intellij. There we learnt how to create projects on Intellij. We then wrote the same Hello World Program on Intellij.

Also, we learnt that codes written on an IDE need not be compiled separately. We just ran it and it was compiled then and there only by the IDE itself. Any changes we made were visible in the output every time we ran the code. Also, Intellij proved very useful in highlighting the errors we made while writing code.