Module 2: Python Programming Fundamentals

Premanand S

Assistant Professor,
School of Electronics and Engineering,
Vellore Institute of Technology, Chennai

premanand.s@vit.ac.in

August 6, 2024

Programming

> *Any fool can write code that a computer can understand. Good programmers write code that humans can understand.*

[Martin Fowler]

Topics to be covered in Module 2,

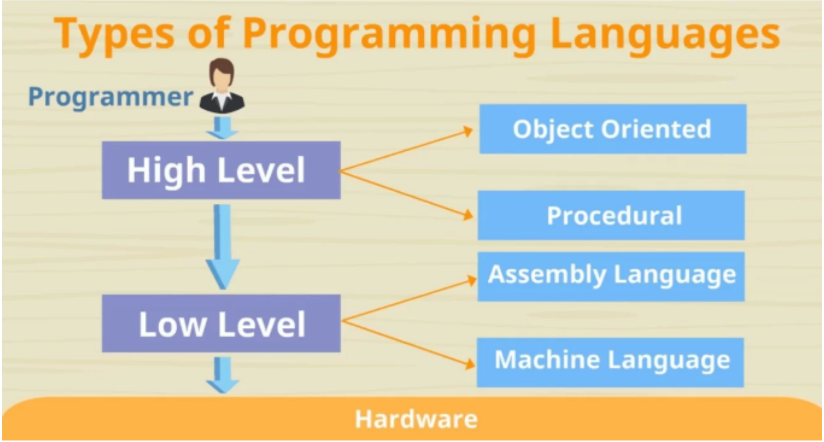
- Introduction to Python
- Interactive and Script mode
- Indentation
- Comments
- Variables
- Reserved words
- Data Types
- Operators and Precedence
- Expressions Built-in functions
- Importing from packages

Brushing the GAPS

Why we need Language (General)?

Why we need Language (Programming)?

Types of Programming Language



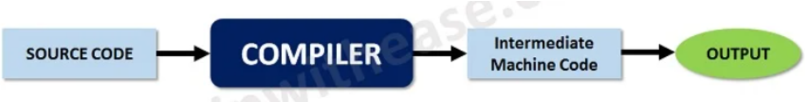
Compiler Vs Interpreter Vs Assembler

Compiler Vs Interpreter Vs Assembler

- A computer is a combination of hardware and software components.
- Hardware parts of the computers only understand the electrical charge, so the software should be written in the machine language for them to understand.
- Machine language is binary language in software
- The binary language consists of two numbers 0 and 1 indicating the power on and off. But writing a program in this language will be hard for a programmer so he writes the program in the high definition languages like C, C++, Python, etc...
- Software should be written in machine-readable form. Here is where the compiler, interpreter, and Assembler help us

Compiler

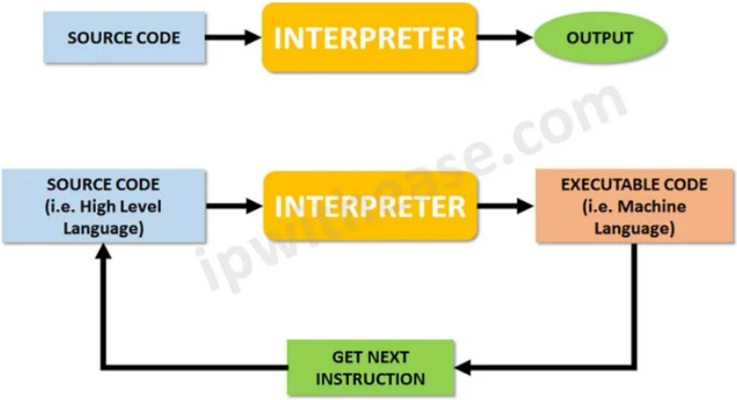
- Compiler converts the high definition programming language into machine-understandable binary codes.
- It acts more like a translator.
- Compiler converts the whole code or program into the machine language at a time.
- It checks the whole program for errors and displays them. And if there are any syntactic or semantic errors they will be indicated by the compiler. You cannot execute the program without fixing the error.



Interpreter

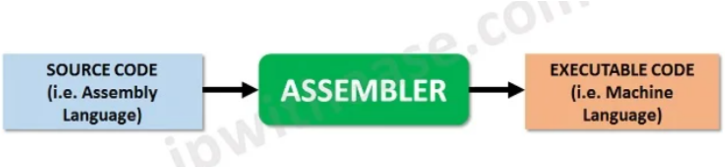
- Interpreter is a programming language translator that converts high-definition programs into machine-readable codes.
- The difference is it converts the program line by line. Thus the scanning time is lower but the overall execution time is higher.
- It translates or scans only one line at a time, you need to fix the error in the first line for the interpreter to translate the next line.

Interpreter


SOURCE CODE
INTERPRETER
OUTPUT
SOURCE CODE (i.e. High Level Language)
INTERPRETER
EXECUTABLE CODE (i.e. Machine Language)
GET NEXT INSTRUCTION

Assembler

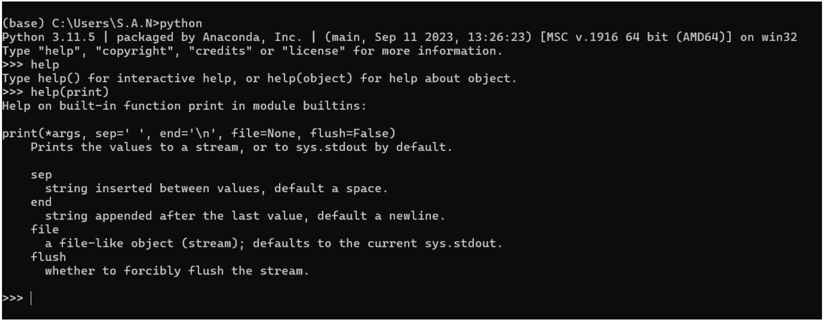
- Assembly language is like an in-between of high definition language and machine languages.
- It is also known as the low-level language.


SOURCE CODE (i.e. Assembly Language)
ASSEMBLER
EXECUTABLE CODE (i.e. Machine Language)

Understanding about Programming Language

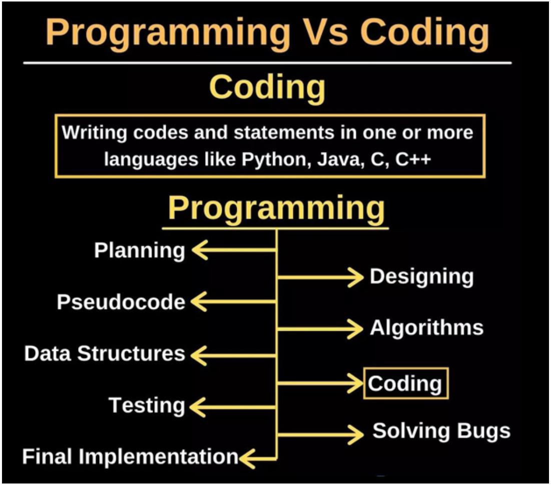
- Programming is the process of creating a set of instructions that tell a computer how to perform a task.
- It's the ability to take an idea, break it apart to individual pieces, and write the statements in proper order, using proper syntax, so that it can be fed to a machine to get the desired results.
- Communication TOOL
- Analogy – Construction

Do machines speak?



```
(base) C:\Users\S.A.N>python
Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help
Type help() for interactive help, or help(object) for help about object.
>>> help(print)
Help on built-in function print in module builtins:

print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by default.

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newline.
    file
      a file-like object (stream); defaults to the current sys.stdout.
    flush
      whether to forcibly flush the stream.

>>> |
```

Programming Vs Coding



Programming Vs Coding

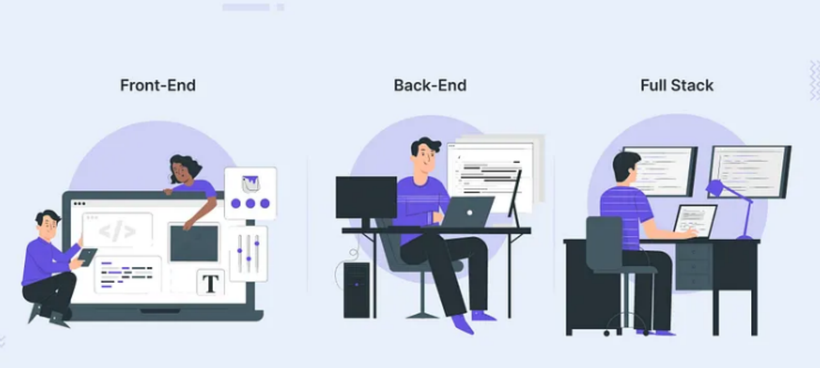
Coding

Writing codes and statements in one or more languages like Python, Java, C, C++

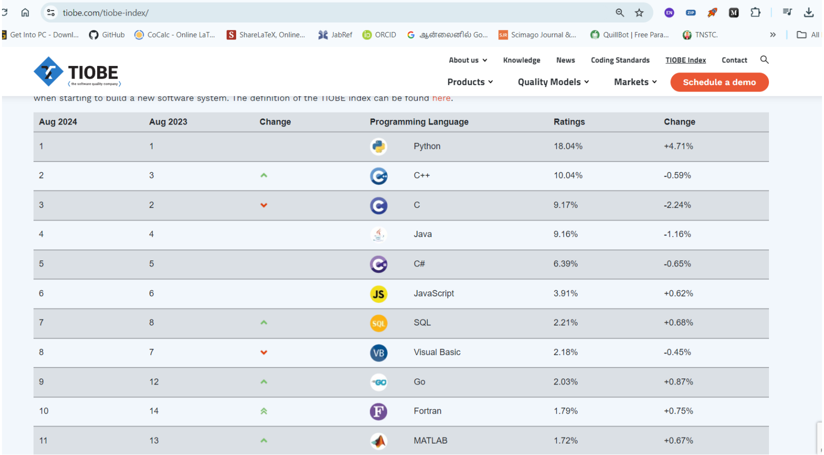
Programming

- Planning
- Designing
- Pseudocode
- Algorithms
- Data Structures
- Coding
- Testing
- Solving Bugs
- Final Implementation

Front-end Vs Back-end Vs Full Stack Programming Language



Python – best Programming Language



when starting to build a new software system. The definition of the TIOBE index can be found here.

Aug 2024	Aug 2023	Change	Programming Language	Ratings	Change
1	1		Python	18.04%	+4.71%
2	3	▲	C++	10.04%	-0.59%
3	2	▼	C	9.17%	-2.24%
4	4		Java	9.16%	-1.16%
5	5		C#	6.39%	-0.65%
6	6		JavaScript	3.91%	+0.62%
7	8	▲	SQL	2.21%	+0.68%
8	7	▼	Visual Basic	2.18%	-0.45%
9	12	▲	Go	2.03%	+0.87%
10	14	▲▲	Fortran	1.79%	+0.75%
11	13	▲	MATLAB	1.72%	+0.67%

Verdict – Language (Programming)

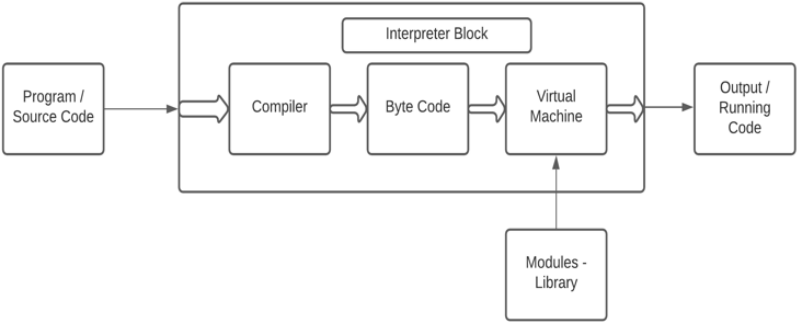
- For any relationship to be successful, there needs to be loving communication, appreciation, and understanding.

Python Programming Language

Python - Introduction

- High-level programming language
- Interpreter
- Guido Van Rossum, February 20, 1991
- General purpose
- Works in different OS
- Syntax like English
- Few lines of code - libraries
- New line means end of the command
- Python 2.x Vs Python 3.x
- .py or .ipynb

How is Python code executed?



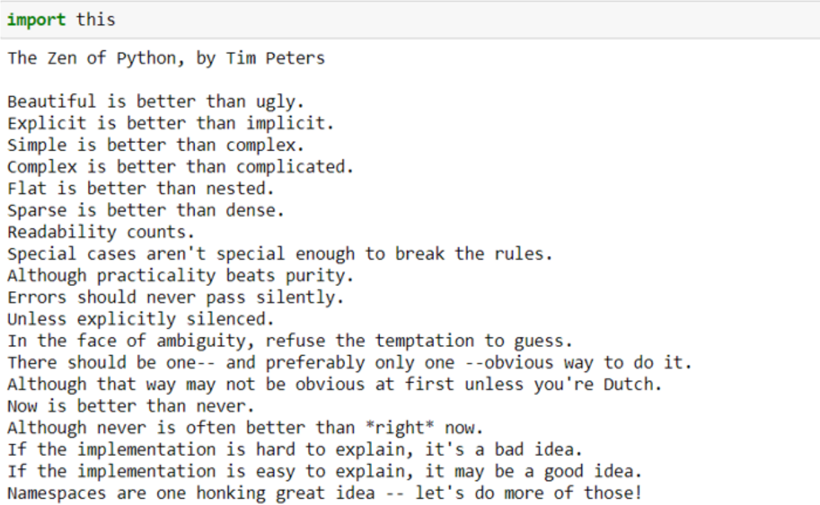
How is Python code executed?

- **Read – Evaluate – Print – Loop (REPL)**
  - Read: The Python interpreter reads the user input (source code).
  - Evaluate: It evaluates the code, processing the instructions.
  - Print: It prints the result of the evaluation.
  - Loop: It loops back to read the next input.
- **Lexing** - Line code to Token (Tokenizer - def,for,+,-,variables,datas)
- **Parsing** - Token to structure (Abstract Syntax Tree - AST)
- **Compiling** - Abstract Syntax Tree to Bytecode (.pyc) - executed by the Python Virtual Machine (PVM)
- **Interpreting** - PVM is an interpreter that translates each bytecode instruction into machine code and executes it on the hardware.

Python - Fun Facts

- Hobby Project
- Monty Python's Flying Circus
- Python overtakes French in school, UK
- Cpython, Jpython, Micro Python, Ruby Python, Brython
- MAANG almost all MNCs
- import antigravity
- import this

Python - Poet



```python
import this
```

```
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Python - Why?

- Simplicity and Readability
- Versatility and Flexibility
- Large Standard Library and Ecosystem
- Community and Support
- Interpreted and High-Level Language
- Integration with C, C++, Java, .NET...
- Portability between OS
- Industry Adoption and Career Opportunities

Python - Applications

- Website development - Django, Flask
- Desktop GUI applications - Kivy, PyQT, and Tkinter
- Gaming and 3D Graphics - Pygame, PyopenGL, Pyglet, Panda3D
- Computer Vision - Fastai, Ipsdk, Imutils, Keras, Opencv, Pytessarct, PyTorchCV, Scikit-Image, SimpleCV
- Machine Learning - Numpy, Pandas, Matplotlib, Seaborn, Scikit-learn
- Robotics - Pyro, DART, PyRobot, PyDy, Klamp't, Pybotics
- Web Scraping - Beautiful Soup, Selenium
- Scientific Computing - Scipy, BioPython
- Medical - NeuroKit2
- OS

Python - Disadvantage

- Slow speed
- Memory Efficient
- Runtime Errors
- Database Access – ODBC, JDBC
- Mobile development - KIVY, Beeware
- Simplicity
- Python is not multithreading – Global Interpreter Lock
- Incompatibility between versions
- Lacks Web Development Capabilities
- Depends on Third-party frameworks and libraries

Python - How to Code in Python?

- Python command prompt
- Command prompt or Anaconda command prompt window
- Text editor – Notepad and command prompt
- Anaconda IDE / Pycharm
- Google Colab

Python - Code Editors Vs IDE

- Code Editor (Home made food) – Text Editor, Write and Edit
- Integrated development environment (IDE) (Restaurant foods)– Software Application, Code Editor, Build Automation tools, Debugger
- IDE makes code easier by having tons of option, Code Editors is for editing the code
- IDE requires more disk space, more memory, faster processor, High end IDE costlier
- IDE key features include code editors, compiling, debugging, GUI, Syntax highlighting and many more, Code Editors key features include Syntax Highlighting, Printing, Multiview, Preview Window

Python - Modes

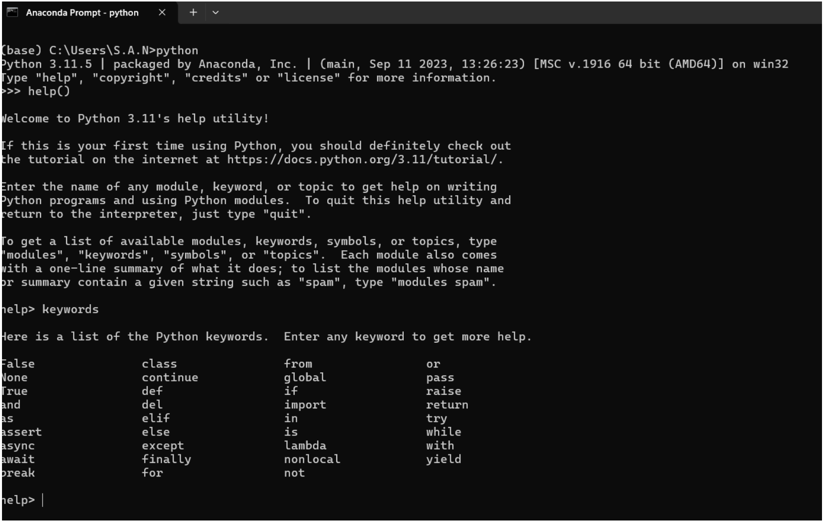
- Interactive mode
- Script mode

Python - Keywords

Python - Keywords

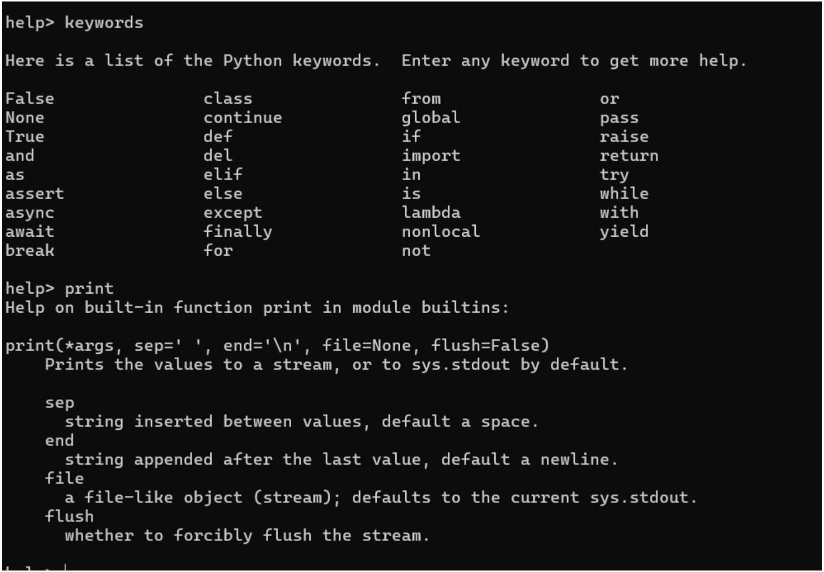
- Reserved words
- Fundamental building blocks for program
- Specific meanings and restrictions around how they should be used
- Cannot be used as variable name and function name
- Case sensitive
- 3.9.7 – 36 keywords (may differ for each versions)
- IDEs - Highlight keywords to differentiate them from other words in your code
- True, False, None – remaining all in lower case

Python - Keywords



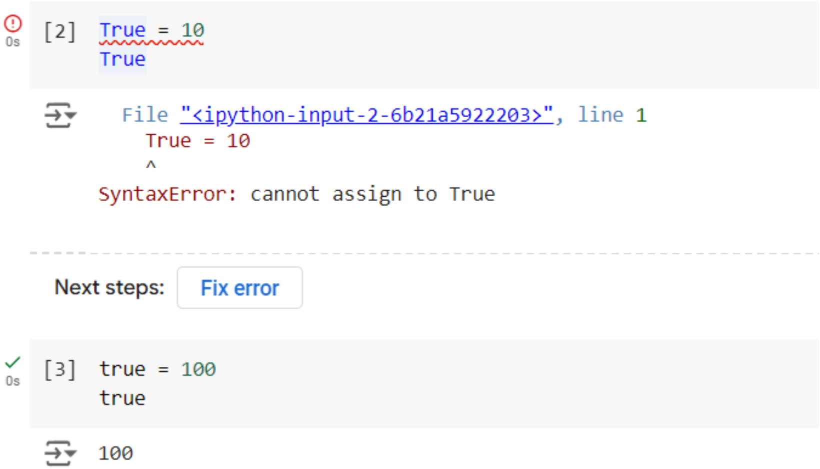
```
(base) C:\Users\S.A.N>python
Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 3.11's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the internet at https://docs.python.org/3.11/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not

help>
```

Python - Keywords - help()



```
help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not

help> print
Help on built-in function print in module builtins:

print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by default.

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newline.
    file
      a file-like object (stream); defaults to the current sys.stdout.
    flush
      whether to forcibly flush the stream.

help>
```

Keywords - Syntax Error



```
True = 10
True
```

```
  File "<ipython-input-2-6b21a5922203>", line 1
    True = 10
    ^
SyntaxError: cannot assign to True
```

```
true = 100
true
```

```
100
```

Keywords - Importance

- **Value Keywords:** True, False, None
- **Operator Keywords:** and, or, not, in, is
- **Control Flow Keywords:** if, elif, else
- **Iteration Keywords:** for, while, break, continue, else
- **Structure Keywords:** def, class, with, as, pass, lambda
- **Returning Keywords:** return, yield
- **Import Keywords:** import, from, as
- **Exception-Handling Keywords:** try, except, raise, finally, else, assert
- **Asynchronous Programming Keywords**: async, await
- **Variable Handling Keywords:** del, global, nonlocal
- Understanding their proper use is key to improving your skills and knowledge of Python.

mail me: er.anandprem@gmail.com / premanand.s@vit.ac.in
ring me: +91 73586 79961
Follow me: Linkedin
Medium Blogs
Analytics Vidhya: Blogs

**Learning gives Creativity,
Creativity leads to Thinking,
Thinking provides Knowledge,
and
Knowledge makes you Great
- Dr APJ Abdul Kalam**