WHAT IS A LIBRARY?

- A library is a collection of pre-combined codes that can be used iteratively to reduce the time required to code.
- They are particularly useful for accessing the pre-written frequently used codes, instead of writing them from scratch every single time.
- Similar to the physical libraries, these are a collection of reusable resources, which means every library has a root source.
- This is the foundation behind the numerous open-source libraries available in Python.

TERMINOLOGY

- **Scripts**
- **Modules**
- **Packages**
- **Libraries**

SCRIPT

- A **script** is a Python file that's intended to be run directly. When you run it, it should do *something*. This means that scripts will often contain code written outside the scope of any classes or functions.

MODULE

- A **module** is a Python file that's intended to be imported into scripts or other modules. It often defines members like classes, functions, and variables intended to be used in other files that import it.

PACKAGE

A **package** is a collection of related modules that work together to provide certain functionality. These modules are contained within a folder and can be imported just like any other modules. This folder will often contain a special __init__ file that tells Python it's a package, potentially containing more modules nested within subfolders

LIBRARY

- A **library** is an umbrella term that loosely means "a bundle of code."
- These can have tens or even hundreds of individual modules that can provide a wide range of functionality. Matplotlib is a plotting library.
- The Python Standard Library contains hundreds of modules for performing common tasks, like sending emails or reading JSON data.
- What's special about the Standard Library is that it comes bundled with your installation of Python, so you can use its modules without having to download them from anywhere

- Python is one of the most popular and widely used programming languages and has replaced many programming languages in the industry.
- There are a lot of reasons why Python is popular among developers and one of them is that it has an amazingly large collection of libraries that users can work with.
- Here are a few important reasons as to why Python is popular:
- Python has a huge collection of libraries.
- Python is a beginner's level programming language because of it simplicity and easiness.
- From developing to deploying and maintaining Python wants their developers to be more productive.
- Portability is another reason for huge popularity of Python.
- Python's programming syntax is simple to learn and is of high level when we compare it to C, Java, and C++.

- Hence, only a few lines of code make new applications.
- The simplicity of Python has attracted many developers to create new libraries for machine learning. Because of the huge collection of libraries Python is becoming hugely popular among machine learning experts.

OBJECT-ORIENTED PROGRAMMING

- Object-oriented programming is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields, and code, in the form of procedures.

- A feature of objects is that an object's own procedures can access and often modify the data fields of itself.

- In object-oriented programming you write classes that represent real-world things and situations, and you create objects based on these classes.

- When you write a class, you define the general behavior that a whole category of objects can have.

INSTANCES OF CLASS

- Making an object from a class is called instantiation,
- And you work with instances of a class.

ATTRIBUTES

- Variables that are accessible through instances like this are called attributes.

TOP 10 PYTHON LIBRARIES:

- TensorFlow
- Scikit-Learn
- Numpy
- Keras
- PyTorch
- LightGBM
- Eli5
- SciPy
- Theano
- Pandas

TENSORFLOW

- This library was developed by Google in collaboration with Brain Team. TensorFlow is a part of almost every Google application for machine learning.
- TensorFlow works like a computational library for writing new algorithms that involve a large number of tensor operations, since neural networks can be easily expressed as computational graphs they can be implemented using TensorFlow as a series of operations on Tensors. Plus, tensors are N-dimensional matrices which represent your data.

FEATURES OF TENSORFLOW

- TensorFlow is optimized for speed, it makes use of techniques like XLA for quick linear algebra operations.

1. Responsive Construct
2. Flexible
3. Easily Trainable
4. Parallel Neural Network Training
5. Large Community
6. Open Source

USE OF TENSORFLOW

- You are using TensorFlow daily but indirectly with applications like Google Voice Search or Google Photos. These are the applications of TensorFlow.
- All the libraries created in TensorFlow are written in C and C++. However, it has a complicated front-end for Python. Your Python code will get compiled and then executed on TensorFlow distributed execution engine built using C and C++.
- The number of applications of TensorFlow is literally unlimited and that is the beauty of TensorFlow.

SCIKIT-LEARN

- It is a Python library is associated with NumPy and SciPy. It is considered as one of the best libraries for working with complex data.

FEATURES OF SCIKIT-LEARN

- 1. Cross- Validation
- 2 Unsupervised Learning algorithm
- 3. Feature extraction

USE OF SCIKIT - LEARN

- It contains a numerous number of algorithms for implementing standard machine learning and data mining tasks like reducing dimensionality, classification, regression, clustering, and model selection.

NUMPY

- Numpy is considered as one of the most popular machine learning library in Python.
- TensorFlow and other libraries uses Numpy internally for performing multiple operations on Tensors. Array interface is the best and the most important feature of Numpy.

USE OF NUMPY

- This interface can be utilized for expressing images, sound waves, and other binary raw streams as an array of real numbers in N-dimensional.
- For implementing this library for machine learning having knowledge of Numpy is important for full stack developers.

USE OF NUMPY

- This interface can be utilized for expressing images, sound waves, and other binary raw streams as an array of real numbers in N-dimensional.

- For implementing this library for machine learning having knowledge of Numpy is important for full stack developers.

KERAS

- Keras is considered as one of the coolest machine learning libraries in Python. It provides an easier mechanism to express neural networks. Keras also provides some of the best utilities for compiling models, processing data-sets, visualization of graphs, and much more.
- In the backend, Keras uses either Theano or TensorFlow internally. Some of the most popular neural networks like CNTK can also be used. Keras is comparatively slow when we compare it with other machine learning libraries. Because it creates a computational graph by using back-end infrastructure and then makes use of it to perform operations. All the models in Keras are portable.

FEATURES OF KERAS

- It runs smoothly on both CPU and GPU.
- Keras supports almost all the models of a neural network – fully connected, convolutional, pooling, recurrent, embedding, etc. Furthermore, these models can be combined to build more complex models.
- Keras, being modular in nature, is incredibly expressive, flexible, and apt for innovative research.
- Keras is a completely Python-based framework, which makes it easy to debug and explore

A PROGRAM MUST IMPORT A LIBRARY MODULE BEFORE USING IT.

- Use `import` to load a library module into a program's memory.
- Then refer to things from the module as `module_name.thing_name`.
- Python uses . to mean "part of".
- Using `string`, one of the modules in the standard library:

```
import string
 print('The lower ascii letters are', string.ascii_lowercase)
print(string.capwords('capitalise this sentence please.'))
```

```
The lower ascii letters are abcdefghijklmnopqrstuvwxyz
Capitalise This Sentence Please
```

IMPORT SPECIFIC ITEMS FROM A LIBRARY MODULE TO SHORTEN PROGRAMS.

- Use `from ... import ...` to load only specific items from a library module.
- Then refer to them directly without library name as prefix.

```
from string import ascii_letters print('The ASCII letters are', ascii_letters)
```

```
The ASCII letters are abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Use help to learn about the contents of a library module.

```
help(string)
```

```
Help on module string:
NAME string - A collection of string constants.
MODULE REFERENCE https://docs.python.org/3.6/library/string The following documentation is automatically generated from the Python source files.
It may be incomplete, incorrect or include features that are considered implementation detail and may vary between Python implementations.
When in doubt, consult the module reference at the location listed above.
DESCRIPTION Public module variables:
whitespace -- a string containing all ASCII whitespace
ascii_lowercase -- a string containing all ASCII lowercase letters
ascii_uppercase -- a string containing all ASCII uppercase letters
ascii_letters -- a string containing all ASCII letters
digits -- a string containing all ASCII decimal digits hexdigits -- a string containing all ASCII hexadecimal digits
```

Create an alias for a library module when importing it to shorten programs.

- Use `import ... as ...` to give a library a short *alias* while importing it.
- Then refer to items in the library using that shortened name.

```
import string as s print(s.capwords('capitalise this sentence again please.'))
```

```
Capitalise This Sentence Again Please.
```

CONCLUSION

- Most of the power of a programming language is in its libraries.
- A program must import a library module in order to use it.
- Use `help` to learn about the contents of a library module.
- Import specific items from a library to shorten programs.
- Create an alias for a library when importing it to shorten programs.