1] History of Python

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in the December 1989 by **Guido Van Rossum** at CWI in Netherland.
- **Fun fact**. Python is not named after the snake. It's named after the British TV show Monty Python.
- Of course, Python, like other languages, has gone through a number of versions. Python 0.9.0 was first released in 1991.
- In 2000, Python 2.0 was released. This version of was more of an open-source project from members of the National Research Institute of Mathematics and Computer Science. This version of Python included list comprehensions, a full garbage collector, and it supported Unicode.
- Python 3.0 was the next version and was released in December of 2008 (the latest version of Python is 3.6.4). Although Python 2 and 3 are similar there are subtle differences. Perhaps most noticeably is the way the print statement works, as in Python 3.0 the print statement has been replaced with a print () function.

2] Compiler Vs. Interpreter

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming language like Python, Ruby use interpreters.	Programming language like C, C++ use compilers.

3] What types of applications can be developed by using Python?

1. GUI-Based Desktop Applications:

Python has simple syntax, modular architecture, rich text processing tools and the ability to work on multiple operating systems which make it a desirable choice for developing desktop-based applications. There are various GUI toolkits like wxPython, PyQt or PyGtk available which help developers create highly functional Graphical User Interface **(GUI). The various applications developed using Python includes:**

Image Processing and Graphic Design Applications:

Python has been used to make 2D imaging software such as Inkscape, GIMP, Paint Shop Pro and Scribus. Further, 3D animation packages, like Blender, 3ds Max, Cinema 4D, Houdini, Lightwave and Maya, also use Python in variable proportions.

Scientific and Computational Applications:

The higher speeds, productivity and availability of tools, such as Scientific Python and Numeric Python, have resulted in Python becoming an integral part of applications involved in computation and processing of scientific data. 3D modeling software, such as FreeCAD, and finite element method software, such as Abaqus, are coded in Python.

Games:

Python has various modules, libraries and platforms that support development of games. For example, PySoy is a 3D game engine supporting Python 3, and PyGame provides functionality and a library for game development. There have been numerous games built using Python including Civilization-IV, Disney's Toontown Online, Vega Strike etc.

2. Web Frameworks and Web Applications:

Python has been used to create a variety of web-frameworks including CherryPy, Django, TurboGears, Bottle, Flask etc. These frameworks provide standard libraries and modules which simplify tasks related to content management, interaction with database and interfacing with different internet protocols such as HTTP, SMTP, XML-RPC, FTP and POP. Plone, a content management system; ERP5, an open source ERP which is used in aerospace, apparel and banking; Odoo – a consolidated suite of business applications; and Google App engine are a few of the popular web applications based on Python.

3. Enterprise and Business Applications:

With features that include special libraries, extensibility, scalability and easily readable syntax, Python is a suitable coding language for customizing larger applications. Reddit, which was originally written in Common Lips, was rewritten in Python in 2005. Python also contributed in a large part to functionality in YouTube.

4. Operating Systems:

Python is often an integral part of Linux distributions. For instance, Ubuntu's Ubiquity Installer, and Fedora's and Red Hat Enterprise Linux's Anaconda Installer are written in Python. Gentoo Linux makes use of Python for Portage, its package management system.

5. Language Development:
Python's design and module architecture has influenced development of numerous languages. Boo language uses an object model, syntax and indentation, similar to Python. Further, syntax of languages like Apple's Swift, CoffeeScript, Cobra, and OCaml all share similarity with Python.

6. Prototyping:
Besides being quick and easy to learn, Python also has the open source advantage of being free with the support of a large community. This makes it the preferred choice for prototype development. Further, the agility, extensibility and scalability and ease of refactoring code associated with Python allow faster development from initial prototype. Since its origin in 1989, Python has grown to become part of a plethora of web-based, desktop-based, graphic design, scientific, and computational applications. With Python available for Windows, Mac OS X and Linux / UNIX, it offers ease of development for enterprises. Additionally, the latest release Python 3.4.3 builds on the existing strengths of the language, with drastic improvement in Unicode support, among other new features.

4] List down Python versions.
Python 3.7.3, documentation released on 25 March 2019.
Python 3.7.2, documentation released on 24 December 2018.
Python 3.7.1, documentation released on 20 October 2018.
Python 3.7.0, documentation released on 27 June 2018.
Python 3.6.9, documentation released on 02 July 2019.
Python 3.6.8, documentation released on 24 December 2018.
Python 3.6.7, documentation released on 20 October 2018.
Python 3.6.6, documentation released on 27 June 2018.
Python 3.6.5, documentation released on 28 March 2018.
Python 3.6.4, documentation released on 19 December 2017.
Python 3.6.3, documentation released on 03 October 2017.
Python 3.6.2, documentation released on 17 July 2017.
Python 3.6.1, documentation released on 21 March 2017.
Python 3.6.0, documentation released on 23 December 2016.
Python 3.5.7, documentation released on 18 March 2019.
Python 3.5.6, documentation released on 8 August 2018.
Python 3.5.5, documentation released on 4 February 2018.
Python 3.5.4, documentation released on 25 July 2017.
Python 3.5.3, documentation released on 17 January 2017.
Python 3.5.2, documentation released on 27 June 2016.
Python 3.5.1, documentation released on 07 December 2015.
Python 3.5.0, documentation released on 13 September 2015.
Python 3.4.10, documentation released on 18 March 2019.
Python 3.4.9, documentation released on 8 August 2018.

Python 3.4.8, documentation released on 4 February 2018.
Python 3.4.7, documentation released on 25 July 2017.
Python 3.4.6, documentation released on 17 January 2017.
Python 3.4.5, documentation released on 26 June 2016.
Python 3.4.4, documentation released on 06 December 2015.
Python 3.4.3, documentation released on 25 February 2015.
Python 3.4.2, documentation released on 4 October 2014.
Python 3.4.1, documentation released on 18 May 2014.
Python 3.4.0, documentation released on 16 March 2014.
Python 3.3.7, documentation released on 19 September 2017.
Python 3.3.6, documentation released on 12 October 2014.
Python 3.3.5, documentation released on 9 March 2014.
Python 3.3.4, documentation released on 9 February 2014.
Python 3.3.3, documentation released on 17 November 2013.
Python 3.3.2, documentation released on 15 May 2013.
Python 3.3.1, documentation released on 7 April 2013.
Python 3.3.0, documentation released on 29 September 2012.
Important differences between Python 2.x and Python 3.x with examples

- Division operator
- print function
- Unicode
- xrange
- Error Handling
- _future_ module

5] List out the features of Python

1) Easy to Learn and Use

Python is easy to learn and use. It is developer-friendly and high level programming language.

2) Expressive Language

Python language is more expressive means that it is more understandable and readable.

3) Interpreted Language

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4) Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5) Free and Open Source

Python language is freely available at offical web address.The source-code is also available. Therefore it is open source.

6) Object-Oriented Language

Python supports object oriented language and concepts of classes and objects come into existence.

7) Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8) Large Standard Library

Python has a large and broad library and prvides rich set of module and functions for rapid application development.

9) GUI Programming Support

Graphical user interfaces can be developed using Python.

10) Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

6] Write advantages of Python language over other languages.

1. It Is Free

Python is an open-source language. The Python company is one of the largest companies and, still, is free. Using python language doesn't require a particular subscription or a custom-built platform either, thus any desktop and laptop is compatible with Python. All the tools that are necessary for python coding, the supporting means, modules, and libraries are absolutely free.

The essential IDEs that is, the integrated development environments that include PTVS, Pydev with eclipse spyder python can be downloaded easily for free. Reduced costs are always beneficial to businesses.

2. It Needs Less Coding

Python by nature has a very simple syntax. The same logic that needs 7 lines in a C++ language, requires just 3 lines in Python. Having a smaller code requires less space, less time, and is well appreciated by coders, as the rework or correction also takes lesser time. The language aces all the parameters of readability. To support itself, the language has many built-ins and libraries that make it easy to comprehend.

Software that requires less time to code utilizes fewer resources and less time and, thus, helps in cost reduction and yields more profits.

3. All Kinds Of Businesses Can Afford It

Being a free platform, all small and medium level companies can use it. Companies that are in the nascent stage can use the python platform and begin their operations with cost-effective software. The ability to develop applications and software quickly makes it suitable for startups, as they can survive in the cutthroat competition by leveraging the speed of the python language.

4. Big Giants Are Using It

It is not only suitable for small-medium companies, but leading companies like Google, Spotify, Instagram, and Dropbox, also vouch for python development over other languages. NASA, Electronic Arts, and Disney are among the top non-IT giants who have migrated to the Python environment.

5. It Is One Of The Most Trending Languages

Java and C++ are the native languages with the Object Oriented approach. Their use is very widespread, and efficiency is tremendous. The only problem with these languages is they are lengthy. The codes are cumbersome and, thus, to correct or to rework is an immensely tedious process. Python, on the other hand, has all the features of object-oriented programming just like Java and C++, and is fast too. The codes are shorter and the syntax simple, thus being easy to amend, rework and optimize.

7] Dynamic vs. Static Types programming

Static Typing

Static typed programming languages are those in which variables need not be defined before they're used. This implies that static typing has to do with the explicit declaration (or initialization) of variables before they're employed. Java is an example of a static typed language; C and C++ are also static typed languages. Note that in C (and C++ also), variables can be cast into other types, but they don't get converted; you just read them assuming they are another type.

Static typing does not imply that you have to declare all the variables first, before you use them; variables maybe be initialized anywhere, but developers have to do so before they use those variables anywhere. Consider the following example:

```
/* C code */

static int num, sum; // explicit declaration

num = 5; // now use the variables

sum = 10;

sum = sum + num;
```

The above code fragment is an example of how variable declaration in static typed languages generally appears. Note that in the above code, static has nothing to do with static typing; it has been used along with int only to initialize num and sum to zero.

Dynamic Typing

Dynamic typed programming languages are those languages in which variables must necessarily be defined before they are used. This implies that dynamic typed languages do not require the explicit declaration of the variables before they're used. Python is an example of a dynamic typed programming language, and so is PHP. Consider the following example:

```
/* Python code */

num = 10 // directly using the variable
```

In the above code fragment, we have directly assigned the variable num the value 10 before initializing it. This is characteristic to dynamic typed programming languages.

8] Procedural vs. Object Oriented Programming+

PROCEDURAL ORIENTED PROGRAMMING	OBJECT ORIENTED PROGRAMMING
In procedural programming, program is divided into small parts called ***functions***.	In object oriented programming, program is divided into small parts called ***objects***.
Procedural programming follows ***top down approach***.	Object oriented programming follows ***bottom up approach***.
There is no access specifier in procedural programming.	Object oriented programming have access specifiers like private, public, protected etc.
Adding new data and function is not easy.	Adding new data and function is easy.

PROCEDURAL ORIENTED PROGRAMMING	OBJECT ORIENTED PROGRAMMING
Procedural programming does not have any proper way for hiding data so it is ***less secure***.	Object oriented programming provides data hiding so it is ***more secure***.
In procedural programming, overloading is not possible.	Overloading is possible in object oriented programming.
In procedural programming, function is more important than data.	In object oriented programming, data is more important than function.
Procedural programming is based on ***unreal world***.	Object oriented programming is based on ***real world***.
Examples: C, FORTRAN, Pascal, Basic etc.	Examples: C++, Java, Python, C# etc.