. what are the pros & cons of Python?

Pros

1. Easy to read, learn & code

Python is high level language & its syntax is very simple & it doesn't need any semicolons or braces & looks like English. thus, it is beginner-friendly. Due to it's simplicity, its maintaince cost is less.

2. Dynamic Typing

In python, there is no need of the decloration of variables. the data types of the variable gets assigned automatically during runtime, facilitating dynamic coding.

3. Free, Open source

It is free and also has an open source licence. This means the source code is available to the public & no one can do modifications to the original code. this modified code can be distributed with no restrictions.

4. Portable

Python is also platform-independent. that is, if you write code of the windows, Mac or Linux operating system, then you can run same code on the other operating systems with no need for any changes.

Python comes with wide range of libraries like NumPy, Pandas etc---

The Python package installer(PIP) helps you to install there libraries in your interpreter/ IDLE.

6. Wide Range of application

Python has many application like web development, making Desktop GUIs, App development, Data science etc--- it has become prefered language by proffesionals.

7. Interpreted language

Python is interpreted language, codes are executed line by line, till an error is encountered if any.

If any errors occurs at a line, it stops executing and give that error to the console. this leads to an easier & step by step debugging proces.

8. Functional, Object-oriented, Procedural

It is Functional, Object-oriented language.

Procedural means codes get executed at the top to bottom fashion.

A functional language works based on functions, rather than just statements.

A functions is a collection of codes that take input & gives output.

a) Involvement in large projects

It is used for implementation in big projects and software like YouTube, Google, Yahoo. It is also prefred language in many companies in various field like, educational, medical, research, etc.

ii) Memory management

Python also excels in managing its memory by using a separate library for this purpose.

Cons

1. It's simple nature

It's simplicity is making it hard for a programmer to adjust to the syntax of the other programming languages.

2. Slow speed & Memory inefficient

The interpreter in Python executed the code line by line, which increases the overall time.

3. Weak mobile computation

Python has many applications on the server-side. But it is hardly seen on the client-side or in mobile applications.

- It occupies a lot of memory
- This makes the process slow
- It also does not facilitate security

4) Poor Database Access

Python databases are weak compared to other technologies like JDBC, etc. This limits its use in high enterprises.

5) Runtime Errors due to dynamic typing.

Because it's a dynamically typed language, you do not declare any variable and a variable storing an integer can store a string later. This might happen unnoticeable but leads to runtime errors while doing operations.

2. History Of Python.

The programming language Python was conceived in the late 1980s, and its implementation was started in December 1989. by Guido van Rossum at CWI in the Netherlands as a successor ~~to~~ ~~at~~ ABC, capable of exception handling and interfacing with the Amoeba operating system.

Early development and release

Python was created as a successor to the ABC programming language, incorporating features like exception handling & interfacing with the Amoeba operating system. Van Rossum aimed to create a language that was more readable and easier to use than existing language like C. The name 'Python' was inspired by the BBC TV show "monty ~~Hy~~thon's Flying Circus".

In February, 1991, Van Rossum published the code (version 0.9.0) to alt.sources, which included class with inheritance, exception handling, functions and core data types like list, dict and str

- Python reached version 1.0 in January 1994, introducing functional programming tools like, lambda, map, filter & reduce.

1. Origins (Late 1980s - 1991)

- Creator: Guido van Rossum, a Dutch programmer working at Centrum Wiskunde & Informatica (CWI) in the Netherlands.
- Inspiration:
  - Guido wanted to create an easy-to-read & easy-to-use programming language.
  - He was influenced by ABC language & by languages like C, Modula-3, & Unix shell scripting.
- Goal: - A language that was powerful yet simple & emphasized code readability.
- Development started: December 1989.
- First public release: February 1991, named Python 0.9.0.
  - This version already included many core features:
    - Functions
    - Exception handling
    - Core data types (str, list, dict, etc)
    - Modules and a built-in module system

2) Python 1.x Era (1991 - 2000)

- Python 1.0 (January 1994)
  - Introduced lambda, map, filter & reduce functions.
  - Python's first wave of community growth began through comp.lang.python newsgroups.
- Python - 1.5 (December 1999)
  - Added packaging improvements, Unicode support (initially limited), & performance enhancements.

- Python 1.6 (September 2000)
  - Managed by BeOpen Python Labs, showing Python's growing popularity.

3) Python 2.x Era (2000 - 2010)

- Python 2.0 (October 2000) - a major upgrade

Key features:

- List comprehensions
- Garbage collection
- Unicode support
- Introduction of set data type
- PEP (Python enhancement Proposal) process began

- Python 2.3 to 2.7 (2003 - 2010):

Continuous improvements, adding:

- Generators (yield keyword)
- Decorators
- New-style classes
- with - statement (context managers)
- Better performance & stability

4) Transition to Python 3.x (2008 - 2020)

- Python 3.0 (December 2008) - "The Great Break"

Key changes:-

- Print became a function: print("Hello")
- Unicode by default (all strings are Unicode)
- Integer division changed (/ gives float, // gives floor division)
- New I/O system
- Removed outdated modules and syntax

- Python 3.3 - 3.8 (2012 - 2019)
  - Added a async & await for asynchronous programming
  - Type hinting (PEP 484) introduced.

5. Python 3.9+ (2020 - Present)
   - modren python versions continue to improve performance, typing & developer productivity

   - Python 3.9 (2020):
     - New dictionary merge (| operator)
     - Type hinting enhancements
   - Python 3.10 (2021):
     - Pattern matching (like switch / case)
     - structural patterns matching (match statement)
   - Python 3.11 (2022):
     - Up to 60% faster performance.
     - Better error messages.
     - Exception groups, an precise error handling
   - Python 3.12 (2023):
     - Improved memory management
     - Enhanced debugging tools
   - Python 3.13 (2024):
     - Performance upgrades
     - Improved support for the new parser (PEG)
     - Removal of deprecated features.

6. Python Today (2025 & beyond)
   - Widely used in:
     - Data science & machine learning
     - Web development (Django, Flask)
     - Automation & scripting.

- AI & Deep Learning (TensorFlow, PyTorch)
- Cloud computing, Cybersecurity, IOT

- Governance:-
  - managed by the Python software Foundation (PSF).
  - Development continues via community-driven PEPs.
- Philosophy (From "The Zen Of Python", PEP 20):
  - Beautiful is better than ugly
  - simple is better than complex
  - Readability counts.

Summary Timeline

Year	version	major Highlights
1989	Concept	Guido starts developing Python.
1991	0.90	First public release
1994	1.0	core functions, modules
2000	2.0	Unicode, garbage collection
2008	3.0	major redesign, print()
2020	End of 2.x	Full transition to 3.x
2021-2024	3.10-3.13	Pattern matching, async, type hints, performance
2025+	Future	Focus on AI, performance, interoperability.

- AI & Deep Learning (TensorFlow, Pytorch)
- Cloud computing, cybersecurity, IOT

- Governance:-
  - managed by the Python software Foundation (PSF).
  - Development continues via community-driven PEPs.
- Philosophy (From "The Zen of Python", PEP 20);
  - Beautiful is better than ugly
  - simple is better than complex
  - Readability counts.

Summary Timeline

Year	version	major highlights
1989	Concept	Guido starts developing Python.
1991	0.90	First public release
1994	1.0	core functions, modules
2000	2.0	Unicode, garbage collection
2008	3.0	major redesign, print()
2020	End of 2.x	Full transition to 3.x
2021-2024	3.10-3.13	Pattern matching, async, type hints, performance
2025+	Future	Focus on AI, performance, interoperability.