UNIT

I

1. Introduction to Python Programming and installation of Python
2. Integrated Development Environment (IDE) and Advantages of IDEs over Text Editors
3. Python Variable
4. Data Types (Integers, Floats, Strings, Booleans)
5. Type Casting and Basic Operators (Arithmetic, Comparison, Logical)
6. Input and Output Operations in Python
7. Basic String Operations in Python and Manipulation Techniques
8. Introduction to Standard Libraries and Third-Party Libraries
9. Installing and Uninstalling Python Libraries

CHAPTER

1

Introduction to Python Programming and Installation of Python

INTRODUCTION TO PYTHON

Python is a general-purpose programming language created by a Dutch programmer, Guido van Rossum and released in 1991. The name Python is inspired from a BBC comedy show Monty Python's Flying Circus **(Fig. 1.1)**.

Python is a high-level, easy-to-learn programming language that is extensively used in scientific and medical fields.

Python is widely used in various fields such as web development, data analysis, artificial intelligence (AI), machine learning (ML), scientific computing, automation, and other various functions related to programming.

It is high level (close to human language) and interpreted, which means you can run Python code directly without a separate compilation step. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

For a Pharmacy student, Python is like a universal solvent, it can be used to dissolve complex data problems, automate lab calculations, and simulate drug-receptor interactions. This makes learning and testing programs faster and easier.

In modern pharmaceutical sciences, computers are widely used for data analysis, drug design, clinical research, and hospital management systems. Among many programming languages, Python has become one of the most popular due to its simplicity and powerful capabilities.

Recruiter demand for Python remains exceptionally high, with approximately 45.7% of recruiters seeking candidates with Python skills. It is currently used by roughly 58% of developers globally.

While performance-critical tasks may still lean toward Rust or Go, Python's developer-friendly environment makes it the standard for backend APIs, data pipelines, and intelligent applications

Features of Python

1. **Easy to read:** The syntax resembles plain English, making it accessible for non-engineers.
2. **Large scientific community:** Thousands of pre-written "libraries" exist for chemistry and biology.
3. **Cross-platform:** Runs on Windows (lab PCs), macOS, and Linux.
4. **Simple and readable syntax-**code looks close to English.

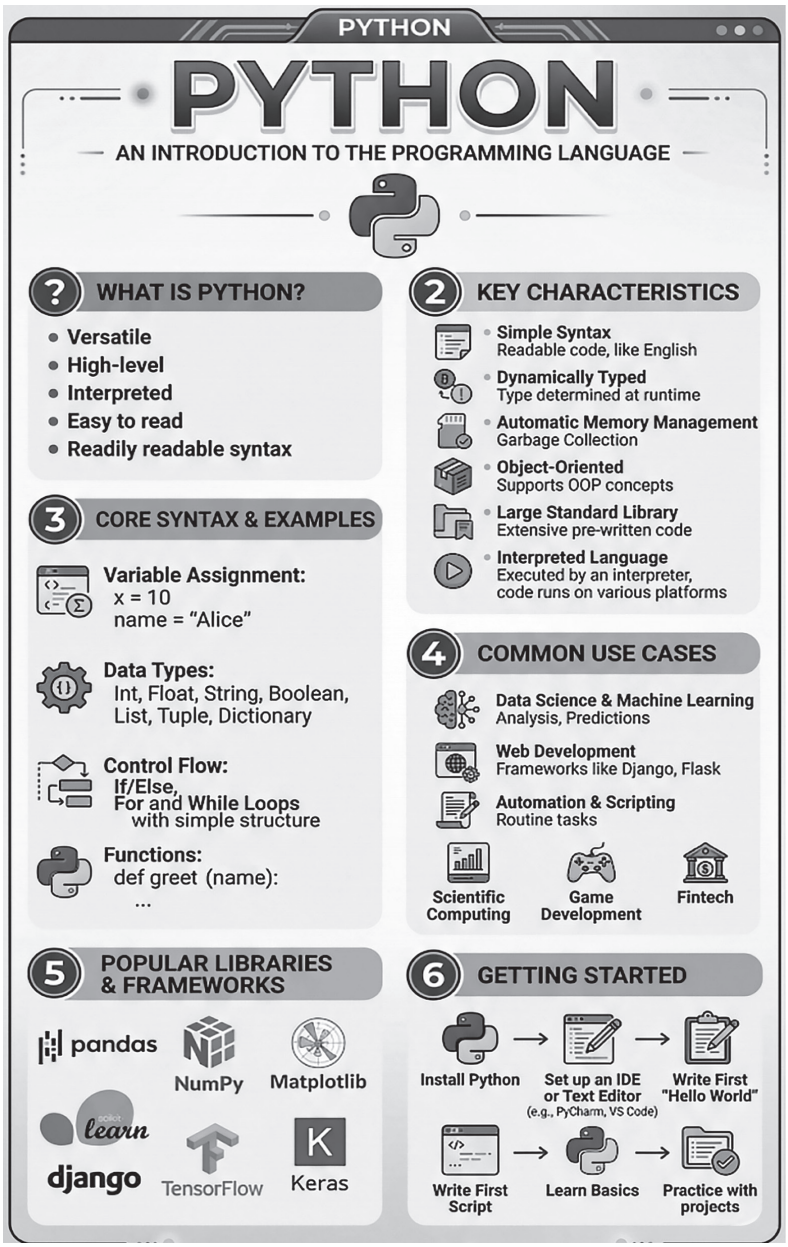


Fig. 1.1: Features of Python

5. **Dynamically typed**—you do not need to declare variable types.
6. **Interpreted language** (no need for compilation)—programs run line by line, making debugging easier.
7. **Open source and free**—available for Windows, Linux, and macOS.
8. **Large community and libraries**—many ready-made tools for science and data.
9. **Platform independent** (runs on Windows, Linux, Mac).
10. **Extensive libraries** for scientific computing.

Preference of Python over other Languages

1. **Readability and Simplicity:** Python's syntax is designed to be clear and resemble plain English. Unlike Java or C++, it does not require complex punctuation like semicolons or curly braces to define blocks of code.
2. **Rapid Development:** Python programs are typically 3 to 5 times shorter than equivalent Java programs and up to 10 times shorter than C++ code. This allows developers to move from prototype to production much faster.
3. **Dynamic Typing and Memory Management:** Developers do not need to explicitly declare variable types, as the Python interpreter detects them at runtime. It also includes automatic garbage collection, which handles memory management more efficiently than languages like C++.
4. **Dominance in AI and Data Science:** Python is the "native language" of the modern AI ecosystem. Libraries such as TensorFlow, PyTorch, and Pandas provide a massive advantage over languages that require building these complex tools from scratch.
5. **Vast Library Ecosystem:** Beyond AI, Python offers over 137,000 libraries for diverse tasks, including web development (Django, Flask) and automation.
6. **Versatile "Glue" Language:** Python excels at connecting disparate software components. It is often used to orchestrate high-performance modules written in C or C++, combining ease of use with underlying speed **(Table 1.1)**.

Applications of Python in Pharmaceutical Sciences

Python is widely used in pharmaceutical sciences because it is simple, flexible, and powerful. In this field, scientists and researchers work with large amounts of data, and Python helps them handle, analyze, and visualize that data quickly and efficiently. Its easy syntax makes it easier to learn than many other programming languages, so students and professionals can use it without much difficulty.

Python is especially useful in drug discovery, clinical research, bioinformatics, and data analysis. It supports important libraries such as Pandas, NumPy, Matplotlib, SciPy, and machine learning tools, which make research work faster and more accurate. It can also be used for automation, saving time in repetitive laboratory and data-processing tasks.

Table 1.1: Comparison with key competitors

Feature	Python	Java	C++	JavaScript
Learning curve	Gentle; very beginner-friendly	Steep; verbose syntax	Very steep; requires memory management knowledge	Moderate; easy to start for web
Development speed	Extremely fast	Moderate	Slow; complex boilerplate	Fast
Performance	Slower (interpreted)	Fast (compiled to bytecode)	Very Fast (compiled to machine code)	Fast for web/ dynamic tasks
Primary use	AI, data science, scripting	Enterprise systems, Android	Systems, gaming engines	Front-end/web development

Compared with many other languages, Python is open-source, cost-effective, and supported by a large community. This means users can easily find help, tutorials, and ready-made tools for pharmaceutical work. It is also suitable for text analysis, scientific computing, and developing research applications.

In pharmaceutical sciences, Python is needed because it improves productivity, reduces manual work, and helps in making better research decisions. For these reasons, Python has become one of the most preferred programming languages in the pharmaceutical industry.

Python is widely used in pharmaceutical sciences for:

- Drug discovery and development
- Pharmacokinetics modeling
- Bioinformatics research
- Hospital and pharmacy management systems
- Statistical analysis and visualization
- Analyzing experimental or clinical data
- Generating graphs of dose-response, time-course, or lab results
- Automating repetitive calculations and report generation tasks
- Preparing for research, data science, and bioinformatics in pharmaceutical sciences
- Easy to learn because of simple syntax
- Very useful for data driven pharmacy work
- Extensible with many libraries for statistics and graphs
- Widely used in industry and research
- Simplifies complex calculations
- Helps in research and data analysis
- Supports visualization of results
- Widely used in healthcare technologies
- Data analysis and statistics (clinical data, lab results)
- Scientific computing (dose calculations, pharmacokinetic models)
- Automation of repetitive tasks (report generation, formatting data)
- Analyzing drug response data
- Drawing graphs of dose-response, time-concentration, or lab value trends
- Automating calculations such as body surface area based dosing and statistical indices.

Limitations of Python

- Slower than compiled languages (e.g. C)
- Requires external libraries for advanced tasks
- Not ideal for very low-level programming

INSTALLATION OF PYTHON

Before writing Python programs, it must be installed and properly configured on your computer.

System requirements

Python is lightweight and runs on most systems. Basic requirements include:

- **Operating system:** Windows, macOS, or Linux
- **Memory (RAM):** Minimum 2 GB (4 GB recommended)
- **Disk space:** At least 500 MB

Downloading Python

To download Python, follow these steps:

1. Open a web browser.
2. Visit the official Python website: https://www.python.org
3. Click on Downloads.
4. The website automatically suggests the latest version for your system.
5. Click Download Python installer.

Installation on Windows

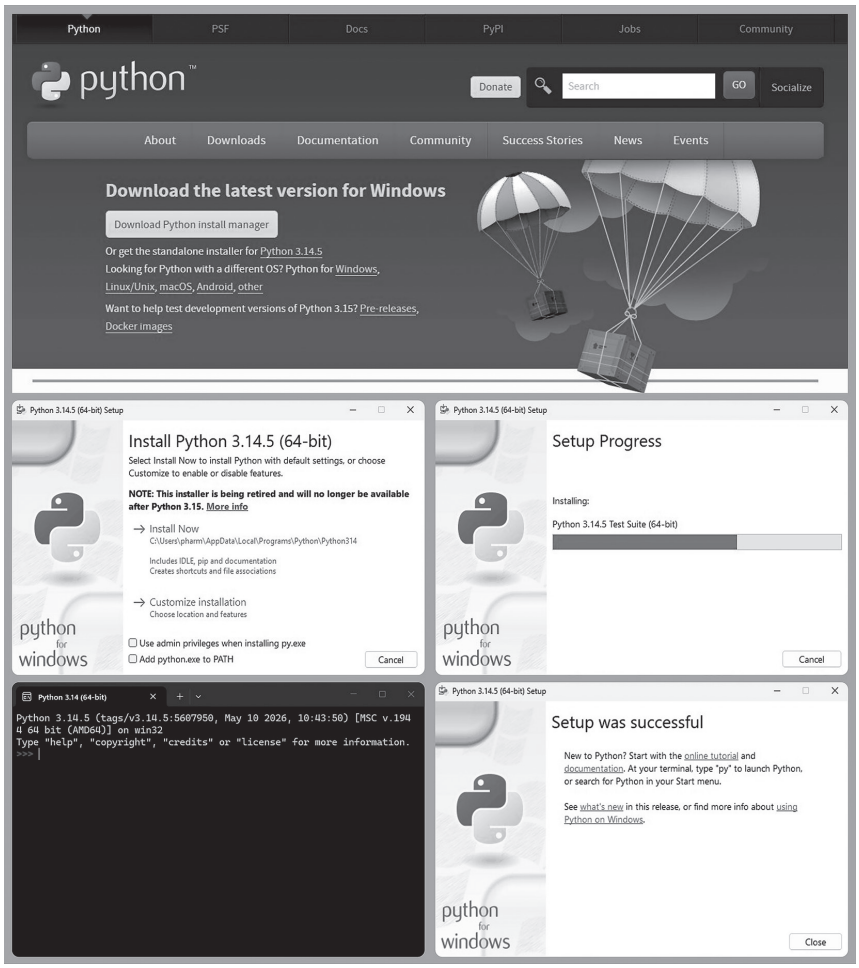


Fig. 1.2: Installation of Python in windows

Steps

1. Locate the downloaded .exe file and double-click it.
2. Check the box "Add Python to PATH".
3. Click install now.
4. Wait for installation to complete.
5. Click close after successful installation.

Installation on macOS

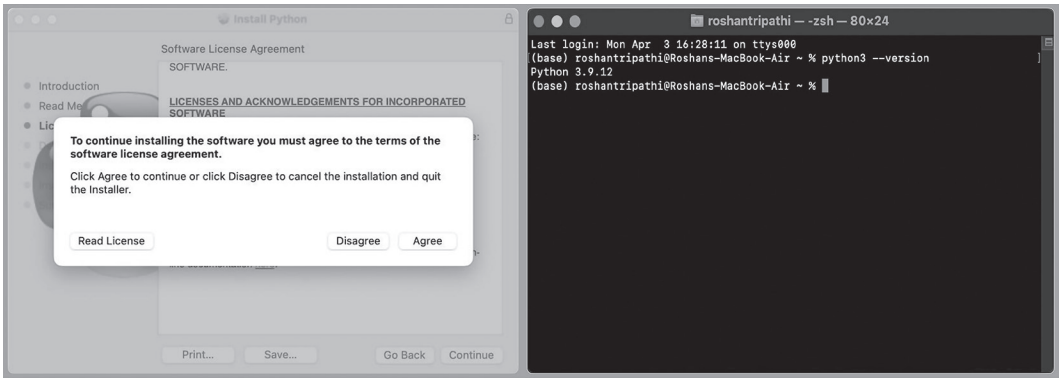


Fig. 1.3: Installation of Python on macOS

Steps

1. Download the .pkg file from the Python website.
2. Open the file and follow the installation wizard.
3. Click continue→agree→install.
4. Enter system password if required.
5. Finish installation.

Updating Python

1. Visit the official website.
2. Download the latest version.
3. Run installer (it will update existing version).

Uninstalling Python (Table 1.2)

- **Windows:** Control Panel→Programs→Uninstall Python
- **macOS:** Remove Python from Applications
- **Linux:** Use package manager (apt remove python3)

Table 1.2: Common installation issues and solutions

Problem	Solution
Python not recognized	Ensure "Add Python to PATH" is checked
Wrong version installed	Download correct version from official website
Permission errors	Run installer as administrator
Command not working	Restart system or terminal

Integrated Development Environment (IDLE)

IDLE stands for Integrated Development and Learning Environment and is Python's built-in editor and shell environment. It is included with the standard Python installation on many systems and is especially useful for beginners.

IDLE provides:

- Python shell
- Program editor
- Syntax highlighting
- Error checking
- Debugging tools

Features of IDLE

- Interactive shell for immediate execution of statements.
- Editor window for writing and saving Python scripts.
- Syntax highlighting and indentation support.
- Simple debugging and error tracing features.
- Helpful for classroom teaching and laboratory practice.

Starting IDLE and Running a Program

Steps to Open IDLE

1. Install Python from the official Python distribution.
2. Search for IDLE in the Start menu or application list.
3. Open the IDLE shell.
4. Type commands directly at the prompt for interactive execution.

Steps to Run a Script in IDLE

1. Open IDLE **(Fig. 1.4)**.
2. Select File > New File.

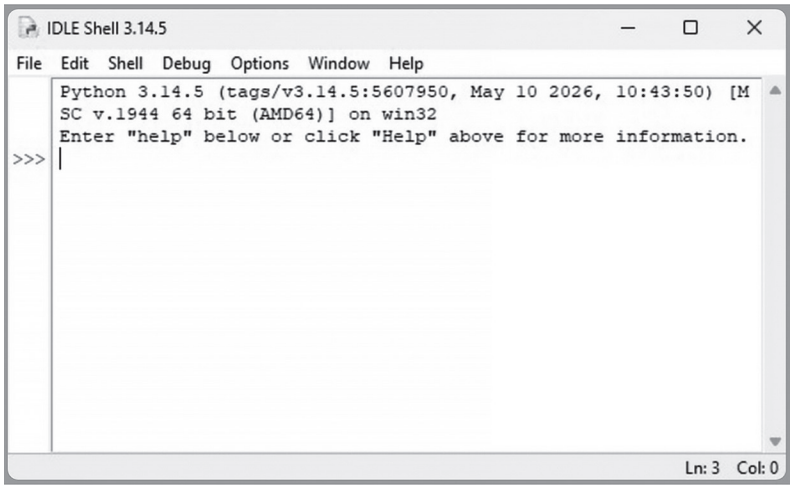


Fig. 1.4: IDLE shell in windows

3. Type your Python program in the editor.
4. Save the file with a .py extension.
5. Select Run > Run Module or press F5.
6. View the output in the shell window.

IDLE Shell and Editor

Shell Window

The shell window is used for direct interaction with Python. It supports immediate command execution and is similar to interactive mode. When a statement is completed, IDLE shows the result below the prompt.

Editor Window

The editor is used to create program files. Nothing runs automatically until the file is saved and executed. This separation helps students develop organized coding habits **(Table 1.3)**.

Table 1.3: Difference between shell and editor

Aspect	Shell	Editor
Purpose	Immediate interaction	Write complete programs
Output	Shown instantly	Shown only after running
Suitable for	Testing single commands	Practical programs
File saving	Not necessary	Required
Example use	2+2	A prescription analysis program

Python Execution Modes

Interactive Mode is a mode in which Python executes commands one at a time immediately after they are entered by the user.
It is also known as the Python Shell Mode.

How to Start Interactive Mode

- Open Python Interpreter or IDLE.
- A prompt (>>>) appears on the screen.
- Type Python statements and press Enter.

Features of Interactive Mode

- Executes one statement at a time.
- Provides instant output.
- Useful for testing small pieces of code.
- Helpful for learning Python syntax.
- No need to save the program before execution.

Advantages

1. Immediate execution and feedback.
2. Easy debugging of small code segments.

3. Suitable for beginners.
4. Useful for mathematical calculations.

Disadvantages

1. Programs are not saved automatically.
2. Difficult to manage large programs.
3. Not suitable for software development projects.
4. Code must be re-entered each time after closing the interpreter.

Applications

- Learning Python commands.
- Testing functions and expressions.
- Performing quick calculations.
- Demonstrating Python concepts in classrooms.

Script Mode

Script Mode allows users to write Python programs in a file, save them with the extension .py, and execute them later.

Use of Script Mode

Step 1: Open IDLE or any Python editor.

Step 2: Create a new file.

File → New File

Step 3: Write the program.

Step 4: Save the file.

Step 5: Run the program.

Run → Run Module or press F5.

Features of Script Mode

- Programs are stored permanently.
- Suitable for large applications.
- Supports multiple statements and functions.
- Easy maintenance and modification of code.

Advantages

1. Programs can be saved and reused.
2. Suitable for large projects.
3. Easier debugging and testing.
4. Supports modular programming.
5. Better code organization.

Disadvantages

1. Requires saving the file before execution.
2. Slightly slower for quick testing compared to Interactive Mode.

Applications

- Software development.
- Scientific computing.
- Data analysis.
- Automation scripts.
- Pharmacy and healthcare applications **(Table 1.4)**.

Table 1.4: Comparison between interactive mode and script mode

Feature	Interactive Mode	Script Mode
Execution	Line by line	Entire program
Saving	Not required	Required
Output	Immediate	After execution
Suitable for	Small programs and testing	Large applications
Debugging	Simple commands	Complete programs
Reusability	Low	High
Program storage	Temporary	Permanent

Python Program Execution Process

When a Python program is executed:

1. Source code is written in a .py file.
2. Python interpreter reads the code.
3. The code is converted into bytecode (.pyc).
4. The Python virtual machine (PVM) executes the bytecode.
5. Output is displayed to the user.

Execution Flow

Python source code

↓

Python interpreter

↓

Bytecode generation

↓

Python virtual machine (PVM)

↓

Output

Role of IDLE in Python Execution

IDLE (Integrated Development and Learning Environment) provides both execution modes:

Interactive Window

>>>

Used for Interactive Mode.

Editor Window

Used for Script Mode by creating and saving .py files.

Thus, IDLE serves as a beginner-friendly platform for learning and executing Python programs.

Practical Demonstrations in IDLE

Program: Displaying Text

```
python
print ("Welcome to Python in Pharmacy")
print ("IDLE is useful for learning")
```

Output

```
text
Welcome to Python in Pharmacy
IDLE is useful for learning
```

Simple calculations and formulas

Dose Calculation Formula

Total dose = Dose per kg × Body weight

Step by Step Numerical Example

A child weighs 18 kg. The prescribed dose is 12 mg/kg.

Total dose = 12 × 18 = 216 mg

Answer: The total dose is 216 mg.

Python Program

```
python
dose_per_kg = 12
weight = 18
total_dose = dose_per_kg * weight
print ("Dose =", total_dose, "mg")
```

Advantages and Limitations

Advantages of Python and IDLE

- Easy to learn and read.
- Suitable for beginners.
- Immediate execution in shell mode.
- Built-in editor and learning support.
- Good for educational and laboratory use.

Limitations

- Not as powerful as advanced IDEs for large projects.
- Limited professional development features compared with modern software tools.
- Not ideal for large-scale team software development.
- Some Linux installations may require separate setup for IDLE.

Recent Advances and Future Scope

Python is increasingly important in pharmacy education because it supports data analysis, visualization, automation, and the early study of AI-related tools. The future scope includes pharmaceutical informatics, clinical data analytics, pharmacovigilance reporting, drug discovery workflows, and decision-support systems.

IDLE remains useful as a learning environment, especially for beginners who need a simple interface for code testing. However, as students progress, they may move to more advanced environments such as VS Code, PyCharm, or Jupyter Notebook.

Applications in Pharmacy and Healthcare

Python and IDLE are increasingly used in the following.

Hospital Pharmacy

- Prescription processing
- Drug inventory management

Clinical Pharmacy

- Patient data analysis
- Adverse drug reaction monitoring

Pharmacovigilance

- Safety signal detection
- Clinical report analysis

Pharmaceutical Research

- Data visualization
- Statistical analysis

Industrial Applications

Pharmaceutical Manufacturing

- Production monitoring
- Quality assurance

Research and Development

- Molecular modeling
- Drug screening

Regulatory Affairs

- Documentation automation

Supply Chain Management

- Inventory forecasting
- Demand prediction

KEY POINTS

- Python is a general-purpose, high-level programming language created by Guido van Rossum in 1991.
- Python is interpreted, object-oriented, and dynamically typed.
- Python syntax is simple and resembles plain English.
- Python is widely used in:
  - Pharmaceutical sciences
  - Bioinformatics
  - Clinical research
  - Drug discovery
  - Hospital management systems
  - Statistical analysis

Preference of Python over Other Languages

- Simple and readable syntax compared to Java and C++.
- Faster program development.
- Automatic memory management.
- Strong support for AI and Data Science.
- Large library ecosystem
- Used as a "Glue Language" to connect software modules.

Applications of Python in Pharmacy

- Drug discovery and development.
- Pharmacokinetic calculations.
- Bioinformatics analysis.
- Clinical data analysis.
- Statistical analysis and graph plotting.
- Automation of laboratory calculations.
- Hospital and pharmacy management systems.
- Dose-response and time-concentration graph generation.
- Helps analyze large datasets efficiently.
- Supports automation of repetitive tasks.

Limitations of Python

- Slower than compiled languages like C++.
- Requires external libraries for advanced tasks.
- Not suitable for very low-level programming.

VERY SHORT QUESTIONS **(2 Marks)**

1. What are the major features of Python?
2. Write any four applications of Python in pharmacy.
3. What is the purpose of IDLE in Python?

Short Questions (7 Marks)

1. Discuss the advantages of Python over other programming languages.
2. Describe the applications of Python in pharmaceutical sciences.
3. Explain the installation and verification process of Python on Windows.
4. Discuss the role of Python in pharmacokinetics, bioinformatics, and drug discovery.

Long Questions (10 Marks)

1. Explain Python programming language in detail, including its features, advantages, and limitations.
2. Discuss various applications of Python in pharmaceutical sciences with suitable examples.
3. Explain the installation procedure of Python on Windows, macOS, and Linux along with common installation problems and solutions.

Objective Questions (MCQs)

1. Python was created by:

A. Dennis Ritchie
B. James Gosling
C. Guido van Rossum
D. Bjarne Stroustrup

2. Python was released in:

A. 1985
B. 1991
C. 2000
D. 1998

3. Python is a:

A. Low-level language
B. Machine language
C. High-level language
D. Assembly language

4. Which of the following is a Python IDE?

A. Eclipse
B. IDLE
C. Oracle
D. MATLAB

5. Which library is widely used for numerical computing in Python?

A. Flask
B. NumPy
C. Django
D. HTML

6. Python is mainly preferred because of:

A. Complex syntax
B. Difficult debugging
C. Readability and simplicity
D. Low portability

7. Which command checks Python version?

A. python–version
B. python install
C. run python
D. install python

8. Which of the following is a limitation of Python?

A. Difficult syntax
B. Slow execution speed
C. No libraries available
D. Platform dependence

9. Python is widely used in:

A. Drug discovery
B. Bioinformatics
C. Clinical data analysis
D. All of the above

10. Python is an:

A. Interpreted language
B. Assembly language
C. Machine language
D. Hardware language

Answers to MCQs

1. C	**2.** B	**3.** C	**4.** B	**5.** B
6. C	**7.** A	**8.** B	**9.** D	**10.** A