

Notes n Hindi

SUBJECT

SCRIPTING LANGUAGES

PREPARED BY

TEAM NOTESINHINDI

Notesinhindi.com

2026 Edition

Unit 1: Introduction, Variables and Data Types

Python की History, Installation, Basic Syntax, Errors, Variables aur Data Types
— Python सीखने की शुरुआत यहाँ से

Introduction

Python एक high-level, general-purpose programming language है, जिसे उसकी simplicity aur readability के लिए जाना जाता है। यह language beginners से लेकर professional developers तक, सभी के लिए एक popular choice है क्योंकि इसकी syntax English भाषा के काफी करीब aur आसान है।

- इस Unit में हम Python की history, installation, basic syntax, errors, variables aur data types जैसे foundational topics सीखेंगे।
- ये सभी topics किसी भी programmer के लिए बहुत जरूरी हैं, क्योंकि इन्हीं basics पर आगे की पूरी programming टिकी होती है।

Is Unit ki Structure:

- 1) History, Version and Features of Python
- 2) Python Installation and Environment Setup
- 3) Working with Python – Writing and Running Code
- 4) Basic Syntax – Input, Output, Comments and Indentation
- 5) Types of Error
- 6) Understanding Variables, Basic Data Types
- 7) String Operations, Basic Operators

1. History, Version and Features of Python

Python एक interpreted, object-oriented, high-level programming language है, जिसे **Guido van Rossum** ने design किया और सबसे पहले **1991** में release किया। इसका नाम British comedy show "Monty Python's Flying Circus" से लिया गया है, ना कि साँप (snake) से।

1.1 History (इतिहास)

- Python की शुरुआत 1980 के अंत में हुई थी, जब Guido van Rossum Netherlands की CWI (Centrum Wiskunde & Informatica) संस्था में काम कर रहे थे।
- उन्होंने एक ऐसी language बनाने की सोची जो ABC language से बेहतर हो और readable, simple code लिखने में मदद करे।
- Python 0.9.0 सबसे पहला public release था, जो **February 1991** में launch हुआ था।
- Python 2.0 साल 2000 में aur Python 3.0 साल 2008 में release हुआ, जिसमें कई बड़े changes किए गए।
- आज Python दुनिया की सबसे popular programming languages में से एक है, जो Web Development, AI, Data Science, aur Automation जैसी fields में बड़े पैमाने पर इस्तेमाल होती है।

Python — Version Timeline

Designer: Guido van Rossum



चित्र 1.1 — Python की evolution 1991 से लेकर आज तक अलग-अलग major versions में हुई।

1.2 Versions (मुख्य Versions)

Version	Release Year	मुख्य बात
Python 1.0	1994	पहला stable major release, functional programming tools (lambda, map, filter) शामिल
Python 2.0	2000	List comprehensions aur garbage collection जैसे नए features add हुए
Python 3.0	2008	Language को पूरी तरह redesign किया गया; यह Python 2 के साथ backward compatible नहीं था
Python 3.x (Current)	2008 – अब तक	Regularly update होता रहता है; आज सभी नए projects Python 3 में ही बनाए जाते हैं

Yaad rakhein: Python 2 की support 1 January 2020 को आधिकारिक रूप से बंद (end-of-life) हो गई थी, इसलिए आज सीखने और नए projects बनाने के लिए हमेशा **Python 3** ही इस्तेमाल करना चाहिए।

1.3 Applications of Python (Python कहाँ-कहाँ इस्तेमाल होती है)

Python सिर्फ सीखने के लिए ही नहीं, बल्कि real-world की बहुत सारी बड़ी industries में daily इस्तेमाल होती है। नीचे कुछ मुख्य areas दिए गए हैं जहाँ Python का इस्तेमाल सबसे ज़्यादा होता है:

Field	Python कैसे इस्तेमाल होती है
Web Development	Django aur Flask जैसे frameworks की मदद से पूरी website या web application का backend बनाया जाता है।
Data Science aur Analytics	Pandas, NumPy जैसी libraries की मदद से बड़ी मात्रा में data को analyze aur visualize किया जाता है।
Artificial Intelligence aur Machine Learning	TensorFlow, PyTorch जैसी libraries से AI models train aur deploy किए जाते हैं, इसीलिए Python इस field की सबसे popular language है।
Automation aur Scripting	Repetitive tasks (जैसे files rename करना, reports बनाना) को automatically करने वाली छोटी-छोटी scripts लिखने के लिए Python बहुत इस्तेमाल होती है।
Game Development	Pygame जैसी libraries की मदद से छोटे aur medium-level games भी बनाए जाते हैं।
Cybersecurity aur Ethical Hacking	Network scanning, penetration testing aur security tools बनाने के लिए Python widely इस्तेमाल होती है।

1.4 Python इतनी Popular क्यों है

1. **Beginner-Friendly Syntax** – Python का code लगभग plain English जैसा दिखता है, इसलिए नए students बहुत कम समय में इसे सीख लेते हैं।
2. **Huge Community Support** – दुनिया भर में लाखों developers Python इस्तेमाल करते हैं, इसलिए किसी भी problem का solution online आसानी से मिल जाता है।
3. **Cross-Domain Usage** – एक ही language को Web Development, Data Science, AI, Automation जैसी कई अलग-अलग fields में इस्तेमाल किया जा सकता है, इसलिए इसे सीखना ज़्यादा valuable माना जाता है।
4. **Rich Ecosystem of Libraries** – हर काम के लिए पहले से बनी हुई libraries उपलब्ध हैं, जिससे developers को हर छोटी चीज़ खुद से coding करने की जरूरत नहीं पड़ती।
5. **Strong Industry Demand** – बड़ी-बड़ी tech companies Python developers को अच्छी salary पर hire करती हैं, इसलिए career के नजरिए से भी यह एक smart choice मानी जाती है।

Quick Recap: Python 1991 में Guido van Rossum ने बनाई, इसका latest major version Python 3.x है, aur आज यह Web, AI, Data Science aur Automation जैसी हर field में इस्तेमाल होती है — यही वजह है कि यह दुनिया की सबसे popular programming languages में गिनी जाती है।

1.5 Features of Python

- **Simple aur Readable Syntax** – Python का code English भाषा जैसा दिखता है, जिससे इसे सीखना और पढ़ना आसान होता है।
- **Interpreted Language** – Python code line-by-line execute होता है, इसलिए errors जल्दी पकड़ में आ जाते हैं और debugging आसान होती है।
- **Dynamically Typed** – Variables का data type declare करने की जरूरत नहीं पड़ती, यह automatically decide हो जाता है।

- **Portable aur Platform-Independent** – एक बार लिखा गया Python code Windows, Linux, aur Mac जैसे अलग-अलग operating systems पर बिना बदलाव के चल सकता है।
- **Large Standard Library** – Python के साथ built-in modules और libraries की एक बड़ी range आती है, जिससे बहुत सारे काम बिना extra code लिखे किए जा सकते हैं।
- **Open Source aur Free** – Python पूरी तरह free है और इसका source code कोई भी देख या modify कर सकता है।

Advantages

- Beginners के लिए सीखना बहुत आसान है क्योंकि syntax simple aur clean है।
- बड़ी community aur documentation उपलब्ध होने से मदद मिलना आसान है।
- Web Development, AI, Data Science जैसी कई fields में इस्तेमाल हो सकती है।
- Third-party libraries की मदद से development speed काफी बढ़ जाती है।

Disadvantages

- Interpreted होने की वजह से यह C या Java जैसी compiled languages से speed में थोड़ी slow होती है।
- Mobile app development में Python का इस्तेमाल अपेक्षाकृत कम होता है।
- Dynamic typing की वजह से बड़े projects में कभी-कभी runtime errors का खतरा बढ़ जाता है।
- Memory ज़्यादा इस्तेमाल करती है, इसलिए यह हमेशा memory-efficient नहीं मानी जाती।

Example: Instagram, Spotify, aur Netflix जैसी बड़ी companies अपने backend systems में Python का इस्तेमाल करती हैं, क्योंकि इससे तेज़ी से aur reliable तरीके से development किया जा सकता है।

2. Python Installation and Environment Setup

Python code को अपने computer पर चलाने के लिए सबसे पहले **Python Interpreter** को install करना पड़ता है, aur code लिखने के लिए किसी **text editor** या **IDE (Integrated Development Environment)** की जरूरत होती है। इस पूरी setup को मिलाकर **Environment Setup** कहा जाता है।

- बिना interpreter install किए, computer Python code को समझ या run नहीं कर सकता।
- सही environment setup होने से code लिखना, run करना aur errors ढूँढना काफी आसान हो जाता है।

2.1 Types of Setup Options

Option	Description
Official Python Installer	python.org से directly Python download करके अपने computer पर install करना
Anaconda Distribution	Data Science aur Machine Learning के लिए popular, इसमें Python के साथ कई libraries पहले से आती हैं
Online IDE / Compiler	बिना install किए browser में सीधे Python code लिखने aur run करने की सुविधा (जैसे Replit, Google Colab)

Code Editor + Extensions

VS Code जैसे editors में Python extension install करके professional development environment बनाना

Python Installation — Step by Step



Installation पूरा होने के बाद आप code लिखना शुरू कर सकते हैं।

चित्र 2.1 — Windows पर Python install करने के मुख्य steps।

2.2 Working / Steps (Windows पर Installation)

1. सबसे पहले official website **python.org/downloads** पर जाकर latest Python version का installer **download** करें।
2. Downloaded file को open करें और installation शुरू करते समय "**Add Python to PATH**" checkbox जरूर select करें, ताकि Python को कहीं से भी command line पर access किया जा सके।
3. "Install Now" पर click करके installation **complete** होने दें।
4. Installation के बाद Command Prompt/Terminal खोलकर `python --version` command चलाकर **verify** करें कि Python सही से install हुआ है या नहीं।
5. अब code लिखने के लिए किसी **text editor या IDE** (जैसे VS Code, PyCharm, या Jupyter Notebook) को install करें।
6. IDE में Python extension/plugin install करें ताकि syntax highlighting, auto-completion, aur run/debug जैसी सुविधाएँ मिल सकें।

2.3 Installation on macOS aur Linux

Windows के अलावा macOS aur Linux पर भी Python install किया जा सकता है, aur इनका तरीका थोड़ा अलग होता है:

- **macOS पर:** macOS में अक्सर Python पहले से (pre-installed) होता है, लेकिन latest version के लिए python.org से installer download करें या brew install python3 command (Homebrew की मदद से) इस्तेमाल करें।
- **Linux पर:** ज़्यादातर Linux distributions में Python पहले से installed आता है। नई version install करने के लिए terminal में sudo apt install python3 (Ubuntu/Debian) जैसी command इस्तेमाल की जाती है।
- दोनों OS पर installation के बाद terminal में python3 --version टाइप करके verify किया जा सकता है कि installation सफल हुई या नहीं।

2.4 pip – Python Package Manager

pip Python का official package manager है, जिसकी मदद से third-party libraries aur packages को internet से download करके अपने project में install किया जाता है। Python installer के साथ pip अपने आप install हो जाता है।

```
# Kisi library ko install karna
pip install requests

# Installed packages ki list dekhna
pip list

# Kisi library ko uninstall karna
pip uninstall requests
```

- pip की मदद से हजारों ready-made libraries (जैसे pandas, numpy, django) कुछ ही seconds में install की जा सकती हैं।
- यह हर बार function या logic को खुद से लिखने की बजाय, पहले से बने हुए tested code को reuse करने में मदद करता है।

2.5 Virtual Environment (venv)

Virtual Environment एक isolated (अलग) space होता है जहाँ किसी specific project के लिए अलग से Python packages install किए जा सकते हैं, बिना उन्हें पूरे system पर असर डाले। इससे अलग-अलग projects की libraries आपस में conflict नहीं करतीं।

```
# Virtual environment banana
python -m venv myenv

# Use activate karna (Windows)
myenv\Scripts\activate

# Use activate karna (macOS/Linux)
source myenv/bin/activate
```

2.6 Features of a Good Python Environment

- Code editor में syntax highlighting होना चाहिए, ताकि keywords aur errors आसानी से पहचाने जा सकें।
- Built-in terminal होना चाहिए, ताकि code editor के अंदर से ही program run किया जा सके।
- Debugging tools होने चाहिए, ताकि errors को step-by-step ढूँढा जा सके।
- Package manager (जैसे pip) होना चाहिए, ताकि जरूरत के according libraries install की जा सकें।
- Extensions/plugins support करता हो, ताकि environment को अपनी जरूरत के हिसाब से customize किया जा सके।

Advantages

- एक बार सही environment set हो जाने पर coding बहुत आसान aur तेज़ हो जाती है।
- IDE की मदद से errors जल्दी पकड़ में आ जाते हैं।
- Extensions की मदद से productivity काफी बढ़ जाती है।
- Online IDEs की मदद से बिना install किए भी कहीं से भी coding संभव है।

Disadvantages

- Beginners के लिए PATH setup करना कभी-कभी confusing हो सकता है।
- अलग-अलग operating systems पर installation steps थोड़े अलग हो सकते हैं।
- पुराने computers पर heavy IDEs धीमे चल सकते हैं।
- Online IDEs के लिए हमेशा internet connection जरूरी होता है।

Example: एक student जो पहली बार Python सीख रहा है, वह VS Code install करके उसमें Python Extension add कर सकता है — इससे उसे code लिखने, run करने aur errors देखने की सारी सुविधाएँ एक ही जगह मिल जाती हैं।

3. Working with Python – Writing and Running Code

Python code को दो मुख्य तरीकों से लिखा aur run किया जा सकता है —

Interactive Mode (Interpreter/Shell) aur **Script Mode (.py file)**।

Interactive mode में एक-एक line तुरंत execute होती है, जबकि script mode में पूरी file एक साथ लिखकर run की जाती है।

- Interactive mode छोटे-छोटे code test करने के लिए useful होता है, जबकि script mode बड़े programs बनाने के लिए इस्तेमाल होता है।
- Python code हमेशा **top to bottom** order में execute होता है, यानी जो line ऊपर लिखी है वह पहले run होगी।

3.1 Types (Modes of Running Python)

Mode	Description
Interactive Mode	Terminal/Command Prompt में सीधे python टाइप करके खुलता है; हर line टाइप करते ही तुरंत execute हो जाती है
Script Mode	Code को .py extension वाली file में लिखकर save किया जाता है, फिर पूरी file एक साथ run की जाती है
Jupyter Notebook Mode	Code को छोटे-छोटे "cells" में लिखकर अलग-अलग run किया जाता है; Data Science में बहुत popular

Writing aur Running Python Code

Interactive Mode

Terminal में 'python' लिखें
हर line तुरंत execute होती है
Quick testing के लिए best

Script Mode

Code .py file में लिखें
'python file.py' से run करें
बड़े programs के लिए best

चित्र 3.1 — Interactive Mode aur Script Mode में Python code run करने का तरीका अलग होता है।

3.2 Interactive Mode – एक Example

Interactive mode में terminal खोलकर सिर्फ python टाइप करने से Python Shell (जिसे >>> prompt से पहचाना जाता है) खुल जाता है। यहाँ हर line लिखते ही तुरंत उसका result दिख जाता है:

```
$ python
Python 3.12.0
>>> 5 + 3
8
>>> print("Namaste!")
Namaste!
>>> name = "Rahul"
>>> name
'Rahul'
```

यह mode उस समय बहुत useful होता है जब आप किसी छोटे calculation या function को सिर्फ **test** करना चाहते हैं, बिना पूरी file बनाए।

3.3 IDE vs Text Editor vs Online Compiler

Tool Type	Description	Best For
-----------	-------------	----------

IDE (जैसे PyCharm)	Coding, debugging, testing सब कुछ एक ही software में मिलता है, इसमें advanced features होते हैं	बड़े, professional projects
Code Editor (जैसे VS Code)	Lightweight होता है, जरूरत के according extensions add करके इसे customize किया जा सकता है	Beginners से लेकर professionals तक, सभी के लिए
Online IDE (जैसे Replit, Google Colab)	Browser में ही बिना install किए directly code लिखने aur run करने की सुविधा देता है	Quick testing aur जहाँ installation संभव ना हो

3.4 Working Process (Script Mode)

1. सबसे पहले किसी text editor/IDE में एक नई file बनाएँ aur उसे **.py extension** के साथ save करें (जैसे hello.py)।
2. उस file में अपना Python **code लिखें**।
3. File को save करें।
4. Terminal/Command Prompt खोलें और उस folder पर जाएँ जहाँ file save है।
5. python hello.py command टाइप करके file को **run** करें।
6. Program का output terminal पर **दिखाई** देगा; अगर कोई error हो तो वह भी वहीं दिखती है।

```
# hello.py  
print("Hello, World!")
```

Terminal:

```
$ python hello.py
```

Hello, World!

3.5 Features

- Python code को बिना compile किए directly run किया जा सकता है, क्योंकि यह एक interpreted language है।
- Interactive mode की मदद से छोटे code snippets तुरंत test किए जा सकते हैं।
- Script mode में पूरा program एक file में organize होकर रहता है, जिससे बड़े projects manage करना आसान होता है।
- Modern IDEs में "Run" button दिया होता है, जिससे बिना terminal commands याद किए भी code run किया जा सकता है।

Advantages

- Interactive mode से coding concepts जल्दी समझ आते हैं।
- Script mode से बड़े, reusable programs बनाना आसान होता है।
- Errors line-by-line पकड़ में आते हैं, जिससे debugging आसान हो जाती है।
- अलग-अलग modes अलग-अलग जरूरतों (testing vs development) के लिए उपयोगी होते हैं।

Disadvantages

- Interactive mode में लिखा गया code save नहीं होता, session बंद होते ही खो जाता है।
- Beginners के लिए terminal commands याद रखना शुरू में मुश्किल हो सकता है।
- बड़े programs को interactive mode में manage करना practically मुश्किल है।

Example: एक developer जल्दी से किसी formula को test करने के लिए interactive mode इस्तेमाल करता है, लेकिन जब पूरा calculator app बनाना हो, तो वह पूरा code एक .py file में लिखकर script mode में run करता है।

4. Basic Syntax – Input, Output, Comments and Indentation

Python की **Basic Syntax** वे नियम हैं जिनके according code लिखा जाता है, ताकि Python interpreter उसे सही तरीके से समझ सके। इसमें मुख्य रूप से **input लेना, output दिखाना, comments लिखना, aur indentation** इस्तेमाल करना शामिल है।

4.1 Output – print() Function

Screen पर कुछ भी दिखाने के लिए Python में `print()` function इस्तेमाल होता है।

```
print("Namaste, Python!")
```

Output:

Namaste, Python!

4.2 Input – input() Function

User से data लेने के लिए `input()` function इस्तेमाल होता है। यह हमेशा data को **string** के रूप में लेता है।

```
name = input("Apna naam batayein: ")
```

```
print("Namaste,", name)
```

4.3 Comments

Comments वे lines होती हैं जिन्हें Python interpreter **execute नहीं करता** — यह सिर्फ code को समझाने या explain करने के लिए लिखे जाते हैं।

Type	Symbol	Example
Single-line Comment	#	# यह एक comment है

Multi-line Comment	'''...''' या """..."""	''' यह multi-line comment है '''
--------------------	---------------------------	-------------------------------------

4.4 Indentation

Indentation यानी line की शुरुआत में दी गई खाली जगह (spaces)। Python में indentation सिर्फ style के लिए नहीं, बल्कि **compulsory syntax rule** है — गलत indentation से program error दे सकता है। ज़्यादातर developers indentation के लिए **4 spaces** इस्तेमाल करते हैं।

```
if 5 > 2:
    print("5, 2 se bada hai")    # yeh line indent hai
```

Basic Syntax Elements



ये चारों मिलकर Python program की basic structure बनाते हैं

चित्र 4.1 — Python की basic syntax के चार मुख्य elements।

4.5 Features

- Python में line के आखिर में semicolon ; लगाना जरूरी नहीं होता, हर line अपने आप एक statement मानी जाती है।
- Code blocks (जैसे if, loop) को curly braces {} की जगह **indentation** से define किया जाता है।
- Comments सिर्फ readability बढ़ाने के लिए होते हैं, program के output पर कोई असर नहीं डालते।

- `input()` से मिला data हमेशा string type में होता है, जिसे जरूरत पड़ने पर convert करना पड़ता है।
- Python case-sensitive language है, यानी Name aur name दोनों अलग-अलग माने जाएँगे।

Advantages

- Syntax simple होने से code readable aur clean दिखता है।
- Indentation compulsory होने से सभी developers एक जैसी clean coding style फॉलो करते हैं।
- Comments की मदद से बड़ा code भी आसानी से समझा जा सकता है।
- Input/Output functions इस्तेमाल करना बहुत आसान aur beginner-friendly है।

Disadvantages

- Indentation में छोटी सी गलती (extra space) से भी program error दे सकता है।
- Semicolon की आदत रखने वाले (जैसे C/Java से आए) developers को शुरू में confusion हो सकती है।
- Beginners अक्सर `input()` से मिले string को number समझकर सीधे calculation में इस्तेमाल कर लेते हैं, जिससे error आता है।

Example: एक simple program जो user से उसका नाम पूछता है aur फिर उसे greet करता है, वह `input()` aur `print()` दोनों functions का इस्तेमाल करता है — यह basic syntax का सबसे common real-life उदाहरण है।

5. Types of Error

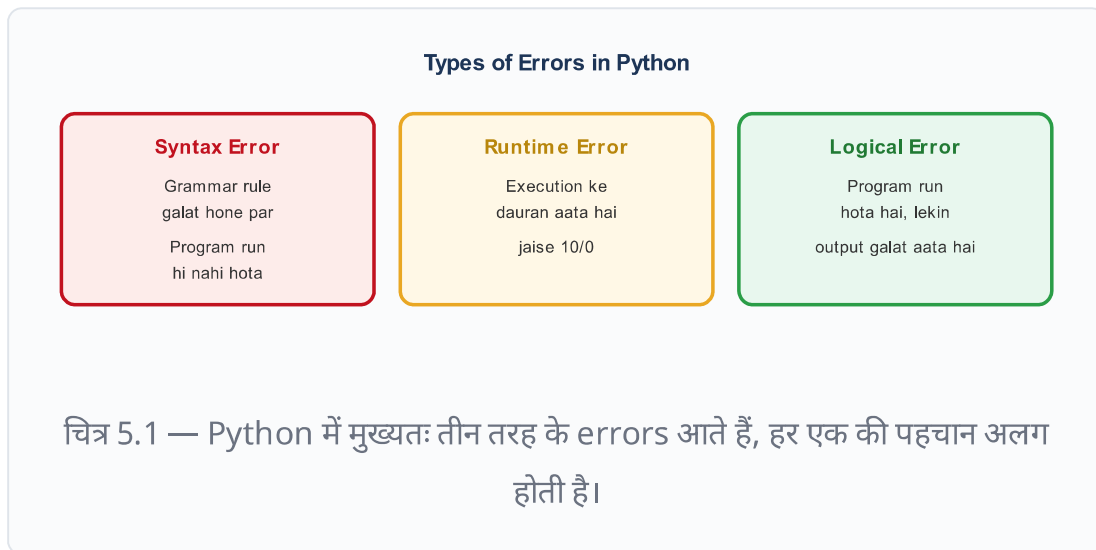
Programming करते समय code में गलतियाँ होना आम बात है, इन्हीं गलतियों को **Errors** कहा जाता है। Python में मुख्य रूप से तीन तरह के errors आते हैं — **Syntax Error, Runtime Error, aur Logical Error**।

- Errors को समझना बहुत जरूरी है, क्योंकि इसी से पता चलता है कि program में गड़बड़ी कहाँ aur क्यों हुई है।
- अलग-अलग errors को ठीक करने का तरीका भी अलग-अलग होता है, इसलिए इनके बीच फर्क समझना debugging के लिए बहुत जरूरी है।

5.1 Types

Error Type	Description	Example
Syntax Error	जब code Python के grammar rules के according सही तरीके से नहीं लिखा जाता, तब यह error आता है — program run ही नहीं होता	print("Hello" (closing bracket missing)
Runtime Error (Exception)	Syntax सही होने के बावजूद, program को run करते समय किसी unexpected स्थिति की वजह से error आता है	10 / 0 — ZeroDivisionError

Logical Error	Program बिना किसी error message के run तो हो जाता है, लेकिन output गलत आता है, क्योंकि logic में ही गलती है	Average निकालने के फार्मूले में गलत operator इस्तेमाल करना
---------------	---	--



5.2 Working / Process (Error कैसे पहचानें)

1. Program run करने पर अगर वह **start ही ना हो** aur error message आए, तो यह ज़्यादातर **Syntax Error** होता है।
2. अगर program कुछ देर सही चलता है aur बीच में अचानक **error message के साथ रुक जाए**, तो यह **Runtime Error** होता है।
3. अगर program पूरी तरह से बिना किसी error message के **run हो जाए**, लेकिन output expected से अलग आए, तो यह **Logical Error** होता है।
4. Error message को ध्यान से पढ़ें — Python आमतौर पर बताता है कि error किस **line number** पर aur किस **type** का है।
5. Error के according code को check aur correct करें, फिर दोबारा program run करके verify करें।

5.3 Features

- Python interpreter हर error के साथ एक **traceback message** देता है, जिससे यह पता चलता है कि error कहाँ हुई।
- Syntax errors compile time पर ही पकड़ में आ जाते हैं, program run होने से पहले ही रुक जाता है।
- Runtime errors को try-except block की मदद से handle किया जा सकता है, ताकि program अचानक crash ना हो।
- Logical errors पकड़ना सबसे मुश्किल होता है, क्योंकि इनमें कोई error message नहीं मिलता।

Advantages (Errors समझने के फायदे)

- Errors को समझने से debugging skills मजबूत होती हैं।
- Error messages developer को सीधे समस्या की जगह तक पहुँचाते हैं।
- Runtime errors को handle करके program को crash होने से बचाया जा सकता है।
- बार-बार errors ठीक करने से programming logic बेहतर होती जाती है।

Disadvantages (चुनौतियाँ)

- Logical errors को ढूँढने aur ठीक करने में काफी समय लग सकता है।
- Beginners के लिए लंबे error/traceback messages को समझना शुरू में मुश्किल होता है।
- बड़े programs में एक छोटी सी syntax mistake भी पूरे program को रोक सकती है।

Example: अगर कोई student `print("Result")` की जगह गलती से `print("Result"` लिखता है (bracket बंद करना भूल जाता है), तो Python तुरंत एक **SyntaxError** देगा aur बताएगा कि bracket missing है।

6. Understanding Variables, Basic Data Types

Variable एक ऐसा नाम (name) है जिसका इस्तेमाल computer की memory में किसी value को store करने aur बाद में उसे access करने के लिए किया जाता है। हर variable का एक **data type** होता है, जो यह बताता है कि उसमें किस तरह की value (number, text, आदि) store है।

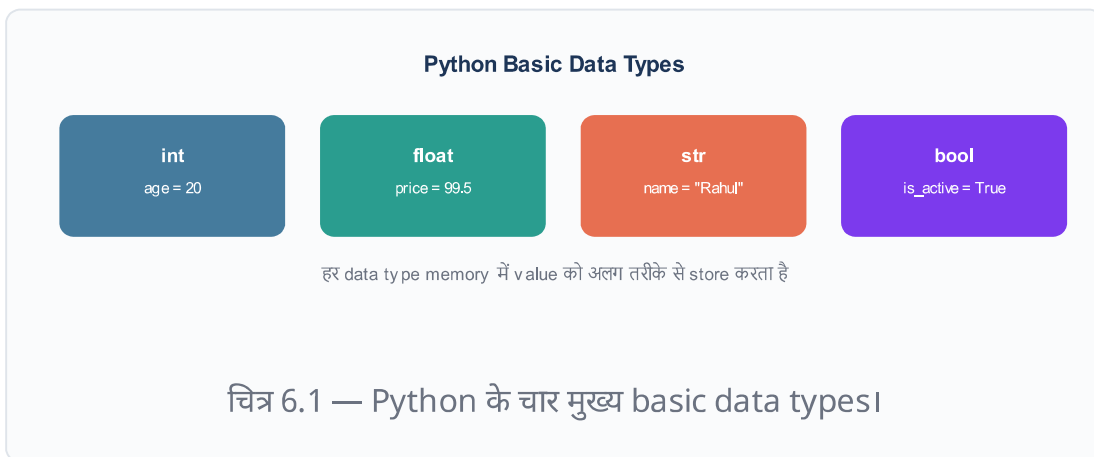
- Python में variable बनाने के लिए किसी special keyword की जरूरत नहीं पड़ती, बस नाम लिखकर उसे कोई value assign कर दी जाती है।
- Python एक **dynamically typed** language है, यानी variable का data type खुद-ब-खुद उस value से decide होता है जो उसमें assign की जाती है।

```
age = 20           # integer
price = 99.5       # float
name = "Rahul"     # string
is_active = True   # boolean
```

6.1 Types (Basic Data Types)

Data Type	Description	Example
Numeric – Integer (int)	बिना decimal point वाली पूरी संख्याएँ (positive या negative)	10, -5, 2000
Numeric – Float (float)	Decimal point वाली संख्याएँ	3.14, -0.5, 99.99

String (str)	Characters या text का समूह, जिसे single या double quotes में लिखा जाता है	"Namaste", 'Python'
Boolean (bool)	सिर्फ दो values ले सकता है — True या False	True, False



6.2 Working / Process (Variable कैसे काम करता है)

1. जब हम $x = 10$ लिखते हैं, तो Python सबसे पहले **10** value को memory में store करता है।
2. फिर **x** नाम को उस memory location से **link (reference)** कर दिया जाता है।
3. Value के according Python automatically उसका **data type decide** कर देता है (यहाँ int)।
4. जब भी code में **x** इस्तेमाल होगा, Python उस memory location से value निकालकर देगा।
5. अगर **x** को दोबारा कोई नई value दी जाए (जैसे $x = \text{"Hello"}$), तो उसका data type भी अपने आप बदलकर **string** हो जाएगा।

6.3 Features

- Variable बनाते समय कोई data type declare करने की जरूरत नहीं पड़ती — यह value से खुद तय हो जाता है।

- `type()` function से किसी भी variable का data type check किया जा सकता है।
- Variable names case-sensitive होते हैं, यानी Age aur age अलग-अलग variables माने जाएँगे।
- Variable names की शुरुआत letter या underscore (`_`) से होनी चाहिए, संख्या (number) से नहीं।
- Python के reserved keywords (जैसे `if`, `for`, `class`) को variable name के रूप में इस्तेमाल नहीं किया जा सकता।

Advantages

- Data type खुद-ब-खुद decide होने से code लिखना तेज़ aur आसान हो जाता है।
- एक ही variable को अलग-अलग समय पर अलग-अलग data types की value दी जा सकती है।
- Variables की मदद से data को बार-बार लिखने की बजाय reuse किया जा सकता है।
- Code को readable aur meaningful बनाया जा सकता है (अच्छे variable names देकर)।

Disadvantages

- Dynamic typing की वजह से कभी-कभी गलत data type की वजह से runtime error आ सकता है।
- बड़े programs में बहुत सारे variables होने पर उन्हें manage करना मुश्किल हो सकता है।
- Type declare ना होने की वजह से बड़े projects में bugs पकड़ना थोड़ा कठिन हो सकता है।

Example: एक online form में जब user अपना नाम, उम्र aur email भरता है, तो programmer इन values को `name`, `age`, aur `email` जैसे variables में store करता है, ताकि आगे इन्हें आसानी से इस्तेमाल किया जा सके।

7. String Operations, Basic Operators

String characters का एक sequence होती है, जिसे quotes में लिखा जाता है, aur इस पर कई तरह के operations किए जा सकते हैं। वहीं **Operators** वे special symbols होते हैं जिनकी मदद से values पर calculations या comparisons किए जाते हैं।

7.1 Types (String Operations)

Operation	Symbol / Method	Example	Result
Concatenation (जोड़ना)	+	"Hello" + " " + "World"	Hello World
Repetition (दोहराना)	*	"Hi" * 3	HiHiHi
Indexing	[index]	"Python"[0]	P
Slicing	[start:end]	"Python"[0:3]	Pyt
Length	len()	len("Python")	6
Case Conversion	.upper(), .lower()	"python".upper() r()	PYTHON

7.2 Types (Basic Operators)

Operator Type	Symbols	Kaam
---------------	---------	------

Arithmetic Operators	+ - * / % **	Numbers पर mathematical calculations करने के लिए
Comparison Operators	== != > < >= <=	दो values को compare करके True/False result देने के लिए
Logical Operators	and, or, not	कई conditions को आपस में जोड़कर check करने के लिए
Assignment Operators	= += -= *= /=	Variable में value assign या update करने के लिए

String Operations aur Operators

String Operations

"Hi" + "There" → HiThere

"Hi" * 2 → HiHi

"Python"[0] → P

len("Py thon") → 6

Operators

Arithmetic: + - * / % **

Comparison: == != > <

Logical: and, or, not

Assignment: = += -=

चित्र 7.1 — Strings पर operations aur Python के मुख्य operator types।

7.3 Working / Process (Example के साथ)

1. दो string variables बनाएँ — first = "Good" aur second = "Morning"।
2. + operator का इस्तेमाल करके दोनों को **concatenate (जोड़ना)** करें — first + " " + second।
3. Result को एक नए variable greeting में **store** करें।
4. print(greeting) से output **display** करें।

5. अगर numbers पर काम करना हो, तो उसी तरह arithmetic operators (+ - * /) का इस्तेमाल करके calculation करें।

```
first = "Good"
second = "Morning"
greeting = first + " " + second
print(greeting)           # Output: Good Morning

a = 10
b = 3
print(a + b)              # 13
print(a > b)              # True
```

7.4 Features

- Strings **immutable** होती हैं, यानी एक बार बनने के बाद उनकी individual characters को directly बदला नहीं जा सकता।
- Python में strings को single quotes ' ' या double quotes " " दोनों से लिखा जा सकता है।
- Arithmetic operators सिर्फ numbers पर नहीं, बल्कि + aur * strings पर भी काम करते हैं (concatenation aur repetition के लिए)।
- Comparison operators हमेशा Boolean value (True/False) return करते हैं।
- Logical operators एक साथ कई conditions को combine करके check करने में मदद करते हैं।

Advantages

- String operations की मदद से text को आसानी से format aur manipulate किया जा सकता है।
- Operators की मदद से complex calculations भी कुछ ही lines में लिखी जा सकती हैं।
- Slicing की सुविधा से string का कोई भी हिस्सा आसानी से निकाला जा सकता है।
- Logical operators से एक साथ कई conditions handle करना आसान हो जाता है।

Disadvantages

- Strings immutable होने की वजह से हर बार modification पर नई string memory में बनानी पड़ती है।
- Operators को गलत data type पर इस्तेमाल करने से (जैसे string aur int को सीधे जोड़ना) error आ सकता है।
- Beginners अक्सर = (assignment) aur == (comparison) operator में confuse हो जाते हैं।

Example: एक login form में जब system user का entered password stored password से match करता है, तो == comparison operator इस्तेमाल होता है, जो True या False return करके decide करता है कि login सफल हुआ या नहीं।