



PYTHON DASAR

Presentation by Zaky Irhab



INTRODUCTION

Python adalah bahasa pemrograman tujuan umum yang ditafsirkan, tingkat tinggi. Dibuat oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991, filosofi desain Python menekankan keterbacaan kode dengan penggunaan spasi putih yang signifikan. Konstruksi bahasanya dan pendekatan berorientasi objek bertujuan untuk membantu pemrogram menulis kode yang jelas dan logis untuk proyek skala kecil dan besar.



INTRODUCTION

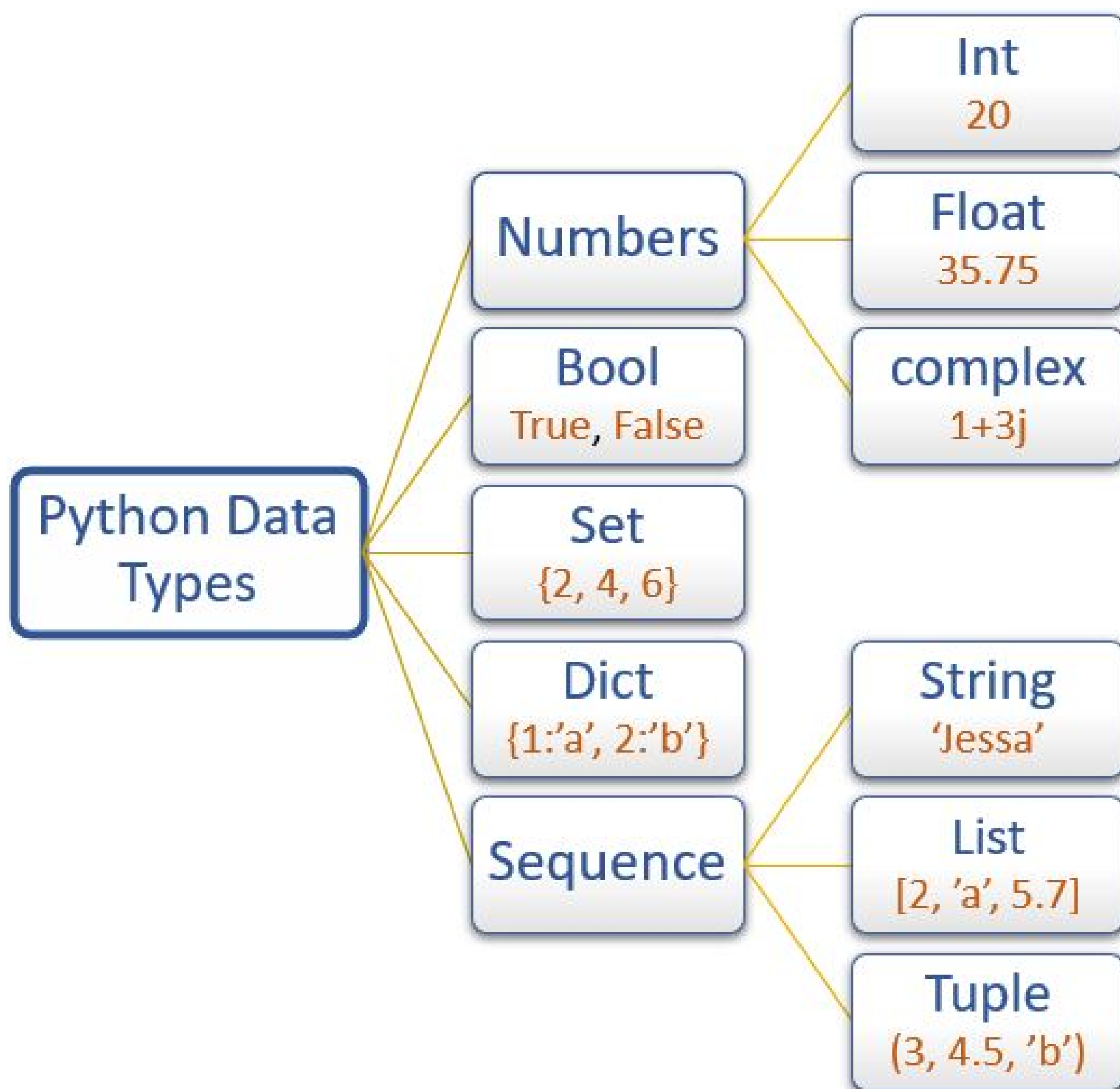
Python diketik secara dinamis dan pengumpulan sampah. Ini mendukung beberapa paradigma pemrograman, termasuk pemrograman terstruktur (terutama, prosedural), berorientasi objek, dan fungsional. Python sering dideskripsikan sebagai bahasa "termasuk baterai" karena perpustakaan standarnya yang komprehensif.



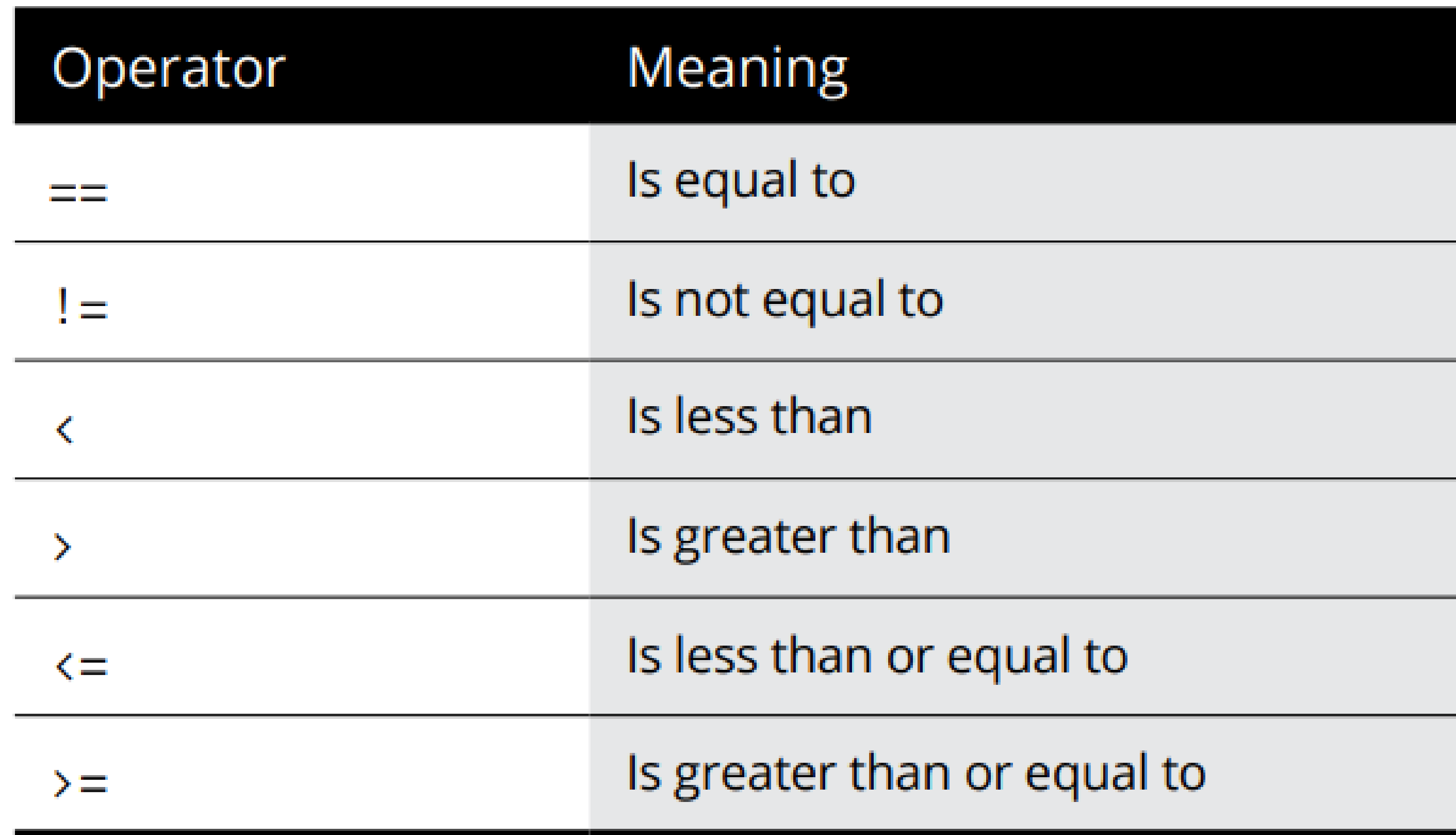
INTRODUCTION

Python dibuat pada akhir 1980-an. Python 2.0, dirilis pada tahun 2000, memperkenalkan fitur-fitur seperti pemahaman daftar dan sistem pengumpulan sampah dengan penghitungan referensi.

Python 3.0, dirilis pada tahun 2008, adalah revisi utama dari bahasa yang tidak sepenuhnya kompatibel dengan versi sebelumnya, dan banyak kode Python 2 yang tidak berjalan tanpa modifikasi pada Python 3.



Operator	Description
**	Exponentiation (raise to the power)
~ + -	Complement, unary plus and minus (method names for the last two are +@ and -@)
* / % //	Multiply, divide, modulo and floor division
+ -	Addition and subtraction
>> <<	Right and left bitwise shift
&	Bitwise 'AND'
^	Bitwise exclusive 'OR' and regular 'OR'
<= < > >=	Comparison operators
<> == !=	Equality operators
= %= /= //= -= += *= **=	Assignment operators
is is not	Identity operators
in not in	Membership operators
not or and	Logical operators



Operator	Meaning
==	Is equal to
!=	Is not equal to
<	Is less than
>	Is greater than
<=	Is less than or equal to
>=	Is greater than or equal to

The general way to write an if elif else condition

1 <some code before>	<some code here> gets executed before checking the conditional. Typically, there's an input statement here.
2 if <conditional statement>:	The keyword "if" starts the conditional line, followed by a conditional statement. The if statement ends with a colon.
3 <do something>	Indented to represent code that's executed only if the condition is True.
4 elif <conditional statement>:	The keyword "elif" followed by a colon
5 <do something>	Indented to represent code that's executed only if the condition is False.
6 else:	The keyword "else" followed by a colon
7 <do something>	
8 <some code after>	Optional: This code runs regardless of the conditional

Python Data Structures

LIST	TUPLE	DICTIONARY	SET
Allows duplicate members	Allows duplicate members	No duplicate members	No duplicate members
Changeable	Not changeable	Changeable, indexed	Cannot be changed, but can be added, non -indexed
Ordered	Ordered	Unordered	Unordered
Square bracket []	Round brackets ()	Curly brackets{ }	Curly brackets{ }

Python Functions

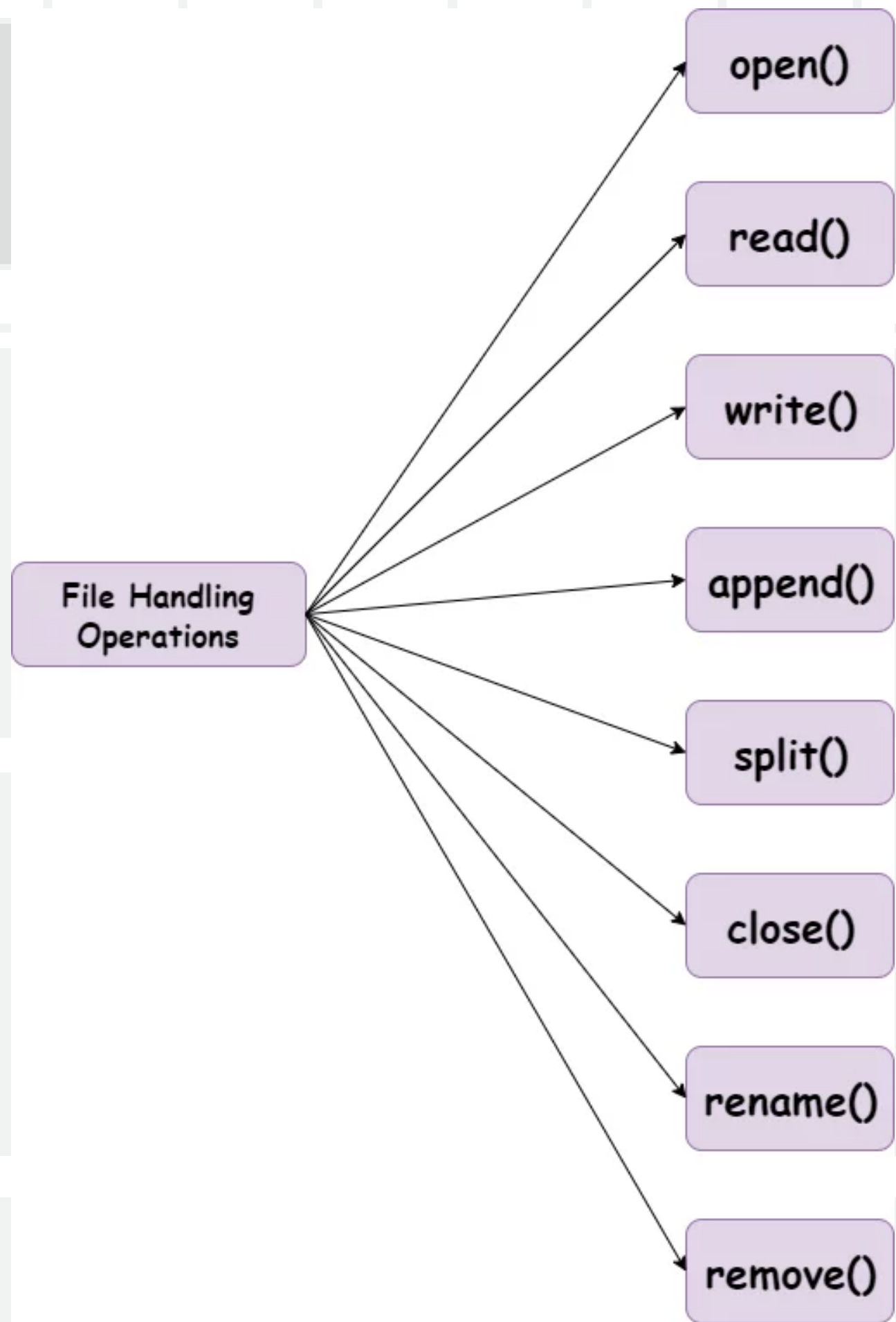
In Python, the **function** is a block of code defined with a name

- A Function is a block of code that only runs when it is called.
- You can pass data, known as parameters, into a function.
- Functions are used to perform specific actions, and they are also known as methods.
- **Why use Functions?** To reuse code: define the code once and use it many times.

The diagram illustrates the components of a Python function. It shows a function definition and a subsequent function call. Annotations with arrows identify key parts: 'Function Name' points to 'add', 'Parameters' points to '(num1, num2)', 'Function Body' is indicated by a bracket on the right for the code between the colon and the return statement, and 'Return Value' points to the value 'addition' after the 'return' keyword. The function call 'res = add(2, 4)' is annotated with 'Function call' pointing to the function name 'add'.

```
def add(num1, num2):  
    print("Number 1:", num1)  
    print("Number 2:", num1)  
    addition = num1 + num2  
  
    return addition
```

res = add(2, 4)
print(res)



try:



Run this code

except:



Execute this code when
there is an exception

else:



No exceptions? Run this
code.

finally:

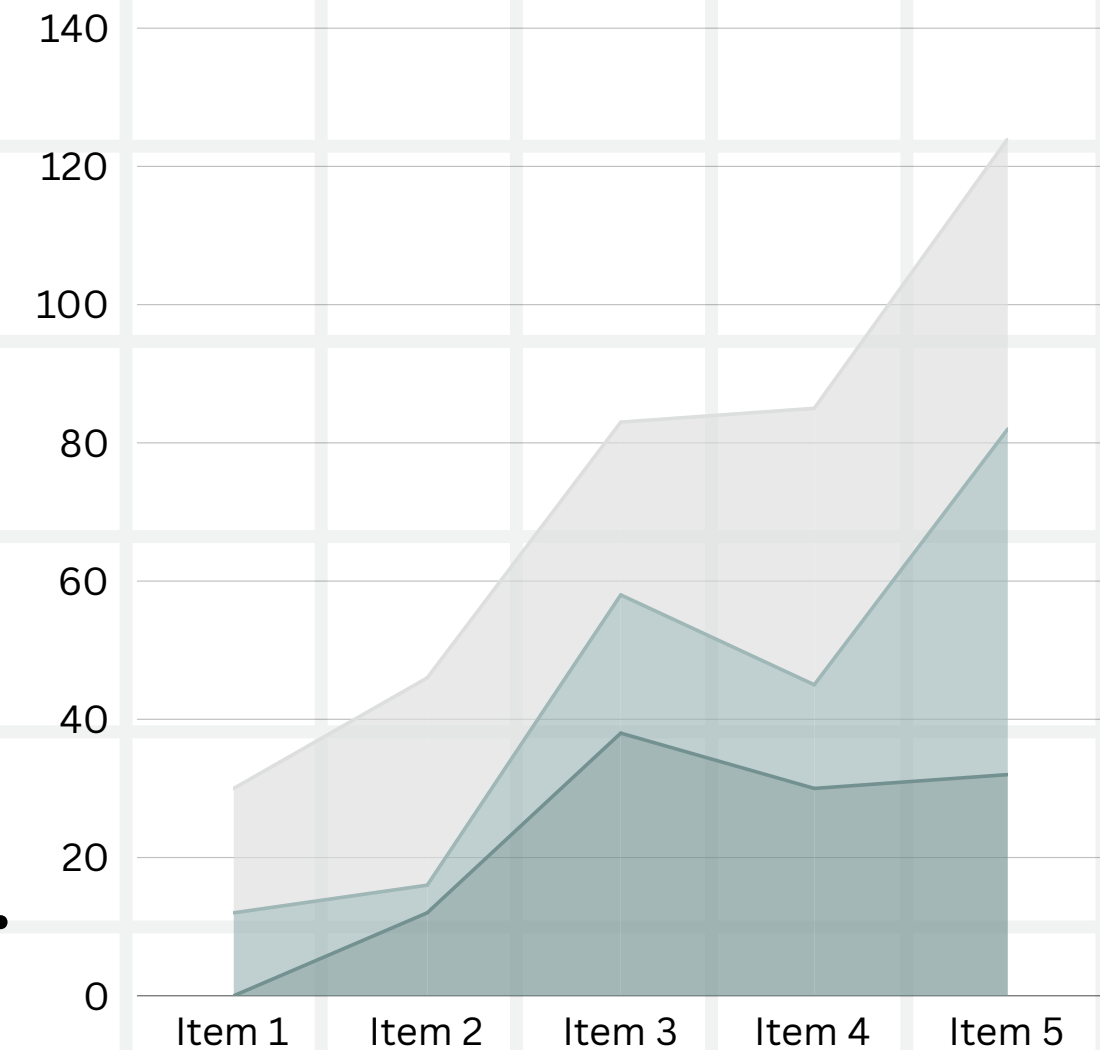


Always run this code.

OOP



Pemrograman Berorientasi Objek (OOP) adalah paradigma pemrograman yang menggunakan konsep "objek" untuk mengorganisir dan struktur kode. Dalam Python, hampir semua hal dianggap sebagai objek. Objek memiliki dua komponen utama: atribut (ciri-ciri atau data) dan metode (fungsi atau tindakan yang dapat dilakukan objek).






OOP

- **Class (Kelas):**
 - Kelas adalah sebuah blueprint atau cetak biru untuk membuat objek.
 - Contoh: Jika kita ingin membuat objek "Mobil", kita akan memiliki kelas "Mobil" yang mendefinisikan ciri-ciri (atribut) dan tindakan (metode) yang dimiliki setiap mobil.
- **Object (Objek):**
 - Objek adalah instance (salinan konkret) dari suatu kelas.
 - Menggunakan kelas "Mobil" sebagai contoh, objeknya bisa menjadi "Mobil BMW" atau "Mobil Toyota".
- **Atribut:**
 - Atribut adalah karakteristik atau data yang dimiliki oleh suatu objek.
 - Misalnya, mobil bisa memiliki atribut seperti warna, kecepatan, atau merek.
- **Metode:**
 - Metode adalah fungsi atau tindakan yang dapat dilakukan oleh suatu objek.
 - Contohnya, mobil dapat memiliki metode "Nyalakan Mesin" atau "Tambah Kecepatan".




FILE HANDLING

- Read Only ('r'):
 - Membuka file teks untuk dibaca. Posisi penanganan (handle) berada di awal file. Jika file tidak ada, akan menimbulkan kesalahan I/O. Ini juga merupakan mode default ketika membuka file.
 - Read and Write ('r+'):
 - Membuka file untuk dibaca dan ditulis. Posisi penanganan berada di awal file. Menimbulkan kesalahan I/O jika file tidak ada.
- 





FILE HANDLING

- **Write Only ('w'):**
 - **Membuka file untuk ditulis. Untuk file yang sudah ada, data akan dipotong dan ditimpa. Posisi penanganan berada di awal file. Membuat file jika file tidak ada.**
 - **Write and Read ('w+'):**
 - **Membuka file untuk dibaca dan ditulis. Untuk file yang sudah ada, data akan dipotong dan ditimpa. Posisi penanganan berada di awal file.**
- 



FILE HANDLING

- **Append Only ('a'):**
 - **Membuka file untuk ditulis. File akan dibuat jika tidak ada. Posisi penanganan berada di akhir file. Data yang ditulis akan dimasukkan di akhir, setelah data yang sudah ada.**
 - **Append and Read ('a+'):**
 - **Membuka file untuk dibaca dan ditulis. File akan dibuat jika tidak ada. Posisi penanganan berada di akhir file. Data yang ditulis akan dimasukkan di akhir, setelah data yang sudah ada.**
- 




```
# Python program to illustrate
# Append vs write mode
file1 = open("myfile.txt","w")
L = ["This is Delhi \n","This is Paris \n","This is London \n"]
file1.writelines(L)
file1.close()

# Append-adds at last
file1 = open("myfile.txt","a")#append mode
file1.write("Today \n")
file1.close()

file1 = open("myfile.txt","r")
print("Output of Readlines after appending")
print(file1.readlines())
print()
file1.close()

# Write-Overwrites
file1 = open("myfile.txt","w")#write mode
file1.write("Tomorrow \n")
file1.close()

file1 = open("myfile.txt","r")
print("Output of Readlines after writing")
print(file1.readlines())
print()
file1.close()
```






MODUL

1. Definisi:

- Sebuah modul dalam Python adalah file yang berisi definisi variabel, fungsi, dan/atau kelas.
- Tujuan utama modul adalah memecah kode menjadi bagian-bagian yang lebih kecil dan terorganisir untuk meningkatkan keterbacaan dan kemudahan pemeliharaan.

2. Penggunaan:

- Anda dapat mengimpor modul ke dalam file Python lain menggunakan pernyataan import.
 - Contoh:
- 



PAKET

1. Definisi:

- Sebuah paket adalah direktori yang berisi satu atau lebih modul serta file spesial bernama `__init__.py`.
- `__init__.py` dapat kosong atau berisi kode untuk menginisialisasi paket.

2. Penggunaan:

- Anda dapat mengimpor modul dari paket menggunakan sintaks `import paket.modul`.



Parameter	Module	Package
Definition	It can be a simple Python file (.py extensions) that contains collections of functions and global variables.	A Package is a collection of different modules with an <code>_init_.py</code> file.
Purpose	Code organization	Code distribution and reuse
Organization	Code within a single file	Related modules in a directory hierarchy
Sub-modules	None	Multiple sub-modules and sub-packages
Required Files	Only Python File(.py format)	' <code>_init_.py</code> ' file and python files
How to import	import module_name	import package_name.module_name
Example	math, random, os, datetime, csv	Numpy, Pandas, Matplotlib, django



UJIAN TIME





THANK YOU

