PAGE NO.
DATE

PAGE NO.
DATE 27 01 26

Introduction to Python & Variables

1. Python Features:

- Interpreted Language: Python executes code line by line, making debugging easier.
- Dynamically Typed: No need to declare data type of a variable explicity, the interpreter infers it at runtime.
- High-level: Syntax resembles the English language.

Variables:

A Variable is a reserved memory location to store values. It acts as a container for data.

- Syntax: Variable_name = value
- Rules for naming -
  - Must start with a letter (a-z, A-Z) or _
  - Cannot start with a number.
  - Case-sensitive (Age and age are different)

Standard data Type-

- Numeric: int (integers), float (decimal), complex (imaginary)
- Sequence: str, list, tuple
- Boolean: True or false
- Dictionary: Key value pair (dict)

PAGE NO.
DATE

. Code

```
# Variable Assignment
x = 10
y = 10.5
name = "Junior"

print(type(x))
```

Output

```
<class 'int'>
```

Operators and Expressions

An operators is a symbol that tells the compiler or interpreter to perform specific mathematical calculation, relational or logic manipulations. An expression is a combination of variable and operators (e.g, a+b)

• Type of Operators:

1. Arithmatic Operators : +, -, *, /, % (modulo) **, // (floor devision)

2. Relational operators : ==, !=, <, >, >=, <=.

3. Logical operators: • and (S1 T S2 T · R T)
• or (T F · T)
• not (T · F

4. Assignment operators : =, +=, -= etc....

Control Structure

Control structure manages the flow of execution in a program.

1. Conditional statement : Used for decision making.

- if statement : Execute a block of code if the condition is true.
- if-else statement : Execute one block if true, another if false.
- if-elif-else : Used for checking multiple condition (ladder).

2. loops : Used to repeat a block of code multiple times

- for loop : Used for iterating over a sequence (like a list, tuple, dict, str).
- While loop : Repeat a statement or groups of statements while given condition is TRUE.

3. Jump statement:

- break : Terminates the loop
- continue : skips the current iteration and goes to the next one.
- pass : Null statement (does nothing). Used as a place holder.

PAGE NO.
DATE

• Code

```
# Conditional
marks = 85
if marks > 90:
    print("Grade A")
elif marks > 80:
    print("Grade B")
else
    print("Grade C")

# for loop

for i in range(5)
    print("Value:", i)

# While loop

Count = 0
while Count < 3:
    print("Count is:", count)
    count = count+1
```

• Output

```
Grade B

Value: 0
"  1
"  2
"  3
"  4

Count is: 0
"  1
"  2
```

String & File Handling

A string is a sequence of characters enclosed in single (' ') or double (" ") quotes. Strings in Python are immutable, meaning they cannot be changed after creation.

- Basic operation

. Subscript operator ([ ]): Used to access individual characters. str[0] gives the first character

. Indexing: Python supports both positive (0 to n-1) and negative (-1 to -n) indexing. -1 refers to the last character.

. Slicing ([:]): Extracting a part of a string.
Syntax: string [start : end : step]
Note: The end is excluded.

- Common Methods:

. len(str): Returns length.
. str.upper() / str.lower(): changes case.
. str.find(substring): Returns index of substring.

PAGE NO.
DATE

Code	Output
`text = "Python"`	P
	~~nohtyP~~ n
`print(text[0])`	Py
`print(text[-1])`	thon
`print(text[0:2])`	
`print(text[2:])`	

- File handling allows a program to store data permanently on a disk. Python provides built-in functions to create, read, update and delete file.
- Opening a File : Syntax : file_object = open("filename", "mode")
- File modes :
  - 'r' (Read) : Default mode. opens file for reading. Error if file doesn't exist.
  - 'w' (write) : Opens file for writing. Creates a new file or overwrites existing content.
  - 'a' (append) : Opens file for writing. Adds data to the end of the file without deleting old content.
- Reading/Writing Methods :
  - read() : Reads the entire file.
  - readline() : Reads one line at a time.
  - Writing(string) : write string to the file.
  - close() : closes the file and frees memory resources.

PAGE NO.
DATE

- Codes

```
f = open("notes.txt", "w")
f.write("Hello Junior!\n")
f.write("Python file handling easy hai.")
f.close()

f = open("notes.txt", 'r')
content = f.read()
print("file content:\n", content)
f.close()
```

data
H-in functions

"filename", "mode")

reading.

a new file

data to the
ng old content.

mory

List and Dictionaries

- A list is an ordered, mutable collection of items enclosed in square brackets [ ]. It can contain mixed data types.

  Syntax: My_list = [1, "Hello", 3.5]

- A Dictionary is an unordered, collection of data stored as key-Value pairs. It is enclosed in ~~Only~~ Curly braces {}. Keys must be unique and immutable (like strings/numbers).

  Syntax: my_dict = {"key": "value", "ID": 101}
  Accessing: my_dict["key"] returns "value".

- Code

```
fruits = ["Apple", "Banana", "cherry"]
fruits.append("Orange")
print(fruits[1])

student = {
    "name": Junior
    "roll_no": 250,
    "branch": "AIML"
}
print(student["name"])
student["grade"] = "A"
```

Output

```
Banana
Junior
```

Function

A function is a block of organized, reusable code that performs a single, related action. It runs only when it is called.

- syntax

```
def function_name(parameters):
    """docstring"""
    statement
    return expression
```

- Key Terms:

def : keyword to define a function.
Parameters: Variable listed inside parenthesis in the defination.
Arguments: Values sent to the function when it is called.
Return : Exits the function and passes a value back to the caller.

- Code

```
def calculate_area(radius):
    pi = 3.14
    area = pi*(radius**2)
    return area

result = calculate_area(5)
print("Area of circle:" result)
```

Output

78.5

PAGE NO.
DATE

OOPs Concept

1. Class ⇒ A user-defined data type that acts as a blueprint/template for creating objects. It defines attributes and method.

2. Objects: An instance of class. It has a state (~~object~~. (attributes) and behavior (methods).

3. The __init__ Method: A specid method (constructor) that is automatically called when an objects is Created. It is used to initalize variables.

4. self : Represents the ccurent instance of the class (like me or this)

```
Class Student:
    def __init__(self, name, marks):
        self.name = name
        Self. marks= marks

    def show_result(self):
        if self.marks>40:
            print(self.name,"is Pass!")
        else
            print(self.name,"is fail!")
```

```
s1 = student("Amit", 85)
s2 = student("Rahul", 30)
s1.show_result()
s2.show_result()
```

Amit is Pass!
~~Am~~
Rahul is Fail!

Inheritance (OOPs advanced)

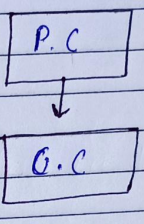
Inheritance allows a class (child) to acquire the properties and method of another class (Papa). It supports reusability of code.

Syntax

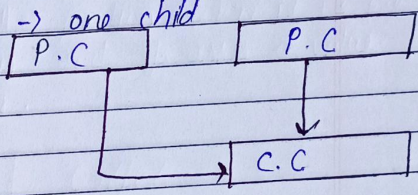
```
class childclassname (Parentclassname)
    # child class body
```

Types -

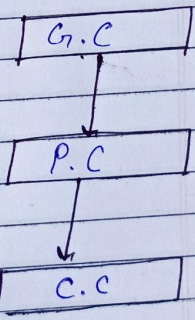
1. Single → One parent → one child



2. Multiple → Multiple parents → one child



3. Multilevel → Grandparent → parent → child



PAGE NO.
DATE

- Code

```
class animal :
    def speak (self) :
        print ("Animal is speaking")

class Dog (Animal) :
    def bark ( self) :
        print (" Dog barks : Woof Woof")

d = Dog ()
d. bark ()
d. speak()
```

Output

```
Dog barks : Woof Woof
Animal is speaking
```