11/09/25

Introduction to python:

→ High-level, interpreted, object-oriented language.

→ Created by Guido von Rossum in 1991.

→ Open-source and cross-platform.

Why should learn python:

1. Easy to learn and Beginner-friendly.
2. Versatile language (used Everywhere)
3. High Demand in Jobs.
4. Community Support
5. Cross-platform (Windows, Mac, Linux).

Features of python:

1. Simple and Easy to learn.
2. Interpreted language.
3. Dynamically Typed.
4. Object-Oriented.
5. High-Level Language.
6. portable and platform independent.
7. Free and open source.
8. Extensive libraries.
9. Strong Community Support.
10. Integration with other languages like C, C++, Java.

Syntax

```
print("Hello, World!")
```

Advantages of python:

1. Easy to learn and Read.
2. Works on all platforms (Windows, Mac, Linux).
3. Many built-in libraries
4. Support oop and functional programming.
5. Fast to Develop projects.

Disadvantages of python:

1. Slower than C, C++, or Java.
2. Use more memory.
3. Not good for mobile apps.
4. Error may come at Runtime.
5. Not strong in database handling.

Assignment

1.) What is python and why is it called an interpreted language?

A/ 1. python is a high-Level, general-purpose, object-Oriented programming language.

2. It was created by Guido Van Rossum in 1991.

3. It is Widely used for Web development, data science, AI, machine learning, automation and more.

4. It is interpreted because it executes the program line by line using the python interpreter.

2) What are the key features of python that make it popular for beginners and professionals?

A) 1. Easy to learn and use.
2. Expressive language.
3. Interpreted language.
4. Cross-platform language.
5. Free and open source.
6. Object-oriented language.
7. High-level Language.
8. Dynamic Typed.

3) What is the Difference between python 2 & python 3?

A)

python 2	python 3
1. It was Released in 2000.	1. It was Released in 2008.
2. Legacy version.	2. Current version.
3. print is a Statement ↳ print "Hello"	3. print is a function. ↳ print ("Hello")
4. Strings are ASCII by default.	4. Strings are unicode by default.
5. Supports old-style classes and new-style.	5. only new-style classes.
6. More focus on backward Compatability.	6. More focus on future improvements.

4) What are python's applications in real-world projects ?

A) python is everywhere because it's simple & powerful.

i) Web Development. (Django, Flask)

ii) Data science and Analytics (pandas, Numpy, Matplotlib).

iii) Artificial Intelligence and ML (Tensorflow, scikit-learn).

iv) Automation / scripting.

v) Game Development (pygame)

vi) IOT.

5) What is PEP 8 and why is it important in python programming ?

A) PEP = python Enhancement proposal.

→ PEP 8 is a style guide for writing clean, readable, and consistent python code.

→ It gives rules on naming, indentation, spacing, line length, Comments, imports, etc.

Why it is important ?

1. Readability → code looks neat and easy to understand.
2. Consistency → Everyone in a team follows the same style.
3. Collaboration → Makes it easier to work on large projects with multiple developers.

4. professional standard → Most Companies follow PEP 8 to maintain high-quality code.

6) Who developed python and which year was it released?

A) → python was developed by Guido van Rossum.

→ first released in 1991.

7) What do you mean by "Dynamically Typed" in python?

A) python is Dynamically Typed because the type of variable is decided at runtime, not during declaration.

8) What is the difference between Compiler and interpreter and which does python use?

A) Compiler: A Compiler Executes the whole program at once.

Interpreter: A Interpreter Executes the program "Line by Line"

→ python uses an Interpreter.

12/09/25

Data Type:

It Defines about the kind of data.

Two types of Data Types:

1. Single valued Data Types.
2. Multiple valued Data Types

1. Single valued Data Types :

i) int
ii) Float
iii) Complex
iv) String.

2. Multivalued Data types :

i) List
ii) Tuple
iii) Dict
iv) Set
v) String.

Input / output function :

Input() ; When we take input from the user.

output() ; Something we have display for the users.

```
name = input("enter your name: ")
age = int(input("Enter your age: "))
```

Output :

Enter your name : Vasantha.
Enter your age : 21

If I want to print name and age in one line.

like my name is Vasantha and age is 21

print ("My name is", name, "and age is", age)

o/p: My name is vasantha and age is 21.

We have to use format string:

print("My name is {} and age is {}".format(name, age))

o/p: My name is vasantha and age is 21.

We have to use fstring:

print(f"my name is {name} and age is {age}")

o/p: my name is vasantha and age is 21.

15/09/25

Conditional statements:

1) if
2) if else
3) if else ladder
4) Nested if else.

if: Code to be executed if the condition is true.

Syntax:

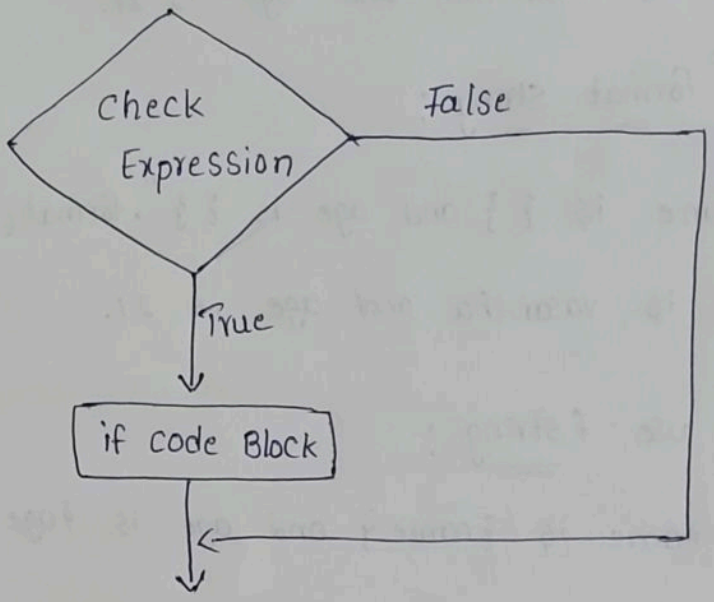
```
if (condition):
    statements
```

program:

```
age = int(input())
country = input()
if (age>=18 and country == "india"):
    print("Eligible for voting")
```

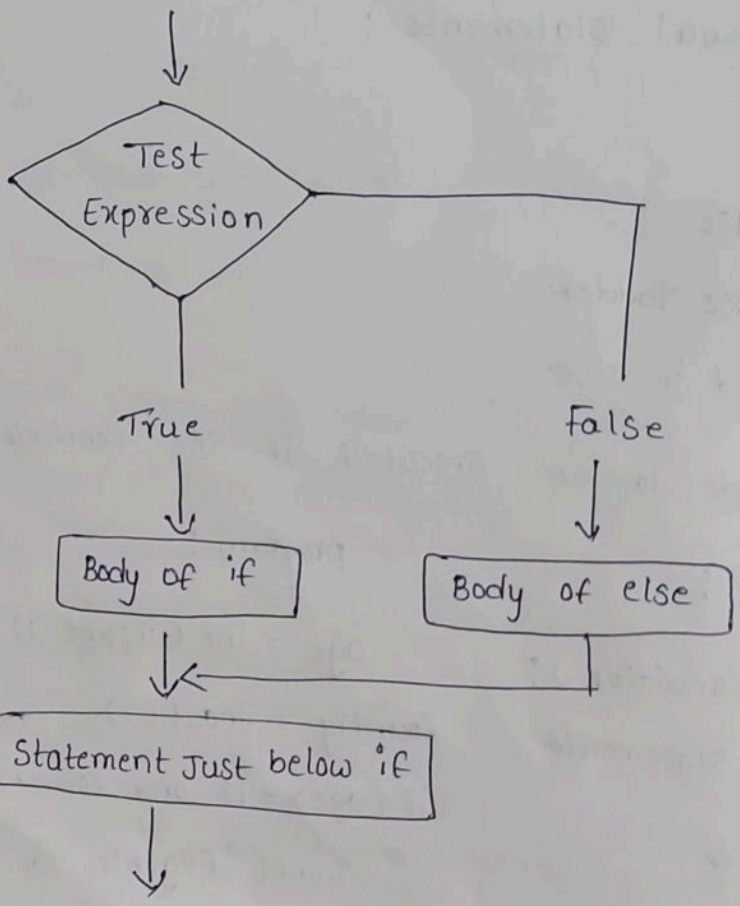
o/p: 19
india
Eligible for voting.

1) if conditional statement flow control :



2) if else :

Flowchart of if-else statement



Definition :

→ if-else is a decision-making statement

→ It lets the program choose different paths based on conditions.

→ Condition → must evaluate to True or False.

Syntax :

```
if (condition):
    Statement
else:
    Statement of else.
```

program :

```
age = int (input())
Country = input()
if (age>=18 and country == "india"):
    print ("Eligible for voting")
else:
    print ("Not eligible for voting")
```

Olp : 19
india
Eligible for voting.

3) if else ladder :

→ An if-Else ladder is used when multiple conditions need to be checked one after another.

Syntax :

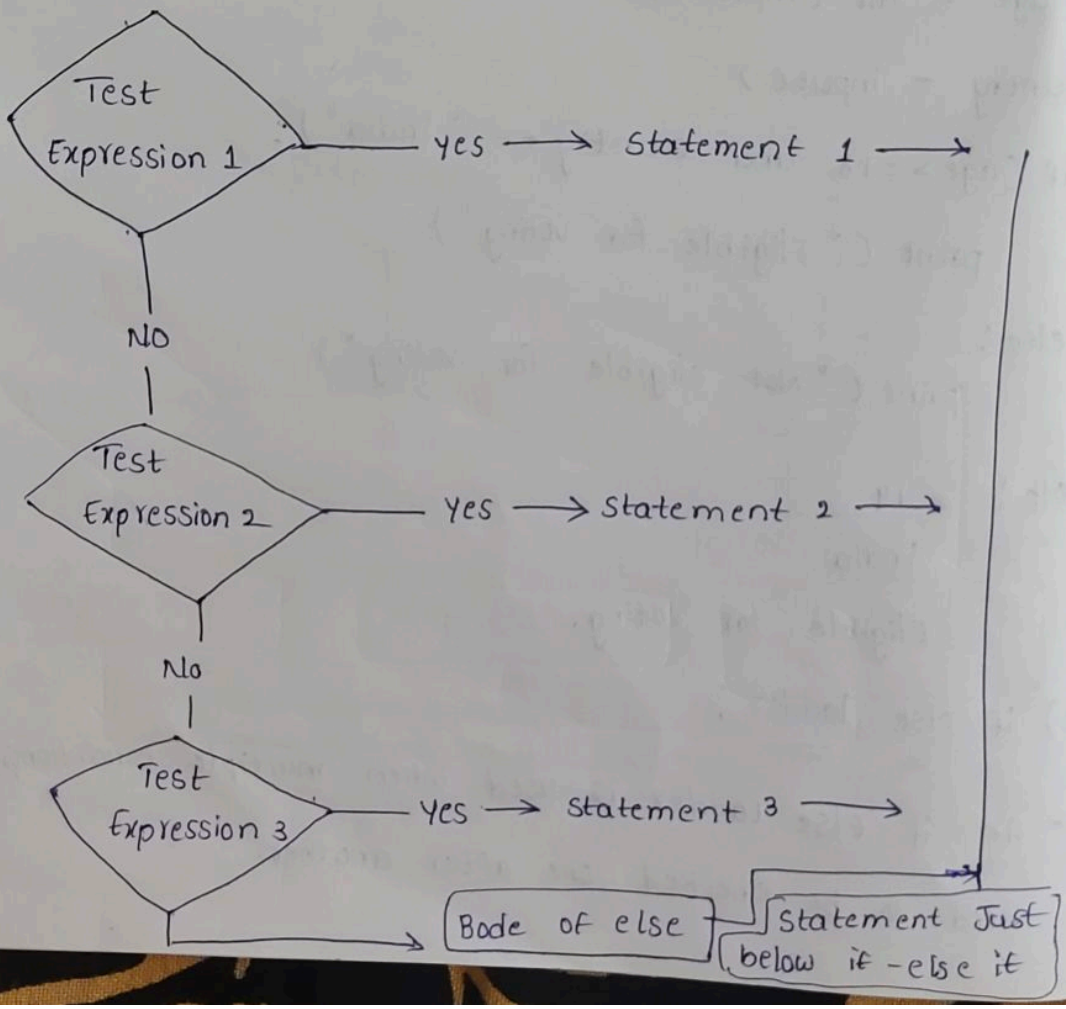
1)

```
if (condition 1):
    Statements of condition 1
else:
    if (condition 2):
        Statements of condition 2
    else:
        if (condition 3):
            Statements of condition 3
        else:
            default statements.
```

2)

```
if (condition 1):
    Statements of condition 1
elif (condition 2):
    Statements of condition 2
elif (condition 3):
    Statement of condition 3
else:
    default conditions.
```

Flow chart of if else ladder statement



program :

1) Write a program to display age group based on their age

i) 1 to 10 — children

ii) 11 to 18 — teenage

iii) 19 to 40 — adult

iv) above 40 — old age.

```
age = int(input("Enter your age:"))
if (age >= 1 and age <= 10):
    print("children")
elif (age >= 11 and age <= 18):
    print("teenage")
clif (age >= 19 and age <= 40):
    print("adult")
elif (age >= 40 and age <= 20):
    print("old age")
else:
    print("Invalid input")
```

Assignment :

1) percentage

i) 85 to 100 = distinction.

ii) 60 to 84 = first class

iii) 50 to 59 = second class

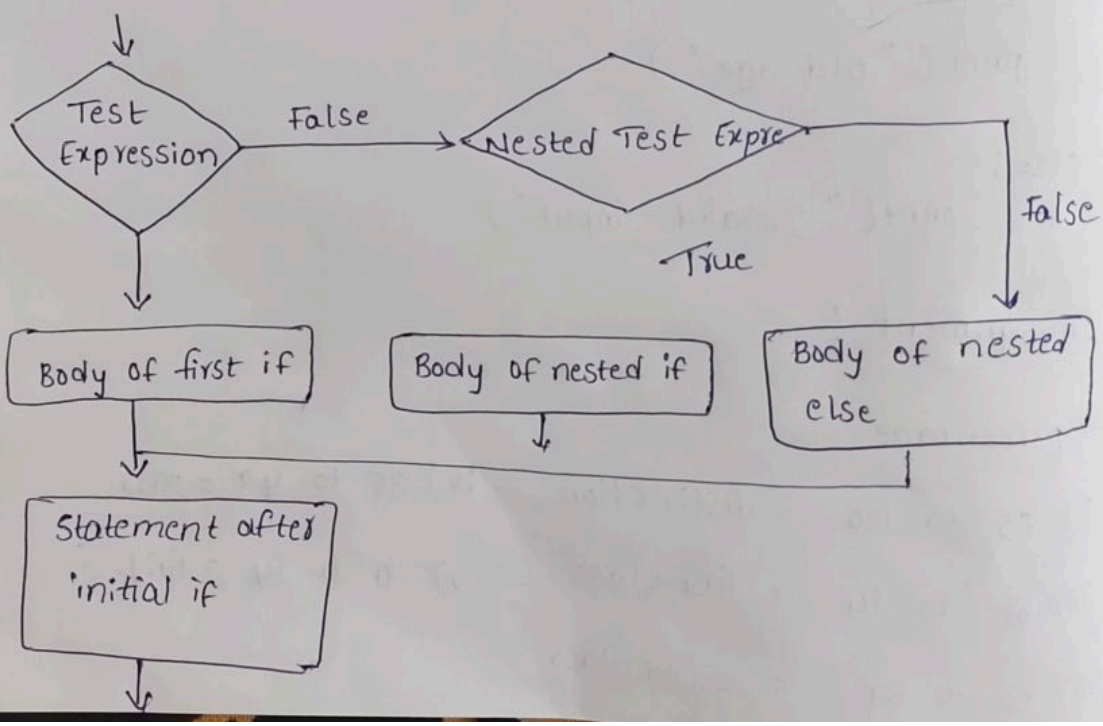
iv) 35 to 49 = pass

v) 0 to 34 = fail.

```
marks = int(input("Enter your marks: "))
if (marks >=85 and marks <=100):
    print("Distinction")
elif (marks >= 60 and marks <=84):
    print("first class")
elif (marks >=50 and marks <=59):
    print("second class")
elif (marks >=35 and marks <=49):
    print("pass")
elif (marks >=0 and marks <=34):
    print("fail")
else:
    print("Invalid input")
```

16/9/25

4) Nested if else :



Definition: To evaluate Multiple Conditions.

Syntax:

```
if (condition 1): # outer if
    if (condition 1): # inner if
        Statements of inner if
    else:
        Statements of inner if else
else:
    Statements of outer if else.
```

Example program:

1) Write a program to find the given number is +ve, -ve zero.

A)
```
num = float(input("Enter any number: "))
if (num>=0):
    if (num>0):
        print("Number is positive")
elif (num<=0):
        print("Number is Negative")
else:
        print("Number is zero").
```

Output:

```
Enter any number : 200
Number is positive.
```

18/09/25

Loops:

→ Loops are used to repeat a block of code multiple times until a condition is met.

→ python mainly provides two types of loops.

1. for
2. While.
3. Nested loop.

1. for loop:

* Used when you know how many times you want to run the code.
* It iterates over a sequence.

Syntax:

```
for variable in sequence:
```

Ex:

① Value = "Besant Tech"

```
for a in value:
    print(a)
```

output:

```
B
e
s
a
n
t

T
e
c
h
```

② for i in range (0, 10, 1): # i = 0, 1, 2, 3, ---- 9

print ("Besant Tech")

for loop has two types

1) for loop with sequence.

2) for loop with range.

1) for loop with sequence :

→ A sequence in python means an ordered collection of items.

Ex: List, tuple, string, range.

Syntax:

```
for var in sequence-name : # var = iterative variable
    Statements
```

Ex:

```
animals = ['monkey', 'lion', 'dog', 'cat', 'rabbit', 'cow']
for i in animals :
    print(f"Hii {i} good afternoon', ;)
```

2) for loop with range :

→ The range() function generates a sequence of numbers, which is commonly used with for loops.

Syntax:

range(Start value, Stop value, Step value).

Ex:

```
value = "Besant Tech"

for i in range (0, len (value )):  # 0, 1, 2, 3 Len value

    print (f" position = {i}, value = { value [i] }" )
```