Explain Programming and Python in de-tail?

Programming is the process of coriting instructions that tells a computer what to do. These instructions help the computer solve problems or performs tasks like calculations, data Storage/ decision-making.

Example:-

A Program to add two numbers.

```
a = 10
b = 20
Print (a+b)
```

What is Python?

Python is a high-level programming language that is easy to learn and understand. It was developed by Guido Van Rossum and is widely used because of its simple ~~System~~ syntax and readability.

Chavecteristics of Python

=> easy to read and write
=> Uses simple english-like syntax
=> Large collection of libraries
=> Free and open source.
=> supports multiple Programming styles.

Applications of Python.

=> Web development
=> Data Analysis

=> Automation

=> scientific research

Example

```
Print ["Hello, world"]
```

Types of comments in python.

Comments are statements in a program that are not executed by Python they are used to explain the coole and improve readability.

1. Single - line comments.

14

=> used to explain a single line of code

=> Begins with the # symbol.

=> python ignores this line during excecution.

Sytax:

```
# This is a single - line comment
```

Example:

```
# Assign value to variable
x = 10
```

2. Multi - line comments

=> used to explain multiple lines at once

=> written using triple quotes (''' or """)

=> commonly used for documentation.

Syntax

```
""" 
This is a
multiple - line comment
"""
```

Ex: ... Average"

Example.

""" This Program calculates sum of two numbers """

```
a = 5
b = 6
Print (a+b)
```

Importance of comments

=> makes code easy to understand
=> Helps in debugging
=> useful for documentation.

Importance of python in modern software Development.

=> It is easy to learn and use
=> simple syntax reduces coding time.
=> supports multiple programming styles
=> Has many libraries and frame works
=> used in AI, data science, web development and automation
=> works on different operating systems.

Data Types and operating systems in Python.

Python provides different built-in data types to store various kinds of data and operators to perform operations on them.

Built-in Data types in python.

1. Numerical Data types

Used to store numbers such as integers and decimal values.

```
a = 10   # integer
b = 3.5  # float
```

2. Sequence Data Types:- Tuple - ordered collection immutable used to store a collection of values.

```
name = "Prasu"  # string
marks = [80,85,90] # list - ordered collection ; mutable.
```

3. Set Data Type.

Stores unique elements

```
nums = {1,2,3}
```

List - []
Tuple - ()
set - {}

4. Mapping Data type.

Stores data in key-values pairs.

```
student = {"name": "Vardhan", "age": 20}
```

5. Boolean Data Type.

Stores True or false values

Result : True

int = a=10, b=-12, c=1234567

Float = x=1.0, y=12.3 ; z=13.6

complex = Alphabet with number ; A=2+5j

String - sequence of charecters reprasented by quotation marks. In python eve can use single, double/triple quotes to define a string.

"hello python"

Type Identification using Type()

The type() function in python is used to identify the data type of of variable. It tells what kind of value is stored in the variable.

Types of Python operators.

Python operators are categorized in the following categorie

1. Arithmetic operators.

Used for mathematical calculations.

Operator	meaning	example.
+	Addition	10 + 5 = 15
-	subtraction	10 - 5 = 5
*	multiplication	10 x 5 = 50
/	Division	10/5 = 2.0
%	modulur [remainder]	10 % 3 = 1
//	Floor Division	10 // 3 = 3
**	Exponent	2** 3 = 8

Real world use:
Calculating total marks, salary, discounts, average score etc.

2. Assignment operators.

Used to assign values to variables

operators	Examples	meaning.
=	x = 10	Assign value
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 2	x = x * 2

ype [variable name

```
a = 10
print [Type a]) # int

b = 3.5
Print [Type (b]) # float
name = "python"

print [Type [name]] # str
nums = [1, 2, 3]
Print [Type [nums]) # list

Flag = True
Print [Type [Flag) # bool
```

operators in python [x + y]

=> used to perform operations on variables and values.

=> python operators are special symbols used to perform specific operations on one or more operators.

unary operators

Python operators that requires one operand to perform a specific operation known as unary operators.

Binary operators.

Python operators that requires two operands toperform a specific operations.

Operands

Variables, Values or expressions that are used with opera to perform a specific operations.

Operator	meaning.
in	value exists
not in	value does not exist

Example - 'a' in apple # True.

Real - world use:

Checking object identify in programs.

Python' Input and out put operations.

Input operations in python.

Input() function

=> used to take input from the user

=> Default data type is String (str)

```
name = input ["Enter your name:")
```

Type conversion while Taking input

Since input is always string, type conversion is required.

```
age = int [input ["enter age:"))
salary = Float [input ["enter salary :"))
```

Taking multiple inputs

multiple inputs can be taken in one line using split()

```
a, b = input ["enter two numbers :"). split ()
a = int (a)
b = int (b)
```

out put operation in python.

print() function

used to display out put on the screen.

```
Print ["Hello. Python")
```

Real world use:
Updating bank balence, increasing counters, modifying scor

3. comparingsion operators.
used to compare two values. Result is True/False.

operator	Meaning.
= =	equal
! =	Not equial
>	Greater than
<	less than.
>=	Greater than or equal
<=	Less than or equal.

Real-world use:
cheaking pass/fail marks, age eligibility, login validation.

4 Logical operators.
used to combine conditions.

operators	meaning.
and	True if both conditions are true
or	True if any one condition is true
not	Reverses the result.

Real-world use:
checking multiple conditions like age and quantification

Membership operators
used to check conditions like age and avg wheather a value exists in a sequence.

separator (SCP)

used to Separate multiple values.

```
Print [10, 20, 30, sep = " , ")
```

Output

10, 20, 30

Format specifiers | Formatted output

Using format ()

```
Print ["My age is { }". Format (age))
```

Using F - string

```
Print [F"My salary is {salary}")
```

Real-world use:

Displaying seperts, student details, invoices, bills.

control statements and Décision-making statements.

meaning of control statements

They control the flow of exeantiman of a program.

They decide which statement to eceants and when.

Importance

=> enables decision making

=> marks programs logical and dynamic

=> Avoids unnecessary execntion.

Types of control statements.

1. Decision-making statements.
2. Looping statements.
3. Jump statements.

Decission-making statements

1. if statement

executes a block of code only if the condition is true.

Syntax

```
if condition:
    Statement
```

Example

```
if marks >=35:
    Print ["Pass")
```

2. If-else statement

executes on block if Condition is true, otherwise excutes another block.

Sytax

```
if condition:
    Statement 1
else
    Statement 2
```

Example

```
if age >=18
    Print ("eligible to vote")
else
    Print ("Not eligible")
```

3. If-elif-else statement

used to check multiple conditions

Syntax

```
    if condition 1:
        statement 1:
· elif
      condition 2:
      statement 2
```

```
else
    statement 3
```

Example.

```
if marks >=75;
    Print ["Distinction")
elif marks>=60
    Print("First class")
elif marks >=35
    Print ("Pass")
else
    Print ("Fail")
```

Execution Flow

=> conditions are checked top to bottom

=> first true condition is executed.

=> Remaining conditions are skipped

Example (IF-else execution flow)

1. Condition age >=18 is checked.
2. condition is False
3. else block executes
4. Program continues after if-else.

Python executes only the block whose condition is true.

indentation plays a crucial role in controlling flow.

write an essay on python Programming Fundamentals.

1. Role of Programming in Problem solving.

=> Programming helps in solving real-world problems using logical steps.

=> if breaks complex problems into smaller,

Managable tasks.

=> If breaks complex problems into smaller, managable task

=> Enables automation of repetitive work.

=> Helps in accurate calculations and data processing.

=> Improves efficiency and saves time.

2. Python syntax simplicity and Readability.

=> Python has simple & english - like syntax

=> easy to learn for beginners.

=> Does not use braces { } or semi colons ;

=> Uses indentation to define code blocks

=> Programs are easily to read, write & maintain.

3. Use of comments For code Documentation.

=> Comments explain the purpose of code statements.

=> Improve readability and understanding of programs.

=> Helpful for developing and future modifications.

=> Single - line comments use #

=> multi - line comments use triple qvote [''' or """]

=> comments are not executed by the interpreter.

4. Data types, operators and input/output operations.

Python supports built - in data type like

int, float, str and bool.

=> operators perform arithmetic, comparision, logical and assignment operations.

=> input () function is used to take input from the user.

=> Default data type of input () is "string"

=> print() function is used to display output
=> supports Formatted output for better presentation

5. control flow using Decision-making statements.

=> Control Flow decides the order of execution in a Program.
=> Decision-making statements help in making logical choices.
=> if statement executes code when condition is true.
=> if else provides two-say decision making
=> if elif-else handless multiple conditions.
=> makes - Programs Flexible and dynamic.

Conclusion.

=> Python fundamental form the base for advanced programming
=> simple syntax and powerful features makes Python popular.
=> Widely used in education, science, data analysis and software development.

① Movie Ticket pricing

A movie theatre charges;

₹150 for children [age < 13)
₹250 for adults [age 13-59)
₹200 for seniors [age ≥ 60)

If the person is watching a 3D movie, add ₹250 extra
Write Program that takes age and is 3D [1 or 0) and Prints the final ticket price.

```
age = int(input("Enter age: "))
is_3d = int(input("is it a 3D movie? (1 for yes, 0 for NO): "))
if
    age < 13
    Price = 150
```

```
elif age <= 59:
    Price = 250
else
    price = 200
if is 3D = 1
    Price += 50
Print ("Final Ticket Price: ₹", Price)
```

2. College Attendance Rule: A student is allowed to write the exam if;
Attendance ≥ 75
or
attendance ≥ 60 and has medical certificate (1=yes, 0=no)
Take attendence percentage and medical certificate as input and print "Allowed or "Not Allowed".

```
attendance = int (input ("enter attendence percentage:"))
medical = int (input ("medical certificate (1 for yes, 0 for no):"))
if attendance >= 75 or (attendance >= 60 and medical == 1):
    Print ("Allowed")
else:
    print ("Not allowed")
```

3. E-comerce Discount
A shopping site gives
20% discount if bill = 5000
20% discount if bill between 2000 & 4999
20% discount if bill < 2000
But if the customer is a prime number they get extra

5% discount.

Input: bill amount; is prime (1 or 0)

Print final amount to be paid

```
bill = float(input("Enter bill amount: "))
if bill >= 5000:
    discount = 0.20
elif bill >= 2000:
    discount = 0.10
else
    discount = 0.0
i.e prime = int(input("is the customer a prime member? (1
        for yes, 0 for NO): "))
if is prime == 1:
    discount += 0.05
Final amount = bill - (bill * discount)
print("Final amount to be paid: ₹", final-amount).
```

4 Smartphone Battery warning.

A phone shows:

"Low Battery" if ~~between~~ battery ≤ 20

"normal" if battery between 21-80

"full" if battery > 80

But if phone is charging, it should show "charging" instead of any message.

Input: battery percentage, is charging (1 or 0)

```
battery = int(input("enter battery percentage: "))
```

```
is charging = int (input ("Is the device charging? (1 for yes, 0
                for NO):"))
if is charging == 1 :
    Print ("charging")

else
    if battery <= 20:
        Print ("Low Battery")
```

4 Driving License check

A person can get a driving License if :

age ≥ 18

AND

Passed driving test (1 = yes)

But If age ≥ 60, driving test is not required

Input age, test passed

Print "eligible" for) "Not eligible"

```
age = int (input ("enter age:"))
test passed = int (input ("Passed driving test? (1 for yes, 0 for NO):"))
if age >= 60:
    Print ("eligible")
elif age >= 18 and test passed == 1 :
    Print ("eligible")
else
    print ("not eligible")
```

5 Online Food delivery

A restaurant gives free delivery if :

order amount ≥ 500

OR

user is a gold member

But if the distance is more than 10 km, delivery is never free.

Input: amount, is Gold [1 or 0], distance

```
amount = float (input ("enter order amount: "))
is Gold = int (input ("is Gold member? (1 for yes, 0 for NO): "))
distance = float (input ("enter delivery distance (km): "))

if distance > 10:
    Print ("Delivery charged")

elif amount >= 500 or is Gold == 1;
    Print ("free Delivery")

else:
    Print ("Delivery charged")
```

6 Bank loan Approval

A bank Approves loan if:

Salary ≥ 30,000 And credit score ≥ 700

or

Salary ≥ 50,000 (credit score ignored)

input: salary, credit score

Print "loan Approved" or Loan Rejected".

```
Salary = int (input ("enter salary: "))
Credit score = int (input ("enter credit score: "))
if salary >= 50000 or (salary >= 30000 and creditscore >= 700)

else:
    print ("Loan Rejected")
```

7 electricity Bill

units consumed

First 100 units ⇒ ₹ 2/unit

Next 100 units = ₹ 3/unit
Above 200 units = ₹5 /unit
Note: No loops
Print final bill amount

```
units = int (input ("enter units consumed: "))
if units <= 100:
    bill = units * 2
elif units <= 200
    bill = (100*2) + (units - 100) * 3
else bill = (100*2) + (100*3) + (units - 200) * 5
print ("electricity Bill Amount: ₹", bill)
```

8 Student scholorship
A student gets a sdolorship if:
marks ≥ 85
AND
Family income < 500000
But if the student is a single parent child, income condition is ignored.
Input: marks, income, single parent (1 or 0)

```
marks = int (input ("enter marks: "))
income = int (input ("enter family income: "))
single parent = int (input "single parent child? (1 for yes, 0 for No): "))
if marks >= 85 and (income < 500000 or single parent == 1):
    Print ("scholorship Granted")
else
    Print ("scholarship Not Granted")
```

9 online exam Result

A student passes if :

theory ≥ 40 AND practical ≥ 40

But if total (theory + practical) ≥ 100, pass even if one is less than 40

```
Input : theory, practical
Theory = int (input ("enter theory marks :"))
Practical = int (input ("enter practical marks : "))
if (theory > = 40 and practical >= 40) or theory + practical
    > = 100):
  print ("pass")
else print ("fail")
```

10 Hotel Room Pricing

A hotel charges

₹ 3000 per day for normal days

₹ 4000 per day on weekends

if customer stays more than 3 days, give 15% discount

input : is weekend (1 or 0), days stayed.

Print final bill.

```
is weekened = int(input (" is a weekend stay ? (1 for yes, 0
For No): "))
days stayed = int (input ("enter number of days stayed : "))
if is weekend = =1:
        rate = 4000

else rate = 3000
  bill = rate * days stayed
```

if days stand > 3:

bill = bill * 0.85

Print ("Total bill amount : ₹", bill)

11 Gronining Gaming Level unlock

a game unlocks next level if :

score ≥ 100

or

Player has a premimum pass

But if player used cheating, acus is denied

input : score, is premimum, used cheat

score = int (input ("enter score : "))

is premimum = int (input ("Is premimum player ? (1 for yes, 0 for No): "))

if used cheat == 1 :

Print ("Acess Denied")

elif score >= 100 or is premium == 1 :

Print ("Next Level enlocked")

else print ("Level Locked")

12. mobile data usage

A Network gives unlimited data if :

dailly usage < 2GB :

or

user has unlimited data if :

But if roaming is on, ullimited Plan does not work

Input: data used, Has unlimited plan, is Roaming.

```
data used = float(input("enter daily data usage (GB): "))
has unlimited plan = int(input("Has unlimited plan? (1 for yes,
                    0 for no): "))
is Roaming = int(input("is roaming on? (1 for yes, 0 for NO): "
if is Romaing == 1:
    if data used <= 2:
        Print("Unlimited Data")
    else:
        Print("Limited Data")
else
    if data used <= 2 or has Unlimited Plan == 1:
        Print("Unlimited Data")
    else:
        Print("Limited Data")
```

13 Office entry system

An employee can entes the office if:

ID card is valid.

AND

(Fingerprint matches OR face scan matches)

But if it is a holiday, entry is denied for everyone.

Input: id valid, Fingerprint, face scan, is Holiday

```
id valid = int(input("ID card valid? (1 for yes, 0 for No: "))
Finger Print = int(input("finger print match? (1 for yes, 0 for NO)
                    : "))
Face scan = int(input("face scan match? (1 for yes, 0 for NO)
                    : "))
is Holiday = int(input("is today a holiday? (1 for yes, 0 for NO): "))
```

```
if is Holiday == 1
  Print ("entry Denied")
elif idvalid == 1 and (Finger print == 1 or face scan == 1):
    Print ("entry Allowed")
else
  print("entry Denied")
```

14 Movie Rating Display

A movie app shows rating based on average score:

Average $\geq$ 8.5 $\rightarrow$ "excellent"

Average between 6.0 and 8.4 $\rightarrow$ "Good"

Average $<$ 6.0 $\rightarrow$ "Average"

But if the movie is marked as editor's choice, always show "Recommended".

Input : average Rating, is editors choice (1 or 0)

Point the message

```
average Rating = float (input ("enter average rating : "))
is editors choice = int (input ("is editor's choice ? (1 for yes, 0 for NO) : "))
if is editors choice == 1:
    print ("Recommended")
elif average Rating >= 8.5:
    Print ("Excellent")
elif average Rating >= 6.0:
    Print ("Good")
else:
    Print ("Average")
```