What are Interpreted Programming Languages?

- No Compilation: Instead of being translated into machine code all at once before execution (like compiled languages), interpreted languages are translated line by line as the program runs.
- The Interpreter: This is a special program that acts as a translator. It reads each line of your code and immediately executes the instructions.
- Flexibility: You can make changes to the code and see the results quickly without having to recompile the entire program.

Key Points for Exam Notes

- Examples:
  - Python
  - JavaScript
  - Ruby
  - PHP
  - BASIC
- Advantages:
  - Easy to learn and use: Great for beginners.
  - Great for prototyping and testing: Make quick changes, see immediate results.
  - Platform independence: Often run on different operating systems without changes (if the interpreter is available).
- Disadvantages:
  - Slower execution: Translation at runtime can be slower than compiled code.
  - Dependency on interpreter: The interpreter program needs to be installed on the machine to run the code.

Exam Note Tip:

Remember the phrase “Interpreted = Line-by-line translation”

Related Posts:

1. What are General-Purpose Programming Languages?
2. What is a variable in Python?
3. Does Python have double, short long data types
4. What are High-Level Programming Languages?
5. Download Python
6. How to run a Python Program
7. Python program to find GCD of two numbers
8. Python Program to find the square root of a number by Newton’s Method
9. Python program to find the exponentiation of a number
10. Python Program to find the maximum from a list of numbers
11. Python Program to perform Linear Search
12. Python Program to perform binary search
13. Python Program to perform selection sort
14. Python Program to perform insertion sort
15. Python program to find first n prime numbers
16. Python program Merge sort
17. NumPy
18. Python library
19. Python Installation and setup
20. Python Variables
21. Python Data Types
22. Python lists

23. Python Creating and Accessing List
24. Python List Manipulation
25. Python Input function
26. Python list slicing
27. Python Class and Object
28. Python find the output programs
29. Python Introduction
30. Python basic syntax
31. Python int data type
32. Python float data type
33. Understanding Floating-Point Precision in Python: Avoiding Numerical Computation Errors
34. How to search Python library using command line tool
35. Which python libraries are used to load the dataset ?
36. Why is there no need to mark an int float in a variable in Python ?
37. Python Intro: Top 20+ Questions and Answers for Beginners