Basic Python

1. What is Python, and why is it popular?
   ->Python is a programming language that is easy to understand and write, making it a popular choice among developers and programmers. It is compatible with various programming paradigms, such as procedural, object-oriented, and functional programming. Python has a large and active community of developers, an extensive standard library, and third-party packages for various domains.

2. What is an interpreter in Python?

->An interpreter in Python is a program that reads and executes Python code line by line. It takes the human-readable Python code (source code) and converts it into machine code that the computer can understand and execute. Unlike compilers (which translate the entire code into machine code before running it), an interpreter processes code line by line, executing it immediately.

3. What are pre-defined keywords in Python?
   ->Keywords are reserved words in Python that have special meanings and cannot be used as identifiers (variable names, function names, etc.). Examples of keywords
   include if, else, while, for, def, class, import, try, except, return, True, False, None, etc.Keywords are case-sensitive and cannot be redefined or overridden within a Python program.

4. Can keywords be used as variable name?

->No, keywords in Python cannot be used as variable names. Keywords are reserved words that have a specific meaning in the language, and they are part of the Python syntax. Trying to use a keyword as a variable name will result in a syntax error.

5. What is mutability in Python?

->In Python, objects can be either changeable (modifiable) or unchangeable (unchangeable). Mutable objects, such as lists and dictionaries, can be modified after creation, while immutable objects, such as tuples and strings, cannot. Mutability affects how objects are stored and manipulated in memory, impacting performance, memory usage, and concurrency in Python programs.

6. Why are lists mutable, but tuples are immutables?

->Lists in Python are mutable, which means that you can modify them after they are created. You can change their content by adding, removing, or updating elements.

Tuples, on the other hand, are immutable, which means once a tuple is created, its elements cannot be changed, added, or removed. Attempting to modify a tuple will result in a `TypeError`.

7. What is the difference between "==" and "is" operators in Python?
   ->n Python, both == and `is` are comparison operators, but they are used to compare different aspects of objects.
   The `is` operator checks whether two objects refer to the same memory location (i.e., they are the same object). It compares object identity, not the values inside the objects.
   The == operator checks whether the values of two objects are equal. It compares the content or value of the objects, regardless of whether they are the same object in memory.

8. What are logical operators in Python?

-> Operators are special symbols or keywords that are used to carry out specific actions on numbers or variables in Python expressions.

Python supports various types of operators, including arithmetic operators (+, -, *, /), comparison operators (==, !=, <, >), logical operators (and, or, not), assignment operators (=, +=, -=, *=, /=), etc.

Operators have precedence and associativity rules that determine the order of evaluation in expressions.

9. What is type casting in Python?

- Type casting, also referred to as type conversion, is the process of changing one data type to another in Python.
- Python provides built-in functions for type casting, such as int(), float(), str(), list(), tuple(), dict(), etc.
- Type casting is often necessary for performing arithmetic operations, data manipulation, and input/output operations in Python programs.

10. What is the difference between implicit and explicit type casting?
    -> Implicit Type Casting: Python automatically converts smaller types (e.g., integers) to larger types (e.g., floats) during operations to ensure no data loss.
    Explicit Type Casting: The programmer explicitly converts one data type to another, using built-in functions like `int()`, `float()`, or `str()`.

11. What is the purpose of conditional statements in Python?

->Purpose: Conditional statements allow the program to make decisions based on conditions and control the flow of execution.

Types:

- `if`: Executes code if the condition is `True`.
- `else`: Executes code if the condition is `False`.
- `elif`: Checks additional conditions if the `if` statement is `False`.
- Conditional statements make programs dynamic and able to adapt to different inputs and situations, making them an essential part of programming.

12. How does the elif statement work?

-> The `elif` statement follows an `if` statement. If the `if` condition evaluates to False, the program checks the next condition in the `elif` block.You can have multiple `elif` conditions, and Python will check them one by one, in the order they appear. The first condition that evaluates to True will execute its corresponding block of code, and the rest of the `elif` and `else` blocks will be skipped. If none of the `if` or `elif` conditions are `True`, the `else` block (if present) will execute.

13. What is the difference between for and while loop?

- ->n Python, loops are employed to repeat a sequence of actions or code until a specific condition is fulfilled.
- Python offers two primary types of loops: for loops and while loops.
- "for" loops are used for iterating over a sequence of elements, while "while" loops are used for executing code until a specified condition becomes False

14. Describe a scenario where a while loop is more suitable than a for loop?

➔ `while` loop is more suitable than a `for` loop in situations where you don't know in advance how many times the loop should run, but you need the loop to continue running as long as a specific condition is True.