Python Interview Questions

1. What is python?
    - Python is high level programming language and interpreter language and general-purpose language.

2. What is b/t interpreter and compiler?
    - The interpreter executes the code line by line in case if their error occurs it is not executing the other line of the code.
    - The compiler executes the hole code at a time incase if there are any errors that occur it skips that line and executes the other line.

3. Is Python interpreter language? If so, explain?
    - Yes, the python is interpreter language because it executes the code line by line.

4. What is datatype? What are datatypes in python?
    - A variable is a container used to store values, and the kind of value it holds is defined by its data type.
    - Primitive Data Types: int,float,str,bool,None.
    - Non-Primitive Data Types: list , tuple, dict, set.

5. What is list? Give an example of that?
    - List is collection of sequence data, and it is denoted by "[]".
    - Ex: list = ["33","55","99","88","77"]
      print(list)
      print(type(list)

6. What is a tuple and example of that?
    - The tuple is a collection of sequence of data, and it is denoted with the "()".
    - It is immutable.
    - Ex: tuple = ("str","are","the","is","A","and")
      print(tuple)
      print(type(tuple))

7. What is dict and give example of that?
    - It is built in datatype the python is used to store key: value :pairs
    - Ex: student = {

```
"name": "Sravani",
"age": 22,
"branch": "MCA",
"is_graduated": False
}
```

8. Difference Between Mutable and Immutable Data Types?
    - A mutable data type can be changed after it is created, and we can modify the data.
    - An immutable data type cannot be changed once it is created.

9. Difference Between List and Tuple?
    - List: list is mutable, denoted with the “[ ]”.
    - Tuple: tuple is immutable, denoted with the “( )”.

10. How do you modify a list?
    - We can modify the list by using
    - append() → Add item at end
    - insert() → Add item at a specific index
    - extend() → Add multiple items from another list
    - remove() → Removes first matching value
    - pop() → Removes item at index (default: last)
    - clear() → Empties the entire list
    - sort()
    - reverse()

11. Difference between append, Insert, pop and pop(index)?
    - append() → Add item at end
    - insert() → Add item at a specific index
    - pop() → Removes item at index (default: last)
    - pop(i) → Removes item at given index.

12. How to Update a Dictionary in Python?
    - Change the Value of an Existing Key
      Ex: student = {"name": "Srav", "age": 20}
      ```
      student["age"] = 21
      print(student)   # {'name': 'Srav', 'age': 21}
      ```
    - Add a New Key-Value Pair
      Ex: student["college"] = "KL University"

```
print(student)
# {'name': 'Srii', 'age': 21, 'college': 'KL University'}
```

- Using update() Method

```
Ex: student.update({"age": 22, "branch": "CSE"})
    print(student)
# {'name': 'Srii', 'age': 22, 'college': 'KL University', 'branch': 'CSE'}
```

- Using Variables with update()

```
Ex: new_data = {"year": 3, "status": "Active"}
    student.update(new_data)
```

13. Example of a Nested Dictionary for Electronics?

- Ex:

```
electronics = {
  "Laptop": {
    "Brand": "Dell",
    "Model": "Inspiron 15",
    "Price": 58000,
    "Specs": {
      "RAM": "16GB",
      "Storage": "512GB SSD",
      "Processor": "Intel i5"
    }
  },
  "Mobile": {
    "Brand": "Samsung",
    "Model": "Galaxy A55",
    "Price": 32000,
    "Specs": {
      "RAM": "8GB",
      "Storage": "128GB",
      "Camera": "64MP"
    }
  },
  "Smartwatch": {
    "Brand": "boAt",
    "Model": "Storm Call",
```

```
        "Price": 2499,
        "Specs": {
          "Battery": "7 days",
          "Water Resistant": True,
          "Bluetooth": "v5.0"
        }
      }
    }
    Print(electronics)
```

14. Write a List of dictionaries?

Ex:

```
    students = [
      {
        "name": "Sriram",
        "roll_no": 101,
        "marks": 92
      },
      {
        "name": "Alisha",
        "roll_no": 102,
        "marks": 87
      },
      {
        "name": "Ravi",
        "roll_no": 103,
        "marks": 76
      }
    ]
    Print(student)
```

15. What is an operator?

- An operator is a special symbol or keyword that is used to perform operations on values or variables.

Types of operators:

- Arithmetic (math) → (+, -, *, /, %)
- Comparison (true/false) → (==, !=, > , <)

- Assignment (store value) → (=, +=, -=, etc.)
- Logical decisions (and/or) → (and, or, not)
- Bit-level operations → Topic is pending…..
- Membership/identity checks → (in, not in)

16. What are Arithmetic Operators? Why are They Used?

The arithmetic operators:

- Addition (+) → Adds two values
- Subtraction (-) → Subtracts second from first
- Multiplication (*) → Multiplies two values
- Division (/) → Divides and gives float
- Exponentiation (**) → Raises to the power

Ex: a = 15
b = 4

```
print(a + b)    # 19
print(a - b)    # 11
print(a * b)    # 60
print(a / b)    # 3.75
print(a // b)   # 3
print(a % b)    # 3
print(a ** b)   # 50625
```

17. What are Comparison Operators? Why are they used?

Comparison Operators:

- Equal to (==) → Returns True if both values are equal
- Not equal to (!=) → Returns True if values are not equal
- Greater than (>) → Returns True if the left value is greater
- Less than (<) → Returns True if the left value is smaller
- Greater than or equal to (>=) → Returns True if left value is greater or equal
- Less than or equal to (<=) → Returns True if left value is smaller or equal

Ex: a = 10
b = 20

```
print(a == b)   # False
print(a != b)   # True
print(a > b)    # False
print(a < b)    # True
```

```
print(a >= 10)  # True
print(b <= 15)  # False
```