

# PYTHON

---

## INTERVIEW QUESTIONS

---

with Answers to Crack

**TECHNICAL INTERVIEW**





## \*Disclaimer\*

**Everyone learns uniquely.**

Master Python concepts by through these  
important interview questions.

Take the help of this doc, and crack your Python  
rounds like a pro.

# TABLE OF CONTENTS

1. Basic data types in Python	4-5
2. OOPS concept in Python	6-7
3. String handling functions	8-9
4. Control statements, functions in Python	10-11
5. Special data types in Python	12-13
6. Lambda functions, list comprehension	14-16
7. Libraries used for data science: Pandas, NumPy, Seaborn, Matplotlib	17-19
8. Types of plots in Seaborn and Matplotlib and their uses	20-22
9. Library for machine learning: Scikit-learn	23-25

# Topic 1

## Basic data types in Python:

### Q.1 What are the basic data types in Python?

Ans: The basic data types in Python are:

- **Integer:** represents whole numbers.
- **Float:** represents decimal numbers.
- **String:** represents a sequence of characters.
- **Boolean:** represents either True or False.
- **List:** represents an ordered collection of elements.

### Q.2 How do you convert a string to an integer in Python?

Ans: You can use the `int()` function to convert a string to an integer. For example:

```
num_str = "10"
num_int = int(num_str)
```

## Q.3 How do you check the data type of a variable in Python?

Ans: You can use the '**type()**' function to check the data type of a variable. For example:

```
● ● ●  
num = 10  
print(type(num)) # Output: <class 'int'>
```

## Q.4 What is the difference between a list and a tuple in Python?

Ans: A list is mutable, which means you can modify its elements, while a tuple is immutable, meaning its elements cannot be changed after creation.

## Q.5 How do you create an empty dictionary in Python?

Ans: You can create an empty dictionary using either the curly braces **{}** or the **dict()** function. For example:

```
● ● ●  
empty_dict = {}  
empty_dict = dict()
```

## Topic 2

### OOPS concept in Python:

#### Q.1 What is OOPS and how is it implemented in Python?

Ans: Object-Oriented Programming (OOPS) is a programming paradigm that uses objects to represent real-world entities. In Python, OOPS is implemented through classes and objects. Classes are blueprints for creating objects, and objects are instances of a class.

#### Q.2 What are the four principles of OOPS?

Ans: The four principles of OOPS are:

- **Encapsulation:** bundling of data and methods that operate on that data within a single unit (class).
- **Inheritance:** ability of a class to inherit properties and methods from its parent class.
- **Polymorphism:** ability of an object to take on different forms or behaviors based on the context.
- **Abstraction:** representing essential features and hiding unnecessary details to simplify the complexity.

### **Q.3 What is method overloading in Python?**

Ans: Method overloading in Python refers to defining multiple methods with the same name but different parameters within a class. However, Python does not support method overloading by default as it does in languages like Java. In Python, you can achieve a similar effect by using default arguments or using variable-length arguments.

### **Q.4 What is method overriding in Python?**

Ans: Method overriding in Python refers to defining a method in a child class that already exists in its parent class with the same name and signature. The method in the child class overrides the method in the parent class, providing a different implementation.

### **Q.5 What is the difference between a class method and an instance method in Python?**

Ans: A class method is a method bound to the class and not the instance of the class. It is defined using the **@classmethod** decorator and can access only class-level variables. On the other hand, an instance method is bound to the instance of the class and can access both instance and class-level variables.

# Topic 3

## String handling functions:

### Q.1 How do you concatenate two strings in Python?

Ans: You can concatenate two strings using the `+` operator. For example:

```
● ● ●  
str1 = "Hello"  
str2 = "World"  
result = str1 + str2 # Output: "HelloWorld"
```

### Q.2 How do you find the length of a string in Python?

Ans: You can use the `len()` function to find the length of a string. For example:

```
● ● ●  
str1 = "Hello World"  
length = len(str1) # Output: 11
```

### Q.3 How do you convert a string to uppercase in Python?

Ans: You can use the `upper()` method to convert a string to uppercase. For example:

```
● ● ●  
str1 = "hello"  
uppercase_str = str1.upper() # Output: "HELLO"
```

### Q.4 How do you split a string into a list of substrings in Python?

Ans: You can use the `split()` method to split a string into a list of substrings based on a delimiter. For example:

```
● ● ●  
str1 = "Hello,World"  
substrings = str1.split(",")  
# Output: ["Hello", "World"]
```

### Q.5 How do you check if a string contains a specific substring in Python?

Ans: You can use the `in` keyword to check if a substring is present in a string. For example:

```
● ● ●  
str1 = "Hello World"  
is_present = "World" in str1 # Output: True
```

## Topic 4

# Control statements, functions in Python:

### Q.1 What are control statements in Python?

Ans: Control statements are used to control the flow of execution in a program. Common control statements in Python include if-else, for loops, while loops, and break/continue statements.

### Q.2 How do you write an if-else statement in Python?

Ans: An if-else statement in Python is written using the following syntax:

```
● ● ●  
if condition:  
    # Code block executed if the condition is  
    True  
else:  
    # Code block executed if the condition is  
    False
```

### Q.3 How do you define a function in Python?

Ans: A function in Python is defined using the **def** keyword. For example:

```
• • •  
def greet():  
    print("Hello, world!")
```

### Q.4 How do you pass arguments to a function in Python?

Ans: You can pass arguments to a function by including them inside the parentheses when defining the function. For example:

```
• • •  
def greet(name):  
    print("Hello, " + name + "!")
```

### Q.5 How do you return a value from a function in Python?

Ans: You can use the **return** keyword to return a value from a function. For example:

```
• • •  
def add(a, b):  
    return a + b
```

# Topic 5

## Special data types in Python:

### Q.1 What is a set in Python?

Ans: A set in Python is an unordered collection of unique elements. It is defined using curly braces {} or the **set()** constructor. For example:

```
● ● ●  
my_set = {1, 2, 3} # Output: {1, 2, 3}
```

### Q.2 What is a dictionary in Python?

Ans: A dictionary in Python is an unordered collection of key-value pairs. It is defined using curly braces {} or the **dict()** constructor. For example:

```
● ● ●  
my_dict = {"name": "John", "age": 25}  
# Output: {"name": "John", "age": 25}
```

## Q.3 How do you access values in a dictionary in Python?

Ans: You can access values in a dictionary by using the corresponding key. For example:

```
● ● ●  
my_dict = {"name": "John", "age": 25}  
print(my_dict["name"]) # Output: "John"
```

## Q.4 What is a tuple in Python?

Ans: A tuple in Python is an ordered and immutable collection of elements. It is defined using parentheses () or the **tuple()** constructor. For example:

```
● ● ●  
my_tuple = (1, 2, 3) # Output: (1, 2, 3)
```

## Q.5 How do you swap the values of two variables in Python?

Ans: You can swap the values of two variables using a temporary variable or simultaneous assignment. For example:

```
● ● ●  
a = 5  
b = 10  
a, b = b, a  
print(a, b) # Output: 10, 5
```

# Topic 6

## Lambda functions, list comprehension:

### Q.1 What is a lambda function in Python?

Ans: A lambda function is an anonymous function defined using the **lambda** keyword. It is typically used for short, one-line functions. For example:

```
● ● ●  
square = lambda x: x**2  
print(square(3)) # Output: 9
```

### Q.2 What is list comprehension in Python?

Ans: List comprehension is a concise way to create lists in Python based on existing lists or other iterables. It combines the creation of a new list with a loop and optional conditional statements. For example:

```
● ● ●  
numbers = [1, 2, 3, 4, 5]  
squared_numbers = [x**2 for x in numbers]  
print(squared_numbers) # Output: [1, 4, 9, 16,  
25]
```

### Q.3 How do you filter elements in a list using list comprehension?

Ans: You can filter elements in a list using list comprehension by adding a conditional statement. For example, to filter even numbers:

```
● ● ●  
numbers = [1, 2, 3, 4, 5]  
even_numbers = [x for x in numbers if x % 2 ==  
0]  
print(even_numbers) # Output: [2, 4]
```

### Q.4 Can you have multiple if conditions in list comprehension?

Ans: Yes, you can have multiple if conditions in list comprehension by chaining them using the and or or operators. For example:

```
● ● ●  
numbers = [1, 2, 3, 4, 5]  
filtered_numbers = [x for x in numbers if x % 2  
== 0 and x > 2]  
print(filtered_numbers) # Output: [4]
```

## Q.5 How do you create a dictionary using list comprehension in Python?

Ans: You can create a dictionary using list comprehension by specifying key-value pairs within curly braces {}. For example:

```
● ● ●  
keys = ['a', 'b', 'c']  
values = [1, 2, 3]  
my_dict = {k: v for k, v in zip(keys, values)}  
print(my_dict) # Output: {'a': 1, 'b': 2, 'c': 3}
```

# Topic 7

## Libraries used for data science: Pandas, NumPy, Seaborn, Matplotlib

### Q.1 What is a lambda function in Python?

Ans: A lambda function is an anonymous function defined using the `lambda` keyword. It is typically used for short, one-line functions. For example:

```
• • •  
square = lambda x: x**2  
print(square(3)) # Output: 9
```

### Q.2 What is list comprehension in Python?

Ans: List comprehension is a concise way to create lists in Python based on existing lists or other iterables. It combines the creation of a new list with a loop and optional conditional statements. For example:



```
numbers = [1, 2, 3, 4, 5]
squared_numbers = [x**2 for x in numbers]
print(squared_numbers) # Output: [1, 4, 9, 16,
25]
```

### Q.3 How do you filter elements in a list using list comprehension?

Ans: You can filter elements in a list using list comprehension by adding a conditional statement. For example, to filter even numbers:



```
numbers = [1, 2, 3, 4, 5]
even_numbers = [x for x in numbers if x % 2 ==
0]
print(even_numbers) # Output: [2, 4]
```

### Q.4 Can you have multiple if conditions in list comprehension?

Ans: Yes, you can have multiple if conditions in list comprehension by chaining them using the and or or operators. For example:

```
● ● ●  
numbers = [1, 2, 3, 4, 5]  
filtered_numbers = [x for x in numbers if x % 2  
== 0 and x > 2]  
print(filtered_numbers) # Output: [4]
```

## Q.5 How do you create a scatter plot using Seaborn?

Ans: You can create a scatter plot using Seaborn's **scatterplot()** function, specifying the x and y variables from your dataset. For example:

```
● ● ●  
import seaborn as sns  
import pandas as pd  
  
df = pd.read_csv('data.csv')  
sns.scatterplot(x='x_column', y='y_column',  
data=df)
```

# Topic 8

## Types of plots in Seaborn and Matplotlib and their uses:

### Q.1 What are some commonly used plots in Seaborn and Matplotlib?

Ans: Some commonly used plots in Seaborn and Matplotlib include:

- **Line plot:** shows the trend of a variable over time.
- **Scatter plot:** displays the relationship between two variables.
- **Bar plot:** compares categories or groups using rectangular bars.
- **Histogram:** visualizes the distribution of a continuous variable.
- **Box plot:** represents the distribution of a variable and displays outliers.
- **Heatmap:** shows the correlation or relationship between variables using colors.
- **Violin plot:** combines a box plot and a kernel density plot to represent the distribution of a variable.

## Q.2 How do you create a box plot using Matplotlib?

Ans: You can create a box plot using Matplotlib's **boxplot()** function, providing the data and any additional parameters. For example:

```
● ● ●  
import matplotlib.pyplot as plt  
import pandas as pd  
  
df = pd.read_csv('data.csv')  
plt.boxplot(df['column'])
```

## Q.3 How do you create a histogram using Seaborn?

Ans: You can create a histogram using Seaborn's **distplot()** function, specifying the variable and any additional parameters. For example:

```
● ● ●  
import seaborn as sns  
import pandas as pd  
  
df = pd.read_csv('data.csv')  
sns.distplot(df['column'])
```

## Q.4 How do you create a bar plot using Matplotlib?

Ans: You can create a bar plot using Matplotlib's **bar()** or **barh()** functions, providing the data and any additional parameters. For example:

```
● ● ●  
import matplotlib.pyplot as plt  
import pandas as pd  
  
df = pd.read_csv('data.csv')  
plt.bar(df['x_column'], df['y_column'])
```

## Q.5 How do you create a heatmap using Seaborn?

Ans: You can create a heatmap using Seaborn's **heatmap()** function, specifying the data, row and column variables, and any additional parameters. For example:

```
● ● ●  
import seaborn as sns  
import pandas as pd  
  
df = pd.read_csv('data.csv')  
sns.heatmap(data=df, x='x_column', y='y_column',  
cmap='coolwarm')
```

# Topic 9

## Library for machine learning: Scikit-learn:

### Q.1 What is Scikit-learn and how is it used in machine learning?

Ans: Scikit-learn is a popular machine learning library in Python that provides a wide range of algorithms and tools for various tasks such as classification, regression, clustering, dimensionality reduction, and model evaluation. It is widely used for building machine learning models and pipelines.

### Q.2 How do you train a machine learning model using Scikit-learn?

Ans: To train a machine learning model using Scikit-learn, you typically follow these steps:

- Preprocess and prepare your data.
- Choose a suitable algorithm.
- Split your data into training and testing sets.
- Fit the model to the training data using the fit() method.

- Evaluate the model's performance using metrics and test data.

### Q.3 How do you use cross-validation in Scikit-learn?

Ans: Scikit-learn provides the `cross_val_score()` function to perform cross-validation. You can specify the desired number of folds and the scoring metric to evaluate the model's performance. For example:

```
● ● ●  
from sklearn.model_selection import  
cross_val_score  
from sklearn.linear_model import  
LinearRegression  
  
model = LinearRegression()  
scores = cross_val_score(model, X, y, cv=5,  
scoring='r2')
```

### Q.4 How do you make predictions using a trained model in Scikit-learn?

Ans: Once you have trained a model in Scikit-learn, you can make predictions on new data using the `predict()` method. For example:

```
model = LinearRegression()  
model.fit(X_train, y_train)  
predictions = model.predict(X_test)
```

## Q.5 How do you save and load a trained model in Scikit-learn?

Ans: You can save a trained Scikit-learn model to disk using the **joblib** module's **dump()** function. To load a saved model, you can use the **load()** function.  
For example:

```
from sklearn.externals import joblib  
  
# Save the model  
joblib.dump(model, 'model.pkl')  
  
# Load the model  
loaded_model = joblib.load('model.pkl')
```



## WHY BOSSCODER?

 **1000+** Alumni placed at Top Product-based companies.

 More than **136% hike** for every **2 out of 3** working professional.

 Average package of **24LPA**.

The syllabus is most up-to-date and the list of problems provided covers all important topics.

Lavanya  
 Meta



Course is very well structured and streamlined to crack any MAANG company

Rahul .  
 Google



[EXPLORE MORE](#)