The 23 Top Python Interview Questions & Answers

Essential Python interview questions with examples for job seekers, final-year students, and data professionals.

Nov 2022 · 22 min read

Abid Ali Awan
I am a certified data scientist who enjoys building machine learning applications and writing blogs.

TOPICS

Python

Career Services

Python is the most popular language in the tech industry. During the interview, you will be asked to solve the challenge using Python and explain complex Python functionality. To pass the technical and coding stage, you need a guide or mock questions to practice.

In this post, we have outlined the most common question asked during technical interviews. Practicing these questions can help data professionals, developers, and software engineers successfully pass the interview stage.

Earn a Top Data Certification
Advance your career with industry-leading certifications.

Accelerate My Career

Lists and tuples are Python data structures. The list is dynamic and whereas the tuple has static characteristics. They both have various advantages and use cases.

List

The list is the mutable data type, consumes more memory, and it is better for element insertion and deletion. Furthermore, it has several building functions, and the implication of iterations is slower compared to Tuple.

Example:

```
a_list = ["Data", "Camp", "Tutorial"]
```

POWERED BY DATALAB

Tuple

The Tuple is an immutable data type, and it is generally used for accessing the elements. It is faster and consumes less memory, but it lacks built-in methods.

Example:

```
a_tuple = ("Data", "Camp", "Tutorial")
```

POWERED BY DATALAB

Learn more in our Python Lists tutorial.

2. What is __init__() in Python?

It is known as a constructor in OOP terminology. It is used to initiate a state when you create a new object. For example, you can assign values to object properties or run the operations that are necessary when the object is created.

The __init__() method is reserved for Python classes, and it is called automatically when you create a new object.

Example:

We have created a `book_shop` class and added the constructor and `book()` function. The constructor will store the book title name and the `book()` function will print the book name.

To test our code we have initialized the `b` object with "Sandman" and executed the `book()` function.

```python
class book_shop:

    # constructor
    def __init__(self, title):
        self.title = title

    # Sample method
    def book(self):
        print('The tile of the book is', self.title)


b = book_shop('Sandman')
b.book()
# The tile of the book is Sandman
```

POWERED BY DATALAB

unchanged during runtime, for example, a Numeric, String, and Tuple.

4. Explain List, Dictionary, and Tuple comprehension with an example.

List

List comprehension offers one-liner syntax to create a new list based on the values of the existing list. You can use a `for loop` to replicate the same thing, but it will require you to write multiple lines, and sometimes it can get complex.

List comprehension eases the creation of the list based on existing iterable.

```python
my_list = [i for i in range(1, 10)]
my_list
# [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

POWERED BY DATALAB

Dictionary

Similar to a List comprehension, you can create a dictionary based on an existing table with a single line of code. You need to enclose the operation with curly brackets `{}`.

```python
my_dict = {i for i in range(1, 10)}
my_dict
# {1, 2, 3, 4, 5, 6, 7, 8, 9}
```

POWERED BY DATALAB

Tuple

It is a bit different for Tuples. You can create Tuple comprehension using round brackets `()`, but it will return a generator object, not a tuple comprehension.

You can run the loop to extract the elements or convert them to a list.

```python
my_tuple = (i for i in range(1, 10))
my_tuple
# <generator object <genexpr> at 0x7fb91b151430>
```

POWERED BY DATALAB

You can learn more in our Python Tuples tutorial.

Advanced Python Interview Questions

These interview questions are for more experienced Python practitioners.

5. What is monkey patching in Python?

Monkey patching in Python is a dynamic technique that can change the behavior of the code at run-time. In short, you can modify a class or module at run-time.

Example:

Let's learn monkey patching with an example.

and running the `patch()` function.

Instead of displaying "patch() is being called", it has displayed "monk_p() is being called".

```python
class monkey:
    def patch(self):
            print ("patch() is being called")

def monk_p(self):
    print ("monk_p() is being called")

# replacing address of "patch" with "monk_p"
monkey.patch = monk_p

obj = monkey()

obj.patch()
# monk_p() is being called
```

POWERED BY DATALAB

6. What is the Python "with" statement designed for?

The `with` statement is used for exception handling to make code cleaner and simpler. It is generally used for the management of common resources like creating, editing, and saving a file.

Example:

Instead of writing multiple lines of open, try, finally, and close, you can create and write a text file using the `with` statement. It is simple.

```python
# using with statement
with open('myfile.txt', 'w') as file:
    file.write('DataCamp Black Friday Sale!!!')
```

POWERED BY DATALAB

7. Why use else in try/except construct in Python?

`try:` and `except:` are commonly known for exceptional handling in Python, so where does `else:` come in handy? `else:` will be triggered when no exception is raised.

Example:

Let's learn more about `else:` with a couple of examples.

1. On the first try, we entered 2 as the numerator and "d" as the denominator. Which is incorrect, and `except:` was triggered with "Invalid input!".

2. On the second try, we entered 2 as the numerator and 1 as the denominator and got the result 2. No exception was raised, so it triggered the `else:` printing the message "Division is successful."

```python
try:
    num1 = int(input('Enter Numerator: '))
    num2 = int(input('Enter Denominator: '))
    division = num1/num2
    print(f'Result is: {division}')
except:
    print('Invalid input!')
else:
```

```
## Try 2 ##
# Enter Numerator: 2
# Enter Denominator: 1
# Result is: 2.0
# Division is successful.
```

POWERED BY DATALAB

Take the **Python Fundamentals** skill track to gain the foundational skills you need to become a Python programmer.

Python Data Science Interview Questions

For those focused more on data science applications of Python, these are some questions you may encounter.

8. What are the advantages of NumPy over regular Python lists?

Memory

Numpy arrays consume less memory.

For example, if you create a list and a Numpy array of a thousand elements. The list will consume 48K bytes, and the Numpy array will consume 8k bytes of memory.

Speed

Numpy arrays take less time to perform the operations on arrays than lists.

For example, if we are multiplying two lists and two Numpy arrays of 1 million elements together. It took 0.15 seconds for the list and 0.0059 seconds for the array to operate.

Vesititly

Numpy arrays are convenient to use as they offer simple array multiple, addition, and a lot more built-in functionality. Whereas Python lists are incapable of running basic operations.

9. What is the difference between merge, join and concatenate?

Merge

Merge two DataFrames named series objects using the unique column identifier.

It requires two DataFrame, a common column in both DataFrame, and "how" you want to join them together. You can left, right, outer, inner, and cross join two data DataFrames. By default, it is an inner join.

```
pd.merge(df1, df2, how='outer', on='Id')
```

POWERED BY DATALAB

Join

Join the DataFrames using the unique index. It requires an optional `on` argument that can be a column or multiple column names. By default, the join function performs a left join.

```
df1.join(df2)
```

POWERED BY DATALAB

Concatenate

- **join()**: it combines two DataFrames by index.

- **merge()**: it combines two DataFrames by the column or columns you specify.

- **concat()**: it combines two or more DataFrames vertically or horizontally.

10. How do you identify and deal with missing values?

Identifying missing values

We can identify missing values in the DataFrame by using the `isnull()` function and then applying `sum()`. `Isnull()` will return boolean values, and the sum will give you the number of missing values in each column.

In the example, we have created a dictionary of lists and converted it into a pandas DataFrame. After that, we used isnull().sum() to get the number of missing values in each column.

```python
import pandas as pd
import numpy as np

# dictionary of lists
dict = {'id':[1, 4, np.nan, 9],
        'Age': [30, 45, np.nan, np.nan],
        'Score':[np.nan, 140, 180, 198]}

# creating a DataFrame
df = pd.DataFrame(dict)

df.isnull().sum()
# id       1
# Age      2
# Score    1
```

Dealing with missing values

There are various ways of dealing with missing values.

1. Drop the entire row or the columns if it consists of missing values using `dropna()`. This method is not recommended, as you will lose important information.

2. Fill the missing values with the constant, average, backward fill, and forward fill using the `fillna()` function.

3. Replace missing values with a constant String, Integer, or Float using the `replace()` function.

4. Fill in the missing values using an interpolation method.

Note: make sure you are working with a larger dataset while using the `dropna()` function.

```python
# drop missing values
df.dropna(axis = 0, how ='any')

#fillna
df.fillna(method ='bfill')

#replace null values with -999
df.replace(to_replace = np.nan, value = -999)
```

| 0 | 1.0 | 30.0 | nan |
| 1 | 4.0 | 45.0 | 140.0 |

Become a professional data scientist by taking the Data Scientist with Python career track. It includes 25 courses and six projects to help you learn all the fundamentals of data science with the help of Python libraries.

11. Which all Python libraries have you used for visualization?

Data visualization is the most important part of data analysis. You get to see your data in action, and it helps you find hidden patterns.

The most popular Python data visualization libraries are:

1. Matplotlib

2. Seaborn

3. Plotly

4. Bokeh

In Python, we generally use **Matplotlib** and **seaborn** to display all types of data visualization. With a few lines of code, you can use it to display scatter plot, line plot, box plot, bar chart, and many more.

For interactive and more complex applications, we use **Plotly**. You can use it to create colorful interactive graphs with a few lines of code. You can zoom, apply animation, and even add control functions. Plotly provides more than 40 unique types of charts, and we can even use them to create a web application or dashboard.

Bokeh is used for detailed graphics with a high level of interactivity across large datasets.

Python Coding Interview Questions

If you have a Python coding interview coming up, preparing questions similar to these can help you impress the interviewer.

12. How can you replace string space with a given character in Python?

It is a simple string manipulation challenge. You have to replace the space with a specific character.

Example 1: a user has provided the string "I vey u" and the character "o", and the output will be "loveyou".

Example 2: a user has provided the string "D t C mpBl ckFrid yS le" and the character "a", and the output will be "DataCampBlackFridaySale".

In the `str_replace()` function, we will loop over each letter of the string and check if it is space or not. If it consists of space, we will replace it with the specific character provided by the user. Finally, we will be returning the modified string.

```python
def str_replace(text,ch):
    result = ''
    for i in text:
        if i == ' ':
```

```
str_replace(text,ch)
# 'DataCampBlackFridaySale'
```

13. Given a positive integer num, write a function that returns True if num is a perfect square else False.

This has a relatively straightforward solution. You can check if the number has a perfect square root by:

1. Finding the square root of the number and converting it into an integer.

2. Applying the square to the square root number and checking if it's a perfect square root.

3. Returning the result as a boolean.

Test 1

We have provided number 10 to the `valid_square()` function.

1. By taking the square root of the number, we get 3.1622776601683795.

2. By converting it into an integer, we get 3.

3. Then, take the square of 3 and get 9.

4. 9 is not equal to the number, so the function will return False.

a

Test 2

We have provided number 36 to the `valid_square()` function.

1. By taking the square root of the number, we get 6.

2. By converting it into an integer, we get 6.

3. Then, take the square of 6 and get 36.

4. 36 is equal to the number, so the function will return True.

```python
def valid_square(num):
    square = int(num**0.5)
    check = square**2==num
    return check

valid_square(10)
# False
valid_square(36)
# True
```

14. Given an integer n, return the number of trailing zeroes in n factorial n!

To pass this challenge, you have to first calculate n factorial (n!) and then calculate the number of training zeros.

Finding factorial

```
7! = 5040
```

The seven factorials have a total of two zeros and only one trailing zero, so our solution should return 1.

1. Convert the factorial number to a string.

2. Read it back and apply for a loop.

3. If the number is 0, add +1 to the result, otherwise break the loop.

4. Returns the result.

The solution is elegant but requires attention to detail.

```python
def factorial_trailing_zeros(n):

    fact = n
    while n > 1:
        fact *= n - 1
        n -= 1

    result = 0

    for i in str(fact)[::-1]:
        if i == "0":
            result += 1
        else:
            break

    return result


factorial_trailing_zeros(10)
# 2
factorial_trailing_zeros(18)
# 3
```

Take the essential **practicing coding interview questions** course to prepare for your next coding interviews in Python.

FAANG Python Interview Question

Below, we've picked out some of the questions you might expect from the most sought-after roles in the industries, those at Meta, Amazon, Google, and the like.

Facebook/Meta Python interview questions

The exact questions you'll encounter at Meta depend largely on the role. However, you might expect some of the following:

15. String segmentation

You are provided with a large string and a dictionary of the words. You have to find if the input string can be segmented into words using the dictionary or not.

Image by Author

The solution is reasonably straightforward. You have to segment a large string at each point and check if the string can be segmented to the words in the dictionary.

1. Run the loop using the length of the large string.

2. We will create two substrings.

3. The first substring will check each point in the large string from s[0:i]

4. If the first substring is not in the dictionary, it will return False.

5. If the first substring is in the dictionary, it will create the second substring using s[i:0].

6. If the second substring is in the dictionary or the second substring is of zero length, then return True. Recursively call `can_segment_str()` with the second substring and return True if it can be segmented.

```python
def can_segment_str(s, dictionary):
    for i in range(1, len(s) + 1):
        first_str = s[0:i]
        if first_str in dictionary:
            second_str = s[i:]
            if (
                not second_str
                or second_str in dictionary
                or can_segment_str(second_str, dictionary)
            ):
                return True
    return False


s = "datacamp"
dictionary = ["data", "camp", "cam", "lack"]
can_segment_string(s, dictionary)
# True
```

POWERED BY DATALAB

16. Remove duplicates from sorted array

Given an integer sorted array in increasing order, remove the duplicate numbers such that each unique element appears only once. Make sure you keep the final order of the array the same.
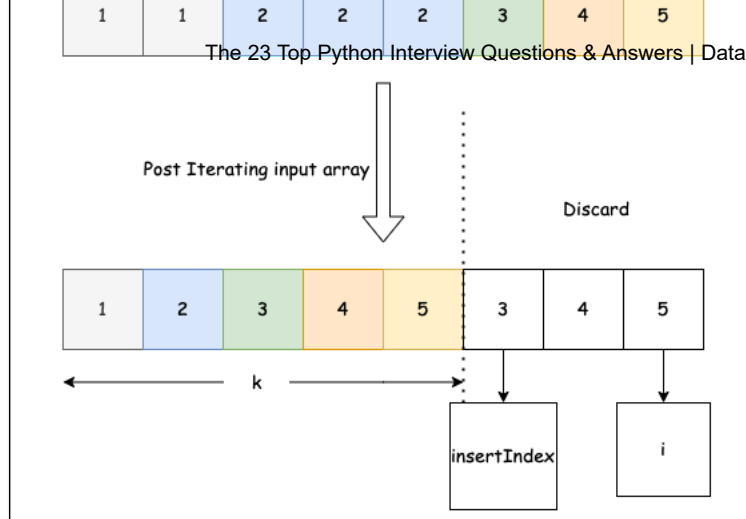
Image from LeetCode

Example 1: input array is [1,1,2,2], the function should return 2.

Example 2: input array is [1,1,2,3,3], the function should return 3.

Solution:

1. Run the loop for the range of 1 to the size of the array.

2. Check if the previous number is unique or not. We are comparing previous elements with the current one.

3. If it is unique, update the array using insertIndex, which is 1 at the start, and add +1 to the insertIndex.

4. Return insertIndex as it is the k.

This question is relatively straightforward once you know how. If you put more time into understanding the statement, you can easily come up with a solution.

```python
def removeDuplicates(array):
    size = len(array)
    insertIndex = 1
    for i in range(1, size):
        if array[i - 1] != array[i]:
            # Updating insertIndex in our main array
            array[insertIndex] = array[i]
            # Incrementing insertIndex count by 1
            insertIndex = insertIndex + 1
    return insertIndex

array_1 = [1,2,2,3,3,4]
removeDuplicates(array_1)
# 4


array_2 = [1,1,3,4,5,6,6]
removeDuplicates(array_2)
# 5
```

POWERED BY DATALAB

17. Find the maximum single sell profit

Example 2: stock_price = [8, 6, 5, 4, 3, 2, 1], buy = 6, and sell = 5. Minimizing the loss.

Solution:

1. We will calculate the global profit by subtracting global sell (the first element in the list) from current buy (the second element in the list).

2. Run the loop for the range of 1 to the length of the list.

3. Within the loop, calculate the current profit using list elements and current buy value.

4. If the current profit is greater than the global profit, change the global profit with the current profit and global sell to the i element of the list.

5. If the current buy is greater than the current element of the list, change the current buy with the current element of the list.

6. In the end, we will return global buy and sell value. To get global buy value, we will subtract global sell from global profit.

The question is a bit tricky, and you can come up with your unique algorithm to solve the problems.

```python
def buy_sell_stock_prices(stock_prices):
    current_buy = stock_prices[0]
    global_sell = stock_prices[1]
    global_profit = global_sell - current_buy

    for i in range(1, len(stock_prices)):
        current_profit = stock_prices[i] - current_buy

        if current_profit > global_profit:
            global_profit = current_profit
            global_sell = stock_prices[i]

        if current_buy > stock_prices[i]:
            current_buy = stock_prices[i]

    return global_sell - global_profit, global_sell

stock_prices_1 = [10,9,16,17,19,23]
buy_sell_stock_prices(stock_prices_1)
# (9, 23)


stock_prices_2 = [8, 6, 5, 4, 3, 2, 1]
buy_sell_stock_prices(stock_prices_2)
# (6, 5)
```

POWERED BY DATALAB

Amazon Python interview questions

Amazon Python interview questions can vary greatly but could include:

18. Find the missing number in the array

You have been provided with the list of positive integers from 1 to n. All the numbers from 1 to n are present except x, and you must find x.

Example:

4	5	3	2	8	1	6

2. By using arithmetic series sum formula, we will find the expected sum of the first n numbers.

3. Return the difference between the expected sum and the sum of the elements.

```python
def find_missing(input_list):

    sum_of_elements = sum(input_list)

    # There is exactly 1 number missing
    n = len(input_list) + 1
    actual_sum = (n * ( n + 1 ) ) / 2

    return int(actual_sum - sum_of_elements)
list_1 = [1,5,6,3,4]


find_missing(list_1)
# 2
```

POWERED BY DATALAB

19. Pythagorean Triplet in an array

Write a function that returns True if there is a Pythagorean triplet that satisfies $a^2 + b^2 = c^2$

Example:

Input	Output
[3, 1, 4, 6, 5]	True
[10, 4, 6, 12, 5]	False

Solution:

1. Square all the elements in the array.

2. Sort the array in increasing order.

3. Run two loops. The outer loop starts from the last index of the array to 1 and the inner loop starts from (outer_loop_index -1) to the start.

4. Create set() to store the elements between outer loop index and inner loop index.

5. Check if there is a number present in the set which is equal to (array[outerLoopIndex] – array[innerLoopIndex]). If yes, return True, else False.

```python
def checkTriplet(array):
    n = len(array)
    for i in range(n):
        array[i] = array[i]**2

    array.sort()

    for i in range(n - 1, 1, -1):
        s = set()
        for j in range(i - 1, -1, -1):
```

```
checkTriplet(arr)
# True
```

20. How many ways can you make change with coins and a total amount?

We need to create a function that takes a list of coin denominations and total amounts and returns the number of ways we can make the change.

In the example, we have provided coin denominations [1, 2, 5] and the total amount of 5. In return, we got five ways we can make the change.

Denomination	1,2,5
Amount	5

No. of combination
1, 1, 1, 1, 1
1, 1, 1, 2
1, 2, 2
5

Total No. of ways	5

Image by Author

Solution:

1. We will create the list of size amount + 1. Additional spaces are added to store the solution for a zero amount.

2. We will initiate a solution list with 1.

3. We will run two loops. The outer loop will return the number of denominations, and the inner loop will run from the range of the outer loop index to the amount +1.

4. The results of different denominations are stored in the array solution. solution[i] = solution[i] + solution[i - den]

5. The process will be repeated for all the elements in the denomination list, and at the last element of the solution list, we will have our number.

```python
def solve_coin_change(denominations, amount):
    solution = [0] * (amount + 1)
    solution[0] = 1
    for den in denominations:
        for i in range(den, amount + 1):
            solution[i] += solution[i - den]

    return solution[len(solution) - 1]

denominations = [1,2,5]
amount = 5
```

Google Python interview questions

As with the other companies mentioned, Google Python interview questions will depend on the role and level of experience. However, some common questions include:

21. Define a lambda function, an iterator, and a generator in Python.

The Lambda function is also known as an anonymous function. You can add any number of parameters but with only one statement.

An iterator is an object that we can use to iterate over iterable objects like lists, dictionaries, tuples, and sets.

The generator is a function similar to a normal function, but it generates a value using the yield keyword instead of return. If the function body contains yield, it automatically becomes a generator.

Read more about Python iterators and generators in our full tutorial.

22. Given an array arr[], find the maximum j – i such that arr[j] > arr[i]

This question is quite straightforward but requires special attention to detail. We are provided with an array of positive integers. We have to find the maximum difference between j-i where array[j] > array[i].

Examples:

1. Input: [20, 70, 40, 50, 12, 38, 98], Output: 6 (j = 6, i = 0)

2. Input: [10, 3, 2, 4, 5, 6, 7, 8, 18, 0], Output: 8 (j = 8, i = 0)

Solution:

1. Calculate the length of the array and initiate max difference with -1.

2. Run two loops. The outer loop picks elements from the left, and the inner loop compares the picked elements with elements starting from the right side.

3. Stop the inner loop when the element is greater than the picked element and keep updating the maximum difference using j - I.

```python
def max_index_diff(array):
    n = len(array)
    max_diff = -1
    for i in range(0, n):
        j = n - 1
        while(j > i):
            if array[j] > array[i] and max_diff < (j - i):
                max_diff = j - i
            j -= 1

    return max_diff

array_1 = [20,70,40,50,12,38,98]

max_index_diff(array_1)
# 6
```

POWERED BY DATALAB

23. How would you use the ternary operators in Python?

Ternary operators are also known as conditional expressions. They are operators that evaluate expression based on conditions being True and False.

```python
score = 75

if score < 70:
    if score < 50:
        print('Fail')
    else:
        print('Merit')
else:
    print('Distinction')
# Distinction
```

POWERED BY DATALAB

Nested Ternary Operator

```python
print('Fail' if score < 50 else 'Merit' if score < 70 else 'Distinction')
# Distinction
```

POWERED BY DATALAB

Python Interview FAQs

How do I prepare for a Python interview? ⌃

You need to revise Python syntax, functions, classes, data types, algorithms, data structures, and exceptional handling. Furthermore, you need to read technical tutorials, review example projects, cheat sheets, and mock questions, and solve coding challenges to pass the interview stage.

You need to prepare for general Python questions on native functionality, job-specific questions (data engineer, data scientist, backend developer), and timed code-based questions. You can practice machine learning Python questions and statistics Python interview questions with DataCamp.

How do you stand out in a Python coding interview? ⌄

What is Lambda in Python? ⌄

What are the five main uses of Python? ⌄

Is map faster than for loop? ⌄

TOPICS

Python   Career Services

Python Interview Courses

🗐 COURSE