







**Python Interview questions:**

1. **Features of Python?**

Ans: Features of Python:

1. Open source: Freely distributable
2. High level language: Natural language elements
3. Interpretable: Python script is saved and executed in the same format in which it was written
4. Object oriented programming language
5. Dynamically typed programming language
6. **Python 2 vs Python 3?**

| **Comparison Parameter** | **Python 2** | **Python 3** |
| --- | --- | --- |
| Year of Release | Python 2 was released in the year 2000. | Python 3 was released in the year 2008. |
| “Print” Keyword | In Python 2, print is considered to be a statement and not a function. | In Python 3, print is considered to be a function and not a statement. |
| Storage of Strings | In Python 2, strings are stored as ASCII by default. | In Python 3, strings are stored as UNICODE by default. |
| Division of Integers | On the division of two integers, we get an integral value in Python 2. For instance, 7/2 yields 3 in Python 2. | On the division of two integers, we get a floating-point value in Python 3. For instance, 7/2 yields 3.5 in Python 3. |
| Exceptions | In Python 2, exceptions are enclosed in notations. | In Python 3, exceptions are enclosed in parentheses. |
| Variable leakage | The values of global variables do change in Python 2 if they are used inside a for-loop. | The value of variables never changes in Python 3. |
| Iteration | In Python 2, the xrange() function has been defined for iterations. | In Python 3, the new Range() function was introduced to perform iterations. |
| Ease of Syntax | Python 2 has more complicated syntax than Python 3. | Python 3 has an easier syntax compared to Python 2. |
| Libraries | A lot of libraries of Python 2 are not forward compatible. | A lot of libraries are created in Python 3 to be strictly used with Python 3. |
| Usage in today’s times | Python 2 is no longer in use since 2020. | Python 3 is more popular than Python 2 and is still in use in today’s times. |
| Backward compatibility | Python 2 codes can be ported to Python 3 with a lot of effort. | Python 3 is not backward compatible with Python 2. |
| Application | Python 2 was mostly used to become a DevOps Engineer. It is no longer in use after 2020. | Python 3 is used in a lot of fields like Software Engineering, Data Science, etc. |

1. **Java vs Python**

Here, are the main differences between Java and Python.

| **Parameter** | **Java** | **Python** |
| --- | --- | --- |
| Compilation | Java is a Compiled Language | Python is an Interpreted Language |
| Static or Dynamic | Java is statically typed | Python is dynamically typed |
| String operations | Offers limited string related functions. | It offers lots of string related functions. |
| Learning curve | Complex learning curve | Easy to learn and use |
| Multiple inheritances | Multiple inheritances is partially done through interfaces. | It offers both single and multiple inheritances. |
| Braces vs. Indentation | It uses curly braces to define the beginning and end of each function and class definition. | Python uses indentation to separate code into code blocks. |
| Speed | Java program runs slowly compared to Python. | Python programs run faster than Java. |
| Portability | Any computer or mobile device which is able to run the Java virtual machine can run a Java application | Python programs need an interpreter installed on the target machine to translate Python code. Compared to Java, Python is less portable. |
| Read file | Java takes 10 lines of code to read from a file in Java. | Python only needs 2 lines of code. |
| Architecture | Java Virtual Machine provides the runtime environment to execute the code and convert bytecode into machine language. | For Python, the interpreter translates source code into machine-independent bytecode. |
| Backend Frameworks | Spring, Blade | Django, Flask |
| Machine Learning Libraries | Weka, Mallet, Deeplearning4j, MOA | Tensorflow. Pytorch. |
| Best features | * Great libraries * Widely used * Excellent tooling * Huge amount of documentation available. | * Readable code * Rapid development * Beautiful code |
| Best use for | Java is best for Desktop GUI apps, Embed Systems, Web application services, etc. | Python is excellent for scientific and numeric computing, Machine learning apps, more. |

1. **Print() function in python, sep, end, formatted string**
2. **How to know the type of a variable in Python**
3. **How to take input from the user? What is the data type of the value taken as input from the user? How to change this data type?**
4. **Raw string**
5. **How to determine the address of a variable? Can 2 variables have the same address?**
6. **Working with strings, splicing, methods of strings, formatted string, split function**

**How to Reverse a String in Python?**

In Python, there are no in-built functions that help us reverse a string. We need to make use of an array slicing operation for the same.

|  |  |
| --- | --- |
| 1 | str\_reverse = string[::-1] |

<https://towardsdatascience.com/41-questions-to-test-your-knowledge-of-python-strings-9eb473aa8fe8>

## 1. How would you confirm that 2 strings have the same identity?

The is operator returns True if 2 names point to the same location in memory. This is what we’re referring to when we talk about identity.

Don’t confuse is with ==, the latter which only tests equality.

animals = ['python','gopher']  
more\_animals = animalsprint(animals == more\_animals) #=> True  
print(animals is more\_animals) #=> Trueeven\_more\_animals = ['python','gopher']print(animals == even\_more\_animals) #=> True  
print(animals is even\_more\_animals) #=> False

Notice above how animals and even\_more\_animals have a different identity even though they are equal.

## 3. Check if a string contains a specific substring

The in operator will return True if a string contains a substring.

print( 'plane' in 'The worlds fastest plane' ) #=> True  
print( 'car' in 'The worlds fastest plane' ) #=> False

## 4. Find the index of the first occurrence of a substring in a string

There are 2 different functions that will return the starting index, find() and index(). They have slightly different behaviour.

find() returns -1 if the substring is not found.

'The worlds fastest plane'.find('plane') #=> 19  
'The worlds fastest plane'.find('car') #=> -1

## 17. Join a list of strings into a single string, delimited by hyphens

Python’s join() function can join characters in a list with a given character inserted between every element.

'-'.join(['a','b','c'])  
#=> 'a-b-c'

## 24. Convert an integer to a string

Use the string constructor, str() for this.

str(5) #=> '5'

## 30. Check if a string begins with or ends with a specific character?

startswith() and endswith() check if a string begins and ends with a specific substring.

city = 'New York'city.startswith('New') #=> True  
city.endswith('N') #=> False

## 40. Remove vowels from a string

One option is to iterate over the characters in a string via list comprehension. If they don’t match a vowel then join them back into a string.

string = 'Hello 1 World 2'vowels = ('a','e','i','o','u')''.join([c for c in string if c not in vowels])  
#=> 'Hll 1 Wrld 2'

1. **Working with lists and tuples, finding min and max values in a tuple**

**Give me an example where you can convert a list to a tuple?**

The below given example will show how to convert a list to a tuple. When we convert a list to a tuple we can make use of the <tuple()> function but do remember since tuples are immutable we cannot convert it back to a list.

fruits = [‘apple’, ‘orange’, ‘mango’, ‘papaya’, ‘guava’]

listAsTuple = tuple(fruits)

print(listAsTuple)

(‘apple’, ‘orange’, ‘mango’, ‘papaya’, ‘guava’)

**Give me an example on how you can convert a list to a string?**

Below given example will show how to convert a list to a string. When we convert a list to a string we can make use of the “.join” function to do the same.

fruits = [ ‘apple’, ‘orange’, ‘mango’, ‘papaya’, ‘guava’]

listAsString = ‘ ‘.join(fruits)

print(listAsString)

apple orange mango papaya guava

**How can you randomize the items of a list in place in Python?**

**Ans:** Consider the example shown below:

|  |  |
| --- | --- |
| 1  2  3  4 | **from** random **import** shuffle  x **=** ['Keep', 'The', 'Blue', 'Flag', 'Flying', 'High']  shuffle(x)  print(x) |

<https://pynative.com/python-list-exercise-with-solutions/>

### Exercise 3: Turn every item of a list into its square

Given a list of numbers. write a program to turn every item of a list into its square.

**Given**:

numbers = [1, 2, 3, 4, 5, 6, 7]

**Expected output:**

[1, 4, 9, 16, 25, 36, 49]

**Solution 1**: Using loop and list method

* Create an empty result list
* Iterate a numbers list using a loop
* In each iteration, calculate the square of a current number and add it to the result list using the append() method.

numbers = [1, 2, 3, 4, 5, 6, 7]

# result list

res = []

**for** i **in** numbers:

# calculate square and add to the result list

res.append(i \* i)

**print**(res)

**Solution 2**: Use list comprehension

numbers = [1, 2, 3, 4, 5, 6, 7]

res = [x \* x **for** x **in** numbers]

**print**(res)

### Exercise 4: Concatenate two lists in the following order

list1 = ["Hello ", "take "]

list2 = ["Dear", "Sir"]

**Expected output:**

['Hello Dear', 'Hello Sir', 'take Dear', 'take Sir']

Solution

list1 = ["Hello ", "take "]

list2 = ["Dear", "Sir"]

res = [x + y **for** x **in** list1 **for** y **in** list2]

**print**(res)

### Exercise 6: Remove empty strings from the list of strings

list1 = ["Mike", "", "Emma", "Kelly", "", "Brad"]

**Expected output:**

["Mike", "Emma", "Kelly", "Brad"]

Use a filter() function to remove None type from the list

list1 = ["Mike", "", "Emma", "Kelly", "", "Brad"]

# remove None from list1 and convert result into list

res = **list**(**filter**(None, list1))

**print**(res)

### Exercise 10: Remove all occurrences of a specific item from a list.

Given a Python list, write a program to remove all occurrences of item 20.

**Given**:

list1 = [5, 20, 15, 20, 25, 50, 20]

**Expected output:**

[5, 15, 25, 50]

**Solution 1**: Use the list comprehension

list1 = [5, 20, 15, 20, 25, 50, 20]

# list comprehension

# remove specific items and return a new list

**def** remove\_value(sample\_list, val):

**return** [i **for** i **in** sample\_list **if** i != val]

res = remove\_value(list1, 20)

**print**(res)

**Solution 2**: [while loop](https://pynative.com/python-while-loop/) (slow solution)

list1 = [5, 20, 15, 20, 25, 50, 20]

**while** 20 **in** list1:

list1.remove(20)

**print**(list1)

<https://www.w3resource.com/python-exercises/list/>

<https://www.plus2net.com/python/list-questions.php>

<https://pynative.com/python-tuple-exercise-with-solutions/>

### Exercise 1: Reverse the tuple

tuple1 = (10, 20, 30, 40, 50)

**Expected output:**

(50, 40, 30, 20, 10)

Solution

tuple1 = (10, 20, 30, 40, 50)

tuple1 = tuple1[::-1]

**print**(tuple1)

### Exercise 2: Access value 20 from the tuple

The given tuple is a nested tuple. write a Python program to print the value 20.

**Given**:

tuple1 = ("Orange", [10, 20, 30], (5, 15, 25))

**Expected output:**

20

tuple1 = ("Orange", [10, 20, 30], (5, 15, 25))

# understand indexing

# tuple1[0] = 'Orange'

# tuple1[1] = [10, 20, 30]

# list1[1][1] = 20

**print**(tuple1[1][1])

### Exercise 4: Unpack the tuple into 4 variables

Write a program to unpack the following tuple into four [variables](https://pynative.com/python-variables/) and display each variable.

**Given**:

tuple1 = (10, 20, 30, 40)

**Expected output:**

tuple1 = (10, 20, 30, 40)

# Your code

print(a) # should print 10

print(b) # should print 20

print(c) # should print 30

print(d) # should print 40

Solution

tuple1 = (10, 20, 30, 40)

# unpack tuple into 4 variables

a, b, c, d = tuple1

**print**(a)

**print**(b)

**print**(c)

**print**(d)

### Exercise 6: Copy specific elements from one tuple to a new tuple

Write a program to copy elements 44 and 55 from the following tuple into a new tuple.

**Given**:

tuple1 = (11, 22, 33, 44, 55, 66)

**Expected output:**

tuple2: (44, 55)

Solution

tuple1 = (11, 22, 33, 44, 55, 66)

tuple2 = tuple1[3:-1]

**print**(tuple2)

<https://www.w3resource.com/python-exercises/tuple/>

<https://csiplearninghub.com/important-tuple-questions-in-python/>

1. **Range data type**
2. **Dictionary data type**
3. **Functions**
4. **Classes**
5. **Modules and packages**
6. **Generating random number**
7. **Pathlib**
8. **File handling functions**
9. **del keyword, del vs none**
10. **How to join elements in a list**
11. **Set**
12. **Pickling and unpickling**
13. **Regular expressions**
14. **Deep copy and shallow copy**

How Will You Merge Elements in a Sequence?

There are three types of sequences in Python:

* Lists
* Tuples
* Strings

Example of Lists -

>>l1=[1,2,3]

>>l2=[4,5,6]

>>l1+l2

Output: [1,2,3,4,5,6]

Example of Tuples -

>>t1=(1,2,3)

>>t2=(4,5,6)

>>t1+t2

Output: (1,2,3,4,5,6)

Example of String -

>>s1=“Simpli”

>>s2=“learn”

>>s1+s2

Output: ‘Simplilearn’

What Is the Difference Between Del and Remove() on Lists?

|  |  |
| --- | --- |
| del | remove() |
| * del removes all elements of a list within a given range * Syntax: del list[start:end] | * remove() removes the first occurrence of a particular character * Syntax: list.remove(element) |

Here is an example to understand the two statements -

>>lis=[‘a’, ‘b’, ‘c’, ‘d’]

>>del lis[1:3]

>>lis

Output: [“a”,”d”]

>>lis=[‘a’, ‘b’, ‘b’, ‘d’]

>>lis.remove(‘b’)

>>lis

Output: [‘a’, ‘b’, ‘d’]

Note that in the range 1:3, the elements are counted up to 2 and not 3.

<https://www.mygreatlearning.com/blog/python-interview-questions/>

**What is a map() function in Python?**

The map() function in Python is used for applying a function on all elements of a specified iterable. It consists of two parameters, function and iterable. The function is taken as an argument and then applied to all the elements of an iterable(passed as the second argument). An object list is returned as a result.

def add(n):

return n + n number= (15, 25, 35, 45)

res= map(add, num)

print(list(res))

o/p: 30,50,70,90

**How do you delete a file in Python?**

Files can be deleted in Python by using the command os.remove (filename) or os.unlink(filename)