python coding interview questions and answers pdf

ideasorblogs.in/python-coding-interview-questions-and-answers-pdf/

Publisher January 24, 2023

This is a list of 13 Python programming questions that can be used in an interview setting to assess a candidate’s knowledge and experience with the language. The questions cover a wide range of topics including data types and structures, functions, decorators, exceptions, and file as well as more advanced concepts such as implementing linked lists, dictionaries, and tree. Answers provide an overview of the concept and how it can be implemented in Python.

1. What is the difference between a tuple and a list in Python?
2. How do you handle exceptions in Python?
3. What is a decorator in Python and how do you use it?
4. Explain the difference between local and global variables in Python.
5. How do you define a function in Python?
6. Explain the difference between `len()` and `count()` in Python.
7. What is the difference between `append()` and `extend()` in a Python list?
8. What is the difference between `list()` and `set()` in Python?
9. How do you perform file I/O operations in Python?
10. How do you define a lambda function in Python?
11. How do you implement a linked list in Python?
12. How do you implement a dictionary in Python?
13. How do you implement a tree in Python?

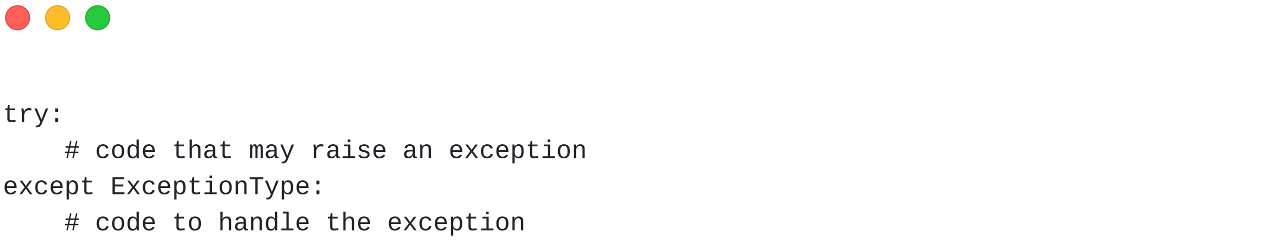
1) What is the difference between a tuple and a list in Python?

- A tuple is an immutable sequence type in Python, which means its elements cannot be modified once assigned. A list, on the other hand, is a mutable sequence type, which means its elements can be modified. For example, you can add or remove elements from a list, but not from a tuple.
- Additionally, tuples are generally faster and use less memory compared to lists because they don’t need to support the extra functionality that lists provide.
- A list, on the other hand, is a mutable data structure, meaning that the elements of a list can be modified after it is created. Lists are defined using square brackets, like this: `[1, 2, 3]`.
- Another difference between tuples and lists is the way they are accessed. Elements in a tuple are accessed by their position (index) in the tuple, while elements in a list are accessed by their position (index) in the list.
- In summary, tuples are immutable and defined using parentheses, while lists are mutable and defined using square brackets. They are similar in that they both allow you to store a collection of items, but they have different properties and use cases.

2) How do you handle exceptions in Python?

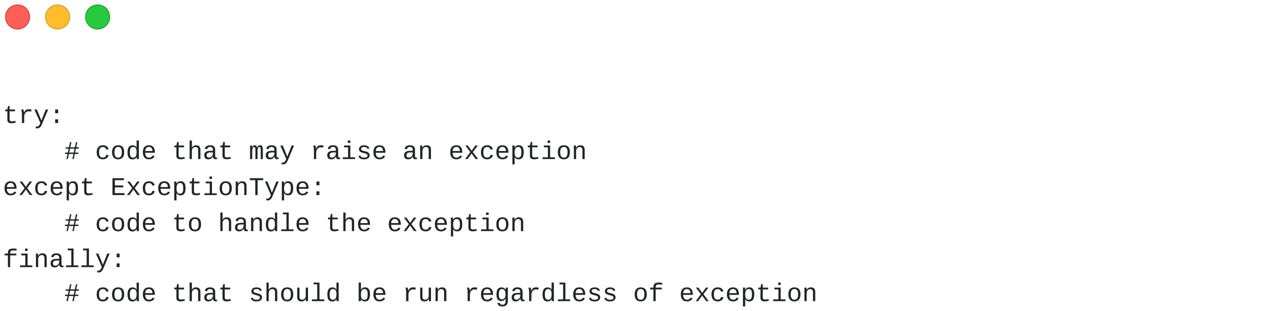
In Python, exceptions are handled using the `try` and `except` keywords. You can use a `try` block to execute code that may raise an exception, and an `except` block to handle the exception if it occurs.

For example:



```
try:
    # code that may raise an exception
except ExceptionType:
    # code to handle the exception
```

You can also use the `finally` block to execute code that should be run whether an exception occurs or not.

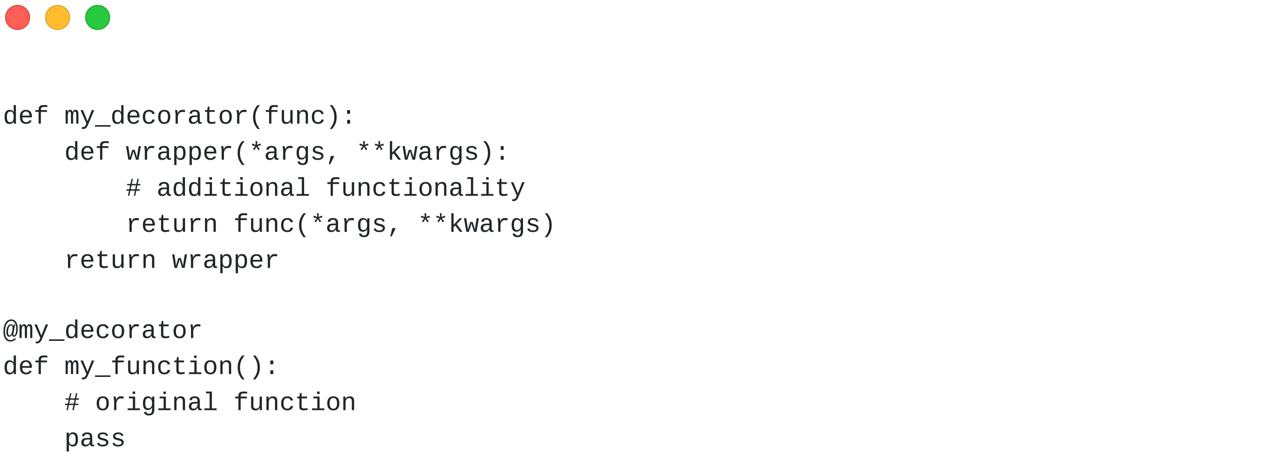


```
try:
    # code that may raise an exception
except ExceptionType:
    # code to handle the exception
finally:
    # code that should be run regardless of exception
```

3) What is a decorator in Python and how do you use it?

A decorator is a function that modifies the behavior of another function, by wrapping it with additional functionality. Decorators are used to add functionality to a function without modifying its source code. They are denoted by the `@` symbol followed by the decorator function.

For example:



```
def my_decorator(func):
    def wrapper(*args, **kwargs):
        # additional functionality
        return func(*args, **kwargs)
    return wrapper

@my_decorator
def my_function():
    # original function
    pass
```

The `my_decorator` function takes the `my_function` as input, wraps it with additional functionality, and returns the new `wrapper` function.

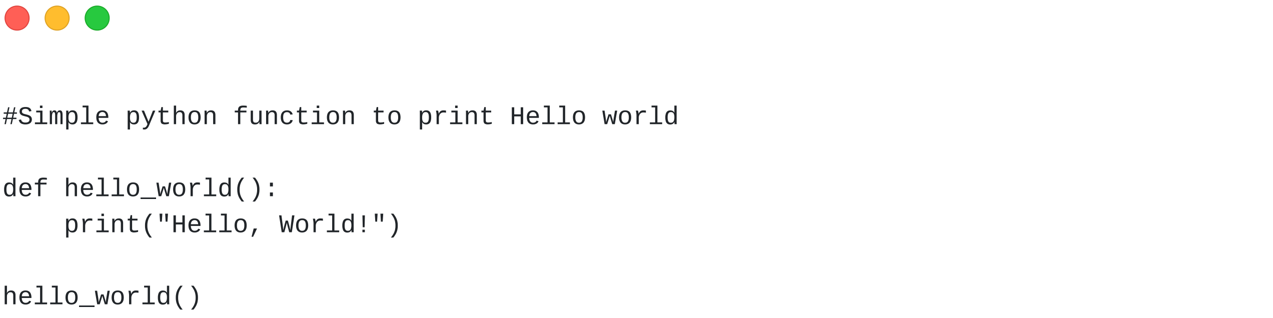
4) Explain the difference between local and global variables in Python.

- Local variables are defined within a function or a class method and have scope only within that function or method. They are not accessible outside of that scope.
- Global variables, on the other hand, are defined outside of any function or method and have scope throughout the entire program. They can be accessed and modified from any function or method in the program.
- It’s generally considered best practice to use local variables as much as possible and to minimize the use of global variables, as they can cause confusion and make code harder to understand and maintain.

5) How do you define a function in Python?

In Python, functions are defined using the `def` keyword, followed by the function name and a set of parentheses that may contain function parameters. The function body is indented under the function definition.

For example:

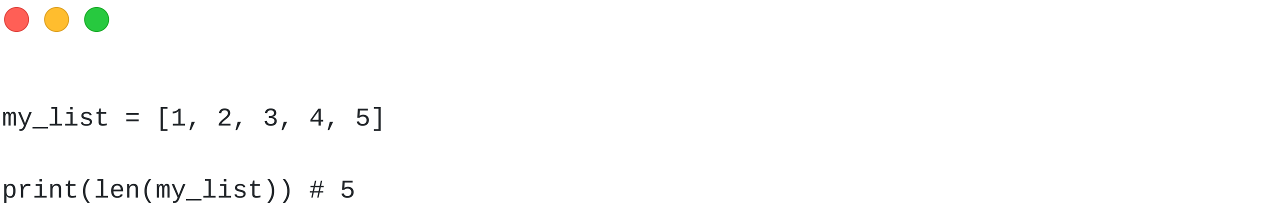


```
#Simple python function to print Hello world

def hello_world():
    print("Hello, World!")

hello_world()
```

6) Explain the difference between `len()` and `count()` in Python.

len() is a built-in function that returns the number of elements in a given sequence, such as a list, string, or tuple.

For example:



```
my_list = [1, 2, 3, 4, 5]

print(len(my_list)) # 5
```

`count()` is a method that is available for some sequence types, such as lists and strings, and returns the number of occurrences of a specific element in the sequence.

For example:

```
my_list = [1, 2, 3, 4, 5, 5]

print(my_list.count(5)) # 2
```

7) What is the difference between `append()` and `extend()` in a Python list?

`append()` is a method that adds an element to the end of a list.

For example:

```
my_list = [1, 2, 3]

my_list.append(4) # my_list is now [1, 2, 3, 4]
```

`extend()` is a method that adds multiple elements to the end of a list. It takes an iterable (such as a list, tuple, or string) as an argument and adds each element from the iterable to the list.

For example:

```
my_list = [1, 2, 3]

my_list.extend([4, 5, 6]) # my_list is now [1, 2, 3, 4, 5, 6]
```

8) What is the difference between `list()` and `set()` in Python?

- `list()` is a built-in function that creates a new list. Lists are ordered collections of elements that can contain duplicate values.
- `set()` is a built-in function that creates a new set. Sets are unordered collections of unique elements. They are commonly used to remove duplicates from a list or to check if an element is present in a collection.

9) How do you perform file I/O operations in Python?

- Python provides a built-in `open()` function that allows you to open and read/write to files. The `open()` function takes the file path and the mode
- **(such as “r” for reading and “w” for writing, and “a” for appending**) as arguments.
- Once the file is opened, you can use various methods such as `read()`, `write()`, `seek()`, and `close()` to perform file operations. For example, to read the content of a file you can use:

```
with open("text_file.txt", "r") as file:
    content = file.read()
    print(content)
```

To write to a file you can use:

```
with open("text_file.txt", "w") as file:
    file.write("Hello, World!")
```

10) How do you define a lambda function in Python?

A lambda function is a small, anonymous function that is defined using the `lambda` keyword. It can take any number of arguments, but can only have one expression. The general syntax for a lambda function is `lambda arguments: expression`.

For example:

```
my_lambda = lambda x: x**2

print(my_lambda(5)) # 25
```

11) How do you implement a linked list in Python?

You can implement a linked list in Python by creating a class for the nodes and a class for the linked list. Each node class should have a value and a reference to the next node, and the linked list class should have a reference to the head and tail of the list and methods for adding and removing nodes.

12) How do you implement a dictionary in Python?

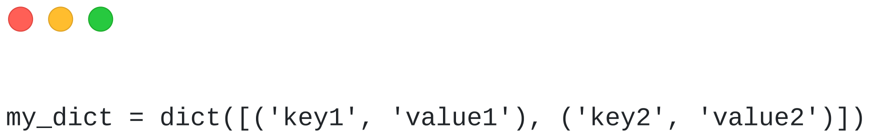
A dictionary in Python is implemented as a built-in data type called `dict`. It is an unordered collection of key-value pairs. Each key is associated with a value, and you can use the key to access the value. You can create a dictionary by using curly braces {} and separating key-value pairs by a colon.

For example:

```
text_dict = {'x': 'hello'},

#Here x is = key and value is = hellow
```

You can also use the `dict()` constructor to create a dictionary from a list of key-value pairs or from another mapping object.

For example:



```
my_dict = dict([('key1', 'value1'), ('key2', 'value2')])
```

13) How do you implement a tree in Python?

> To implement a tree in Python, you can create a class for the nodes and a class for the tree. Each node class should have a value and a list of references to its children nodes, and the tree class should have a reference to the root node and methods for adding and removing nodes.

In conclusion, this list of 13 Python programming questions can be a useful tool for anyone looking to assess their knowledge and experience with the language. It covers a wide range of topics and provides an overview of important concepts and their implementation in Python. Whether you’re preparing for a job interview or simply looking to test your skills, these questions can be a valuable resource.