1. What does an empty dictionary's code look like?

**Solution 1.**

In Python, an empty dictionary is represented by a pair of curly braces {}. Here's an example of what an empty dictionary looks like:

my\_dict = {}

In this example, my\_dict is an empty dictionary. It does not contain any key-value pairs.

You can also create an empty dictionary using the dict() constructor without passing any arguments

my\_dict = dict()

2. What is the value of a dictionary value with the key 'foo' and the value 42?

**Solution 2.**

The value of a dictionary with the key 'foo' and the value 42 can be accessed by using the key within square brackets []. Here's an example:

my\_dict = {'foo': 42}

value = my\_dict['foo']

print(value)

**Output:**

42

3. What is the most significant distinction between a dictionary and a list?

**Solution 3.**

The most significant distinction between a dictionary and a list in Python is the way they store and access elements.

A dictionary is an unordered collection of key-value pairs. Each key in a dictionary is unique and is used to retrieve its associated value. Dictionaries are implemented as hash tables, which allows for efficient lookup of values based on their keys. The key-value pairs in a dictionary are not ordered, and the elements can be accessed using the keys rather than indexing.

On the other hand, a list is an ordered collection of elements. The elements in a list are indexed starting from 0, and you can access them based on their position or index within the list. Lists are mutable, meaning you can modify their elements, add new elements, or remove existing elements.

4. What happens if you try to access spam['foo'] if spam is {'bar': 100}?

**Solution 4.**

If you try to access spam['foo'] where spam is {'bar': 100}, you will encounter a KeyError. This error occurs because the key 'foo' does not exist in the dictionary spam.

In Python, when you try to access a key in a dictionary that does not exist, a KeyError is raised. It indicates that the specified key is not found in the dictionary

5. If a dictionary is stored in spam, what is the difference between the expressions 'cat' in spam and 'cat' in spam.keys()?

**Solution 5.**

The practical difference between these expressions lies in their performance and efficiency.

Using 'cat' in spam directly checks for the key in the dictionary, which internally utilizes a hash table for efficient key lookup. It has a time complexity of O(1), providing a fast and efficient way to check for key existence.

On the other hand, using 'cat' in spam.keys() first creates a list of all the keys in the dictionary using the keys() method. This incurs additional memory and time overhead to generate the list. Then, it checks if the key is present in that list, which has a time complexity of O(n), where n is the number of keys in the dictionary. This approach may be less efficient, especially for larger dictionaries.

6. If a dictionary is stored in spam, what is the difference between the expressions 'cat' in spam and 'cat' in spam.values()?

**Solution 6.**

The expressions 'cat' in spam and 'cat' in spam.values() have different meanings and behaviors when it comes to dictionaries.

'cat' in spam:

This expression checks if the key 'cat' exists in the dictionary spam. It returns a Boolean value of True if the key is present as one of the keys in the dictionary and False otherwise. It checks for the presence of keys in the dictionary.

'cat' in spam.values():

This expression checks if the value 'cat' exists in the values of the dictionary spam. It retrieves all the values from the dictionary using the values() method, which returns a list-like view object representing the values in the dictionary. Then, it checks if the value 'cat' is present in that list of values. It returns a Boolean value of True if the value is found and False otherwise.

7. What is a shortcut for the following code?

if 'color' not in spam:

spam['color'] = 'black'

**Solution 7.**

A shortcut for the given code is to use the dict.setdefault() method. It allows you to set a default value for a key in a dictionary if the key does not already exist. Here's how we can use it to achieve the same result:

spam.setdefault('color', 'black')

The setdefault() method checks if the specified key ('color' in this case) exists in the dictionary (spam). If the key is present, it returns the corresponding value. If the key is not found, it adds the key-value pair to the dictionary with the default value provided ('black' in this case) and returns that default value.

8. How do you "pretty print" dictionary values using which module and function?

**Solution 8.**

To "pretty print" dictionary values in Python, you can make use of the pprint module (Pretty Print). The pprint module provides a pprint() function that formats the output in a more readable and aesthetically pleasing way.

Here's an example of how to use the pprint module to pretty print dictionary values:

import pprint

my\_dict = {'name': 'Alice', 'age': 25, 'city': 'New York'}

pprint.pprint(my\_dict)

**Output:**

{'age': 25,

'city': 'New York',

'name': 'Alice'}