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

Ans: is an empty dictionary, just like an empty pair of [] is an empty list. # The length len() of a dictionary is the number of key-value pairs it has.

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

Ans: The value of a dictionary with the key 'foo' and the value 42 would be 42. In Python, dictionaries are data structures that store key-value pairs, where each key is unique. You can access the value associated with a specific key by using the key in square brackets, like dictionary[key]. Therefore, if you have a dictionary with the key 'foo' and the value 42, accessing the value using the key 'foo' would give you the value 42.

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

Ans: The most significant distinction between a dictionary and a list is their underlying structure and the way they store and access data.

Structure:

List: A list is an ordered collection of items. It maintains the order of elements, and each element is assigned an index based on its position in the list.

Dictionary: A dictionary is an unordered collection of key-value pairs. It does not maintain any particular order, and the elements are accessed using their keys rather than indices.

Accessing Elements:

List: In a list, elements are accessed by their indices. You can retrieve an element by its position using indexing, such as my\_list[index].

Dictionary: In a dictionary, elements are accessed by their keys. Each key in the dictionary is associated with a value, and you can retrieve a value by using the corresponding key, like my\_dict[key].

Key-Value Pair:

List: A list only contains elements without any explicit key-value association. The elements are stored sequentially based on their positions.

Dictionary: A dictionary explicitly associates each value with a unique key. It allows you to retrieve values based on their corresponding keys, making it efficient for data lookup and retrieval.

Mutable vs. Immutable:

List: Lists are mutable, meaning you can modify their elements by assigning new values or manipulating the existing elements.

Dictionary: Dictionaries are also mutable. You can add, remove, or modify key-value pairs within a dictionary.

Overall, lists are suitable for ordered collections of items that you access by index, while dictionaries are ideal for storing key-value pairs and efficient data retrieval based on specific keys. The choice between them depends on the nature of the data and the operations you need to perform.

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

Ans: If you try to access spam['foo'] with the given dictionary spam = {'bar': 100}, you will get a KeyError because the key 'foo' does not exist in the dictionary spam.

In Python, dictionaries use keys to access their corresponding values. When you attempt to access a key that does not exist in a dictionary, Python raises a KeyError exception.

In the given example, the dictionary spam only contains one key-value pair: 'bar': 100. Therefore, when you try to access spam['foo'], Python will raise a KeyError since the key 'foo' is not present in the dictionary spam.

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

Ans: The expressions 'cat' in spam and 'cat' in spam.keys() are slightly different in their behavior when used with a dictionary.

'cat' in spam: This expression checks if the key 'cat' exists in the dictionary spam. It returns a boolean value (True or False) indicating whether the key is present or not. This is a direct membership test for the key within the dictionary.

'cat' in spam.keys(): This expression checks if the key 'cat' exists in the list of keys of the dictionary spam. It first retrieves all the keys of the dictionary using the keys() method, which returns a view object that represents the keys. Then, it checks if 'cat' is present in that list of keys. It also returns a boolean value (True or False).

In terms of the end result, both expressions will yield the same boolean value if 'cat' is a key in the dictionary spam. However, there can be a subtle difference in terms of performance and efficiency.

The expression 'cat' in spam directly checks for the key in the dictionary, which is typically faster because it directly looks up the key without creating an intermediate list of keys. On the other hand, 'cat' in spam.keys() retrieves all the keys and then performs the membership test. This could be slower and less efficient, especially for large dictionaries, as it creates an additional list of keys.

Therefore, it is generally recommended to use 'cat' in spam for checking the presence of a key in a dictionary, as it is more concise and efficient.

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

Ans: The expressions 'cat' in spam and 'cat' in spam.values() have different purposes when used with a dictionary.

'cat' in spam: This expression checks if the string 'cat' exists as a key in the dictionary spam. It returns a boolean value (True or False) indicating whether the key is present or not. It performs a direct membership test for the key within the dictionary.

'cat' in spam.values(): This expression checks if the string 'cat' exists as a value in the dictionary spam. It retrieves all the values from the dictionary using the values() method, which returns a view object representing the values. Then, it checks if 'cat' is present in that collection of values. It returns a boolean value (True or False).

In terms of the end result, both expressions will yield the same boolean value if 'cat' is present either as a key or a value in the dictionary spam. However, they check for the presence of 'cat' in different parts of the dictionary.

If you are interested in checking whether 'cat' exists as a key in the dictionary, you should use 'cat' in spam. On the other hand, if you want to check whether 'cat' exists as a value in the dictionary, you should use 'cat' in spam.values().

In summary, 'cat' in spam checks for the presence of the key 'cat', while 'cat' in spam.values() checks for the presence of the value 'cat' within the dictionary spam.

7. What is a shortcut for the following code?

if 'color' not in spam:

spam['color'] = 'black'

Ans: A shortcut for the given code can be achieved by using the dict.setdefault() method.

The setdefault() method allows you to specify a default value for a key in a dictionary if the key does not already exist. If the key is present, it returns the corresponding value. If the key is not present, it adds the key-value pair to the dictionary with the provided default value and returns the default value.

Using setdefault(), the code can be shortened to:

python

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

This line of code checks if the key 'color' exists in the dictionary spam. If the key is present, it returns the corresponding value. If the key is not present, it adds the key 'color' with the value 'black' to the dictionary spam.

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

Ans: To "pretty print" dictionary values in Python, you can use the pprint module, specifically the pprint() function.

The pprint module provides a way to format and display complex data structures, including dictionaries, in a more readable and visually appealing format. It is especially useful when working with large or nested dictionaries.

To use the pprint module, you need to import it at the beginning of your code:

python

import pprint

Then, you can use the pprint() function to pretty print a dictionary:

python

my\_dict = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3'}

pprint.pprint(my\_dict)

The pprint() function will display the dictionary in a well-formatted and indented manner, making it easier to read and understand the structure of the dictionary.

Note that pprint stands for "pretty print," and it is a built-in module in Python, so you do not need to install any additional packages.