1. An empty dictionary's code looks like this:

   dictionary = {}
   The curly braces **'{}'** represent an empty dictionary. The dictionary does not contain any key-value pairs
   dictionary = {}
   dictionary["name"] = "AAjay"
   dictionary["age"] = 30

   After running this code, the dictionary will contain the key-value pairs "name"="AAjay" and "age"=30.

2. The value of a dictionary with the key 'foo' and the value 42 would be **42**. In a dictionary, values are associated with specific keys, allowing you to access and retrieve values by using the corresponding keys.
   example of a dictionary with the key 'foo' and the value 42:
   my_dictionary = {'foo': 42}

3. The most significant distinction between Dictionary and list is that **dictionaries can only contain unique keys, while lists can contain duplicate elements**. This means that you cannot have two keys with the same value in a dictionary, but you can have multiple elements with the same value in a list.

   dictionary = {"a": 1, "b": 2, "a": 3}

   above code **will raise an error** because **dictionaries cannot contain duplicate keys**. However, the following code will create a list that contains multiple elements with the same value:

   list = ["a", "b", "c", "a"]

4. If you try to access **spam['foo']** and **spam is {'bar': 100**}, you will encounter a **KeyError**. The **KeyError** is raised when you try to access a key in a dictionary that does not exist.

5. The expressions **'cat'** in **spam** and **'cat'** in **spam.keys()** have slightly different meanings, but they both returns a boolean value **'True**' if the 'cat' is found and **'False'** otherwise.
   The difference between the two expressions is that the first expression checks if the string **"cat" is any element in the dictionary**, while the second expression checks if the string **"cat" is a key** in the dictionary.

   'cat' in spam: It directly checks for the presence of the string **'cat'** within the dictionary itself, which is generally more efficient.

   'cat' in spam.keys() : It first creates a list-like object of the keys using spam.keys(), and then checks for the presence of the key within that list. This approach involves an additional step of generating the list of keys

6. The expression "**cat" in spam** checks if the string "**cat" is any value** in the dictionary spam. If it is, the expression will return **'True'** otherwise '**False.'**

The expression **"cat" in spam.values()** checks if the string **"cat" is a value** in the dictionary spam. If it is, the expression will return '**True'**. Otherwise will return '**False'.**

The difference between the two expressions is that the first expression checks if the string **"cat" is any element in the dictionary**, while the second expression checks if the string **"cat" is a value in the dictionary.**

7. A shortcut for the given code can be achieved using the **dict.setdefault()** method
   The **setdefault()** method takes two arguments: the **key** and the **default value**. If the key is not in the dictionary, the method will add the key-value pair to the dictionary with the default value. If the key is already in the dictionary, the method will not modify the dictionary.
   spam.setdefault('color', 'black')
   This single line of code can produce the same result

8. To **"pretty print**" dictionary values we use the **'pprint'** module and function,
   consider the following code:

   ```
   import pprint
   spam = {"name": "Vipin", "age": 30, "color": "blue"}
   pprint.pprint(spam)
   ```

   the "pretty print" output of the dictionary spam will be

   ```
   {
     'name': 'Vipin',
     'age': 30,
     'color': 'blue'
   }
   ```

   The pprint() function takes care of formatting the dictionary structure automatically.