Uses of hash map in python

In Python, a hash map is commonly implemented using a built-in data structure called a dictionary. A dictionary is an unordered collection that stores key-value pairs, where each key must be unique. Python dictionaries are highly efficient and provide fast access to values based on their keys. They are widely used for tasks that involve mapping keys to corresponding values, such as caching, data lookups, and counting occurrences.

1. **Creating a dictionary: You can create a dictionary by using curly braces {} and providing key-value pairs separated by a colon :**

```
# Empty dictionary

my_dict = {}

# Dictionary with some initial key-value pairs

my_dict = {'apple': 1, 'banana': 2, 'orange': 3}
```

2-Adding and updating elements: You can add new key-value pairs to the dictionary or update existing values for a given key.

```
# Adding a new key-value pair

my_dict['grape'] = 4

# Updating the value for an existing key

my_dict['banana'] = 5
```

3-Accessing elements: You can access values in the dictionary using their corresponding keys.

```
# Accessing values

print(my_dict['apple'])  # Output: 1

# Using the get() method to avoid KeyError if the key doesn't exist

print(my_dict.get('banana'))  # Output: 5

print(my_dict.get('watermelon'))  # Output: None
```

4-Checking if a key exists: You can check if a key exists in the dictionary using the in keyword.

```
# Checking if a key exists
if 'apple' in my_dict:
    print("Yes, 'apple' exists in the dictionary.")
```

5-Iterating through the dictionary: You can iterate through the keys, values, or key-value pairs of the dictionary.

```
# Iterating through keys
for key in my_dict:
    print(key)

# Iterating through values
for value in my_dict.values():
    print(value)

# Iterating through key-value pairs
for key, value in my_dict.items():
    print(f"{key}: {value}")
```

6-Removing elements: You can remove elements from the dictionary using the del keyword or the pop() method.

```
# Removing a key-value pair using the 'del' keyword

del my_dict['apple']

# Removing a key-value pair using the 'pop()' method

removed_value = my_dict.pop('banana')
print(removed_value) # Output: 5
```