In [6]:

```python
# Initialize a dictionary
my_dict = {
    'name': 'Alice',
    'age': 30,
    'city': 'New York',
    'hobbies': ['reading', 'hiking', 'coding']
}

# 1. get(key[, default])
# Returns the value for `key` if `key` is in the dictionary; otherwise, returns `default
value_name = my_dict.get('name')
print("Value for 'name':", value_name)  # Output: Alice

value_nonexistent = my_dict.get('address', 'Not Found')
print("Value for 'address':", value_nonexistent)  # Output: Not Found

# 2. keys()
# Returns a view object that displays a list of all the keys in the dictionary.
keys_view = my_dict.keys()
print("Keys:", list(keys_view))  # Output: ['name', 'age', 'city', 'hobbies']

# 3. values()
# Returns a view object that displays a list of all the values in the dictionary.
values_view = my_dict.values()
print("Values:", list(values_view))  # Output: ['Alice', 30, 'New York', ['reading', 'hi

# 4. items()
# Returns a view object that displays a list of a dictionary's key-value tuple pairs.
items_view = my_dict.items()
print("Items:", list(items_view))  # Output: [('name', 'Alice'), ('age', 30), ('city', '

# 5. update([other])
# Updates the dictionary with the key-value pairs from `other`, overwriting existing key
my_dict.update({'age': 31, 'profession': 'Engineer'})
print("After update:", my_dict)
# Output: {'name': 'Alice', 'age': 31, 'city': 'New York', 'hobbies': ['reading', 'hikin

# 6. pop(key[, default])
# Removes the specified key and returns its value. If the key is not found, `default` is
removed_value = my_dict.pop('city')
print("Removed 'city':", removed_value)  # Output: New York
print("After pop:", my_dict)  # Output: {'name': 'Alice', 'age': 31, 'hobbies': ['readin

# 7. popitem()
# Removes and returns the last key-value pair as a tuple. Raises KeyError if the diction
last_item = my_dict.popitem()
print("Last item removed:", last_item)  # Output: ('profession', 'Engineer')
print("After popitem:", my_dict)  # Output: {'name': 'Alice', 'age': 31, 'hobbies': ['re

# 8. clear()
# Removes all items from the dictionary.
my_dict.clear()
print("After clear:", my_dict)  # Output: {}

# Reinitialize the dictionary for additional examples
my_dict = {
    'name': 'Bob',
    'age': 25,
    'city': 'Los Angeles'
}

# 9. copy()
# Returns a shallow copy of the dictionary.
dict_copy = my_dict.copy()
```

```
print("Original dictionary:", my_dict)  # Output: {'name': 'Bob', 'age': 25, 'city': 'Lo
print("Copied dictionary:", dict_copy)   # Output: {'name': 'Bob', 'age': 25, 'city': 'L

# 10. setdefault(key[, default])
# Returns the value of `key` if `key` is in the dictionary. If not, inserts `key` with a
default_value = my_dict.setdefault('occupation', 'Unknown')
print("Value of 'occupation':", default_value)  # Output: Unknown
print("After setdefault:", my_dict)  # Output: {'name': 'Bob', 'age': 25, 'city': 'Los A

# 11. fromkeys(iterable[, value])
# Creates a new dictionary with keys from `iterable` and values set to `value` (default
keys = ['a', 'b', 'c']
default_dict = dict.fromkeys(keys, 0)
print("Dictionary fromkeys:", default_dict)  # Output: {'a': 0, 'b': 0, 'c': 0}
```

```
Value for 'name': Alice
Value for 'address': Not Found
Keys: ['name', 'age', 'city', 'hobbies']
Values: ['Alice', 30, 'New York', ['reading', 'hiking', 'coding']]
Items: [('name', 'Alice'), ('age', 30), ('city', 'New York'), ('hobbies', ['reading', 'h
iking', 'coding'])]
After update: {'name': 'Alice', 'age': 31, 'city': 'New York', 'hobbies': ['reading', 'h
iking', 'coding'], 'profession': 'Engineer'}
Removed 'city': New York
After pop: {'name': 'Alice', 'age': 31, 'hobbies': ['reading', 'hiking', 'coding'], 'pro
fession': 'Engineer'}
Last item removed: ('profession', 'Engineer')
After popitem: {'name': 'Alice', 'age': 31, 'hobbies': ['reading', 'hiking', 'coding']}
After clear: {}
Original dictionary: {'name': 'Bob', 'age': 25, 'city': 'Los Angeles'}
Copied dictionary: {'name': 'Bob', 'age': 25, 'city': 'Los Angeles'}
Value of 'occupation': Unknown
After setdefault: {'name': 'Bob', 'age': 25, 'city': 'Los Angeles', 'occupation': 'Unkno
wn'}
Dictionary fromkeys: {'a': 0, 'b': 0, 'c': 0}
```

In [ ]: