Free Notes
CBSE Ready

CodeStepAcademy.in

Inspire • Learn • Excel

Visit codestepacademy.in for more Notes!

DICTIONARY IN PYTHON

Date : _____

Page No. : _____

- A dictionary is an unordered collection of key-value pairs.
- Keys must be unique and immutable (string, number, tuple, etc.).
- Values can be of any data type and can be duplicate.

1 INTRODUCTION

- Dictionaries are written inside curly braces {}.
- Each item is in the form key : value.
- Items are separated by commas.

Example:

```
d = {'name': 'Anaya', 'age': 16, 'grade': 'A'}
print(d)
# Output: {'name': 'Anaya', 'age': 16, 'grade': 'A'}
```

2 ACCESSING ITEMS USING KEYS

- Access values using their keys.

Example:

```
d = {'name': 'Anaya', 'age': 16, 'grade': 'A'}
print(d['name'])      # Anaya
print(d['age'])       # 16
print(d.get('grade')) # A
print(d.get('city'))  # None (no error)
print(d['city'])      # KeyError
```

3 MUTABILITY OF A DICTIONARY

(A) Adding a new item

```
d['city'] = 'Jaipur'  # Add new key-value
print(d)
```

(B) Modifying an existing item

```
d['age'] = 17         # Update existing value
print(d)
```

(C) Deleting an item

```
del d['city']         # Remove key-value pair
print(d)
```

Remember

If you try to access a key that does not exist, it gives KeyError.

4 TRAVERSING A DICTIONARY

(A) Traversing keys

```
d = {'a': 1, 'b': 2, 'c': 3}
for key in d:
    print(key)
# Output:  a  b  c
```

(B) Traversing values

```
for value in d.values():
    print(value)
# Output:  1  2  3
```

(C) Traversing key-value pairs

```
for key, value in d.items():
    print(key, '->', value)
# Output: a -> 1  b -> 2  c -> 3
```

Tip

Use items() when you need both key exist value.

4 BUILT-IN FUNCTIONS / METHODS OF DICTIONARY

No.	Function / Method	Description	Example	Output
1	len(d)	Returns number of items	len(d)	3
2	dict()	Creates a dictionary	dict(a=1, b=2)	{'a': 1, 'b': 2}
3	keys()	Returns all keys	d.keys()	dict_keys(['a', 'b', 'c'])
4	values()	Returns all values	d.values()	dict_values([1, 2,3])
5	items()	Returns key-value pairs	d.items()	dict_items([('a',1),('b',2),('c',3)])
6	get(key, default)	Returns value for key, else default	d.get('a')	1
			d.get('x', 'NA')	'NA'
7	update(other_dict)	Updates with items from other dict	d.update({'b': 20, 'd':4})	{'a':1, 'b':20, 'c':3, 'd':4}
8	del d[key]	Deletes item with given key	del d['a']	{'b':2, 'c':3}
9	clear()	Removes all items	d.clear()	{}
10	fromkeys(seq, value)	Creates dict from keys in seq with value	dict.fromkeys(['x', 'y'], 0)	{'x':0, 'y':0}
11	copy()	Returns a shallow copy	d.copy()	Copy of dictionary
12	pop(key)	Removes item with key and returns value	d.pop('b')	2 (and 'b' removed)
13	popitem()	Removes and returns last inserted item	d.popitem()	('c', 3)
14	setdefault(key, default)	Returns value if key exists,	d.setdefault('a', 10)	1 (existing value)
		else inserts key with default value	d.setdefault('x', 100)	100 (and inserts 'x':100)
15	max(d)	Returns key with max value	max({'a':2, 'b':5, 'c':3})	'b'
	min(d)	Returns key with min value	min({'a':2, 'b':5, 'c':3})	'a'
16	sorted(d)	Returns sorted list of keys	sorted(d)	['a', 'b', 'c']

6 SUGGESTED PROGRAMS

(A) Count the number of times a character appears in a given string

Example:

```
s = "programming is fun"
freq = {}
for ch in s:
    if ch in freq:
        freq[ch] += 1
    else:
        freq[ch] = 1
print(freq)
# Output: {'p': 2, 'r': 2, 'o': 1, 'g': 2, 'a': 1,
#          'm': 2, 'i': 1, 'n': 2, ' ': 2, 's': 1, 'f': 1, 'u': 1}
```

Explanation:

We check each character. If already present in dictionary, increase its count, else add it with count 1.

(B) Create a dictionary with names of employees, their salary and access them

Example:

```
employees = {'Alice': 45000, 'Bob': 52000,
             'Charlie': 48000, 'Diana': 61000}
print(employees['Bob'])          # Access value
employees['Eve'] = 55000         # Add new employee
employees['Alice'] = 47000       # Update salary
del employees['Charlie']         # Remove employee
for name, sal in employees.items():
    print(name, '->', sal)
# Output:
# Bob -> 52000
# Diana -> 61000
# Eve -> 55000
# Alice -> 47000
```

Explanation:

We can access, add, update, delete and traverse data easily.

Exam Tip

Practice dictionary programs regularly. They are very useful in real-life problems.

Key Points:

- ✓ Dictionary stores data in key-value pairs.
- ✓ Keys must be unique and immutable.
- ✓ Use keys(), values(), items() to traverse.
- ✓ Use get() to access safely without KeyError.

Practice More at www.codestepacademy.in for FREE Notes & Worksheets

Scan to Visit Website →