Python Dictionary

Beginner Notes, Examples & Practice Programs

EngineersTutor.com

1. What is a Dictionary?

A dictionary is a built-in Python data type used to store related data in key-value pairs. A dictionary is written using curly braces { }.

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}
```

Here, "name", "age", and "course" are keys. "John", 18, and "Python" are their corresponding values.

Real-life analogy: a real dictionary has word → meaning; a Python dictionary has key → value.

2. Dictionary Syntax

```
dictionary_name = {
    key1: value1,
    key2: value2,
    key3: value3
}
```

3. Creating a Dictionary

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

print(student)
```

```
Output:
{'name': 'John', 'age': 18, 'course': 'Python'}
```

4. Accessing Dictionary Values

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

print(student["name"])
print(student["age"])
print(student["course"])
```

```
Output:
John
```

```
18
Python
```

student["name"] means: find the value associated with the "name" key.

5. Adding a New Item

```
student = {
    "name": "John",
    "age": 18
}

student["course"] = "Python"

print(student)
```

```
Output:
{'name': 'John', 'age': 18, 'course': 'Python'}
```

6. Changing a Value

```
student = {
    "name": "John",
    "age": 18
}

student["age"] = 19

print(student)
```

```
Output:
{'name': 'John', 'age': 19}
```

7. Removing an Item

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

student.pop("age")

print(student)
```

```
Output:
{'name': 'John', 'course': 'Python'}
```

Another way: del student["course"]

8. Checking Whether a Key Exists

```
student = {
    "name": "John",
    "age": 18
}

if "name" in student:
    print("Name is available")
```

```
Output:
Name is available
```

The expression "name" in student checks whether "name" exists as a key.

9. Dictionary Length

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

print(len(student))
```

```
Output:
3
```

len() returns the number of key-value pairs.

10. Looping Through a Dictionary

Print keys:

```
for key in student:
    print(key)
```

Print values:

```
for value in student.values():
    print(value)
```

Print both keys and values:

```
for key, value in student.items():
    print(key, ":", value)
```

11. Useful Dictionary Methods

Method	Purpose
keys()	Returns dictionary keys
values()	Returns dictionary values
items()	Returns key-value pairs
get()	Gets a value using a key
pop()	Removes an item
clear()	Removes all items
update()	Adds or changes items

12. Using get()

get() is useful when a key may be missing.

```
student = {
    "name": "John",
    "age": 18
}

print(student.get("name"))
```

```
Output:
John
```

```
print(student.get("marks"))
```

```
Output:
None
```

student["marks"] raises a KeyError if the key is missing, while student.get("marks") returns None by default.

13. Updating a Dictionary

```
student = {
    "name": "John",
    "age": 18
}

student.update({
    "age": 19,
    "course": "Python"
})

print(student)
```

```
Output:
{'name': 'John', 'age': 19, 'course': 'Python'}
```

14. Clearing a Dictionary

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

student.clear()

print(student)
```

```
Output:
{}
```

15. Simple Practical Program

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python",
    "marks": 85
}

print("Student Information")
print("------------------")
print("Name    :", student["name"])
print("Age     :", student["age"])
print("Course  :", student["course"])
print("Marks   :", student["marks"])
```

```
Output:
Student Information
------------------
Name    : John
Age     : 18
Course  : Python
Marks   : 85
```

16. Dictionary vs List

List	Dictionary
Stores items in a sequence	Stores data as key-value pairs
Uses indexes	Uses keys
Uses [ ]	Uses { }
["Python", "C"]	{"name": "John"}

Remember: List → access using index | Dictionary → access using key

17. Important Points to Remember

• A dictionary uses curly braces { }.

• Data is stored as key-value pairs.

• Keys are used to access values.

• Keys should be unique.

• Dictionary values can have different data types.

• Dictionaries can be modified after creation.

• New items can be added and existing values can be changed.

• pop() and del can remove items; clear() removes all items.

• keys(), values(), and items() are commonly used methods.

• get() can safely retrieve a value when a key may be missing.

• len() returns the number of key-value pairs.

18. Practice Programs

1. Create a dictionary containing your name, age, and city.

2. Print only the name from the dictionary.

3. Add a new "course" key.

4. Change the student's age.

5. Remove the "city" item.

6. Check whether the "course" key exists.

7. Find the number of items in the dictionary.

8. Use a for loop to print all keys.

9. Use a for loop to print all key-value pairs.

10. Use get() to access an existing key and a missing key.

19. Practice Program Solutions

Solution 1 — Create and Print

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "city": "Delhi"
}

print(student)
```

```
Output:
{'name': 'John', 'age': 18, 'city': 'Delhi'}
```

Solution 2 — Print Name

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "city": "Delhi"
}

print(student["name"])
```

```
Output:
John
```

Solution 3 — Add a New Item

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "city": "Delhi"
}

student["course"] = "Python"

print(student)
```

```
Output:
{'name': 'John', 'age': 18, 'city': 'Delhi', 'course': 'Python'}
```

Solution 4 — Change a Value

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "city": "Delhi"
}

student["age"] = 19

print(student)
```

```
Output:
{'name': 'John', 'age': 19, 'city': 'Delhi'}
```

Solution 5 — Remove an Item

Practice solution:

```
student = {
    "name": "John",
```

```
    "age": 18,
    "city": "Delhi"
}

student.pop("city")

print(student)
```

```
Output:
{'name': 'John', 'age': 18}
```

Solution 6 — Check Whether a Key Exists

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

if "course" in student:
    print("Course is available")
```

```
Output:
Course is available
```

Solution 7 — Find Dictionary Length

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

print(len(student))
```

```
Output:
3
```

Solution 8 — Print All Keys

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

for key in student:
    print(key)
```

```
Output:
name
age
course
```

Solution 9 — Print Key-Value Pairs

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

for key, value in student.items():
    print(key, ":", value)
```

```
Output:
name : John
age : 18
course : Python
```

Solution 10 — Use get()

Practice solution:

```
student = {
    "name": "John",
    "age": 18,
    "course": "Python"
}

print(student.get("course"))
print(student.get("marks"))
```

```
Output:
Python
None
```

20. Quick Dictionary Cheat Sheet

Task	Python
Create dictionary	student = {}
Access value	student["name"]
Add item	student["city"] = "Delhi"
Change value	student["age"] = 19
Remove item	student.pop("age")
Delete item	del student["age"]
Check key	"name" in student
Dictionary length	len(student)
Get value safely	student.get("name")
Get keys	student.keys()
Get values	student.values()
Get both	student.items()
Update dictionary	student.update(...)
Remove everything	student.clear()

Dictionary = Key → Value

Create → Access → Add → Change → Remove → Check → Loop → Practice