Dictionaries in Python

Introduction

- A **dictionary** is an **unordered, mutable** collection of **key-value pairs**.
- Defined using **curly braces {}**.
- Keys must be **unique and immutable** (e.g., strings, numbers, tuples).
- Values can be of **any data type**.

```
person = {
   "name": "Amit",
   "age": 21,
   "city": "Delhi"
}
```

Accessing Items Using Keys

- Use square brackets [] or the get() method.

```
print(person["name"])     # Amit
print(person.get("age"))  # 21
```

Mutability of a Dictionary

➢ **Adding a New Item**

```
person["gender"] = "Male"
```

➢ **Modifying an Existing Item**

```
person["age"] = 22
```

Traversing a Dictionary

```
for key in person:
   print(key, "→", person[key])
```

Or use .items():

```
for k, v in person.items():
   print(k, ":", v)
```

Built-in Functions / Methods

Function / Method	Description	Example
len()	Returns number of key-value pairs	len(d)
dict()	Creates a new dictionary	dict(a=1, b=2) → {'a':1, 'b':2}
keys()	Returns all keys	d.keys()
values()	Returns all values	d.values()
items()	Returns key-value pairs	d.items()
get(key)	Returns value for key; returns None if absent	d.get('age')
update()	Adds or updates key-value pairs from another dict	d.update({'age': 25})
del	Deletes a key-value pair	del d['age']
clear()	Removes all items	d.clear()
fromkeys(seq)	Creates new dict from keys in seq	dict.fromkeys(['a','b'], 0)
copy()	Returns a shallow copy	d.copy()
pop(key)	Removes key and returns its value	d.pop('name')
popitem()	Removes and returns the last inserted item	d.popitem()
setdefault()	Returns value of key or inserts default if absent	d.setdefault('lang', 'Python')
max()	Maximum of keys	max(d)
min()	Minimum of keys	min(d)
sorted()	Sorted list of dictionary keys	sorted(d)

1. Count Characters in a String Using Dictionary

```
string = "banana"
freq = {}

for char in string:
    if char in freq:
        freq[char] += 1
    else:
        freq[char] = 1

print(freq)
# Output: {'b': 1, 'a': 3, 'n': 2}
```

2. Create a Dictionary of Employee Names and Salaries

```
employees = {
    "Ravi": 30000,
    "Neha": 45000,
    "Amit": 50000
}

# Access salary
print("Salary of Neha:", employees["Neha"])

# Add a new employee
employees["Sita"] = 40000

# Update salary
employees["Amit"] = 55000

# Traverse the dictionary
for name, salary in employees.items():
    print(name, "→", salary)
```