Python `f-strings` (formatted string literals)

Python introduced `f-strings` in version 3.6 as a concise and efficient way to format strings. They are denoted by a leading `f` or `F` before the string and allow embedding expressions inside curly braces `{}`.

---

1. Basic String Interpolation

You can directly insert variables into a string.

```
name = "Akshat"

age = 21

greeting = f"My name is {name} and I am {age} years old."

print(greeting)
```

Output:

```
My name is Akshat and I am 21 years old.
```

---

2. Inline Expressions

You can perform calculations or call functions directly inside the braces.

```
a = 10

b = 20

result = f"The sum of {a} and {b} is {a + b}."

print(result)
```

Output:

```
The sum of 10 and 20 is 30.
```

---

3. Formatting Numbers

`f-strings` support formatting options like decimals, alignment, and padding.

```
pi = 3.14159

formatted = f"Pi rounded to 2 decimal places is {pi:.2f}."

print(formatted)
```

Output:

```
Pi rounded to 2 decimal places is 3.14.
```

- **`:2f`**: Rounds to 2 decimal places.
- You can use other formatting codes, like `.3f` for 3 decimal places, or add commas for large numbers.

---

4. Using Dictionaries and Objects

You can access dictionary keys or object attributes.

```
data = {"name": "Akshat", "age": 21}

message = f"{data['name']} is {data['age']} years old."

print(message)
```

Output:

```
Akshat is 21 years old.
```

For objects:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

person = Person("Akshat", 21)
info = f"{person.name} is {person.age} years old."
print(info)
```

Output:

```
Akshat is 21 years old.
```

---

5. Multiline and Raw f-Strings

Multiline f-strings:

You can use `f-strings` in multiline strings by wrapping them in triple quotes.

```
name = "Akshat"
profession = "developer"
bio = f"""
Hello, my name is {name}.
I am a {profession}.
"""
print(bio)
```

Output:

Hello, my name is Akshat.

I am a developer.

Raw f-strings:

Use `r` before the string to handle escape sequences.

path = "C:\\Users\\Akshat"

formatted_path = rf"The file is located at {path}"

print(formatted_path)

Output:

The file is located at C:\Users\Akshat

Summary of f-String Features:

- Embed variables and expressions directly using `{}`.
- Format numbers, align text, and apply padding using format specifiers.
- Access dictionary keys and object attributes seamlessly.
- Combine with triple quotes for multiline formatting.
- Use `rf""` for raw f-strings to avoid escape character issues.

These examples cover the essentials and versatility of `f-strings` in Python. Let me know if you'd like further clarification!