

# Input/Output, String Manipulation, and Comments

---

## 1. Input and Output in Python

### 1.1 Input from the User:

In Python, we use the `input()` function to take input from the user. The data entered by the user is always received as a string, so if you want to use it as a different data type (e.g., integer or float), you'll need to convert it using type conversion functions like `int()` or `float()`.

```
name = input("Enter your  
age = int(input("Enter yc
```



## 🔗 1.2 Output to the Console:

The `print()` function is used to display output to the console. You can use it to display text, variables, or results of expressions.

```
print("Hello, " + name +
```

---

You can also use **f-strings** (formatted string literals) for more readable code:

```
print(f"Hello, {name}! Yc
```

---

## 🔗 2. String Manipulation

Strings are sequences of characters. Python provides many useful methods to manipulate strings.

### 🔗 2.1 Common String Operations:

- **Concatenation:** Joining two or more strings together using the `+` operator.

```
first_name = "John"
last_name = "Doe"
full_name = first_name + last_name
print(full_name) # 0
```

---


- **Repetition:** Repeating a string multiple times using the `*` operator.

```
greeting = "Hello! "
print(greeting * 5) # 0
```

---

## 🔗 2.2 String Methods:

- `upper()` : Converts a string to uppercase.
- `lower()` : Converts a string to lowercase.
- `strip()` : Removes leading and trailing spaces from a string.
- `replace(old, new)` : Replaces a substring with another string.

```
message = " Hello, World"   
print(message.strip()) #  
print(message.upper()) #  
print(message.replace("Wc
```

---

## 🔗 2.3 Accessing String Characters:

You can access individual characters in a string using **indexing**. Python uses zero-based indexing, so the first character has an index of 0.

```
text = "Python"
print(text[0]) # Output:
print(text[2]) # Output:
```

---



You can also use **negative indexing** to start counting from the end of the string.


```
print(text[-1]) # Output
print(text[-3]) # Output
```

---



## 2.4 Slicing Strings:

You can extract a portion (substring) of a string using slicing.

```
text = "Python Programming"   
print(text[0:6]) # Output  
print(text[:6]) # Output  
print(text[7:]) # Output
```

---



## 🔗 3. Comments in Python

Comments are ignored by the Python interpreter and are used to explain the code or leave notes for yourself or others. They do not affect the execution of the program.

- **Single-line comments** start with `#`:

```
# This is a single-line comment
print("Hello, World!")
```

---

- **Multi-line comments** can be written using triple quotes ( `"""` or `'''` ). These are often used to write detailed explanations or temporarily block sections of code:

```
"""
This is a multi-line comment
It can span multiple lines
"""
print("Hello, Python!")
```

---


## 🔗 4. Escape Sequences

Escape sequences are special characters in strings that start with a backslash ( \ ). They are used to represent certain special characters.

Some commonly used escape sequences:

- \n : New line
- \t : Tab space
- \\ : Backslash

### 🔗 Example:

```
print("Hello\nWorld")  #   
# Hello  
# World  
  
print("Hello\tPython")  #
```


---



## 🔗 Homework

1. **Simple Greeting Program:** Write a Python program that asks the user for their name and age, then prints a personalized greeting message. Use both the `+` operator and f-strings for output.

**Example:**

```
Enter your name: Alic   
Enter your age: 25  
Output: Hello, Alice!
```

---

2. **String Manipulation Exercise:**  
Write a Python program that:

- Takes a sentence as input from the user.
- Prints the sentence in all uppercase and lowercase.
- Replaces all spaces with underscores.
- Removes leading and trailing whitespace.

### 3. Character Counter: Write a Python program that:

- Asks the user for a string.
- Prints how many characters are in the string, excluding spaces.

#### Example:

```
t: "Hello World"  
lt: "Number of characte
```



### 4. Escape Sequence Practice:

Write a Python program that uses escape sequences to print the following output:

#### Example:

```
Hello  
    World  
This is a backslash:
```

