Printing in Python

A Comprehensive Tutorial

Table of contents

01

Basic Usage of ‘print()’

Basic Usage of ‘print()’

The **print()** function in Python is a built-in function used to **output** text or other data types to the console.

It’s one of the most commonly used functions for **debugging**, **displaying results**, or **providing user feedback**.

The most straightforward use of the **print()** function is to output text or variables to the console.

Example

Python

```
print("Hello, World!")
print(42)
print(3.14)
```

Output

```
Hello, World!
42
3.14
```

02

Printing Multiple Items

Printing Multiple Items

You can print multiple items **separated by commas**. The **print()** function automatically separates them with a space.

Python

```
print("The value of x is", 42, "and the value of y is", 3.14)
```

Output

```
The value of x is 42 and the value of y is 3.14
```

Printing Multiple Times

The **multiplication Operator (*)** operator repeats the string the specified number of times.

Python

```
print("Hello" * 3)
print("=" * 10)
```

Output

```
HelloHelloHello
==========
```

"Hello" * 3: This prints **“Hello”** three times.

"=" * 10: This prints **“=“** ten times. You can use this to create **visual separation** in console display.

03

Separator (sep) Parameter

Separator (sep) Parameter

The **sep** parameter allows you to define a **custom separator** between the items you are printing. **Default** separator is **single space**.

Python

```
print("apple", "banana", "cherry", sep=“ | ")
print("apple", "banana", "cherry", sep=", ")
print("apple", "banana", "cherry", sep=“ → ")
```

Output

```
apple | banana | cherry
apple, banana, cherry
apple → banana → cherry
```

04

End (end) Parameter

End (end) Parameter

The end parameter lets you specify what is printed at the end of the output. By default, print() ends with a newline character (\n).

Python

```
print("Hello, World!", end=" ")
print("This is printed on the same line.")
```

Output

```
Hello, World! This is printed on the same line.
```

05

Printing with Escape Characters

Printing with Escape Characters

Escape characters are special characters that are used to insert whitespace, newline, or other control characters within a string.

Python

```
print("Hello,\nWorld!") # \n: Inserts a newline
print("Tab\tseparated") # \t: Inserts a tab space
print("Backslash: \\")  # \\: Inserts a backslash
```

Output

```
Hello,
World!
Tab    separated
Backslash: \
```

06

Printing with Format Strings

Printing with Format Strings

Python provides several ways to format strings for printing.

1. Using '%' Operator

Python

```
name = “Gaurav"
age = 30
print("Name: %s, Age: %d" % (name, age))
```

Output

```
Name: Gaurav, Age: 30
```

Explanation:

- **%s:** Placeholder for **strings**.
- **%d:** Placeholder for **integers**.

Printing with Format Strings

Python provides several ways to format strings for printing.

2. Using str.format() Method

Python

```
name = “Nitin"
age = 25
print("Name: {}, Age: {}".format(name, age))
```

Output

```
Name: Nitin, Age: 25
```

Explanation:

- **"Name: {}, Age: {}":** This is a string template that contains placeholders {} where the values of name and age will be inserted.
- **.format(name, age):** Provides the values that is to be placed in theplaceholders

Printing with Format Strings

Python provides several ways to format strings for printing.

3. Using f-Strings

Python

```
name = “Anvesha"
age = 22
print(f"Name: {name}, Age: {age}")
```

Output

```
Name: Anvesha, Age: 22
```

Explanation:

- **"Name: {name}, Age: {age}":** This is an f-string, which allows you to embed expressions inside a string by prefixing the string with the letter **f**.
- **{name} :** Placeholder for name
- **{age}:** Placeholder for age

07

Printing in a tabular structure

Advanced String Formatting with format()

Python

```
headers = ["Name", "Age", “State"]
data = [[“Anvesha", 30, “Mumbai"], [“Nitin", 25, “Punjab"], [“Amar", 22, “Kerala"]]

# Print headers and rows with a fixed width
print("{:<10} {:<10} {:<15}".format(*headers))
print("-" * 35)
for row in data:
    print("{:<10} {:<10} {:<15}".format(*row))
```

Output

```
Name       Age        City
-----------------------------------
Anvesha    30         Mumbai
Nitin      25         Punjab
Amar       22         Kerala
```

Explanation

{:<10}: Aligns the text to the **left** and *sets the **width*** of the *column to **10** characters*.
{:<15}: Aligns the text to the **left** and sets the ***width*** *of the column to **15** characters*.

Knowledge Reinforcement

1

Question

What is the **default separator** used when printing multiple items using the **print() function**?

Answer

a) Comma (,)

b) Single Space (‘ ‘)

c) Newline (\n)

d) Tab (\t)

1

Question

What is the **default separator** used when printing multiple items using the **print() function**?

Answer

a) Comma (,)

b) Single Space (‘ ‘)

c) Newline (\n)

d) Tab (\t)

2

Question

How do you print a string **without** adding a newline at the end?

Answer

a) Use print("Hello", sep="")

b) Use print("Hello", end="")

c) Use print("Hello\n")

d) Use print("Hello", no_newline=True)

2

Question

How do you print a string **without** adding a newline at the end?

Answer

a) Use print("Hello", sep="")

b) Use print("Hello", end="")

c) Use print("Hello\n")

d) Use print("Hello", no_newline=True)

3

Question

What is the output of the following code?

```python
print("Hello", "World", sep="-")
```

Answer

a) Hello-World

b) Hello World

c) Hello- World

d) Hello\nWorld

3

Question

What is the output of the following code?

```python
print("Hello", "World", sep="-")
```

Answer

a) Hello-World

b) Hello World

c) Hello- World

d) Hello\nWorld

4

Question

What is the output of the following code?

Python

```
name = "Dhruv"
age = 25
print(f"Name: {name}, Age: {age}")
```

Answer

a) Name: name, Age: age

b) Name: Dhruv, Age: 25

c) Name: Dhruv Age: 25

d) Name: name Age: age

4

Question

What is the output of the following code?

Python

```
name = “Dhruv"
age = 25
print(f"Name: {name}, Age: {age}")
```

Answer

a) Name: name, Age: age

b) Name: Dhruv, Age: 25

c) Name: Dhruv Age: 25

d) Name: name Age: age

5

Question

Which escape character is used to insert a **newline** in the output of the **print() function**?

Answer

a) \t

b) \\

c) \n

d) \b

5

Question

Which escape character is used to insert a **newline** in the output of the **print() function**?

Answer

a) \t

b) \\

c) \n

d) \b

6

Question

What does the **sep** parameter in the **print()** function do?

Answer

a) Specifies the character to be printed at the end of the output.

b) Specifies the character to separate multiple values.

c) Specifies the format of the output.

d) Specifies the number of times to repeat the print statement.

6

Question

What does the **sep** parameter in the **print()** function do?

Answer

a) Specifies the character to be printed at the end of the output.

b) Specifies the character to separate multiple values.

c) Specifies the format of the output.

d) Specifies the number of times to repeat the print statement.

7

Question

What is the output of the following code?

Python

```
name = "Dhruv"
age = 25
print("Name: {}, Age: {}".format(name, age))
```

Answer

a) Name: Dhruv, Age: 25

b) Name: {}, Age: {}

c) Name: name, Age: age

d) Name: , Age:

7

Question

What is the output of the following code?

Python

```
name = "Dhruv"
age = 25
print("Name: {}, Age: {}".format(name, age))
```

Answer

a) Name: Dhruv, Age: 25

b) Name: {}, Age: {}

c) Name: name, Age: age

d) Name: , Age:

8

Question

How will you write “I will not talk in the class” 500 times?

Answer

a) print("I will not talk in the class" * 500)

b) print("I will not talk in the class\n" * 500)

c) print("I will not talk in the class" + 500)

d) print("I will not talk in the class\n * 500”)

8

Question

How will you write “I will not talk in the class” 500 times?

Answer

a) print("I will not talk in the class" * 500)

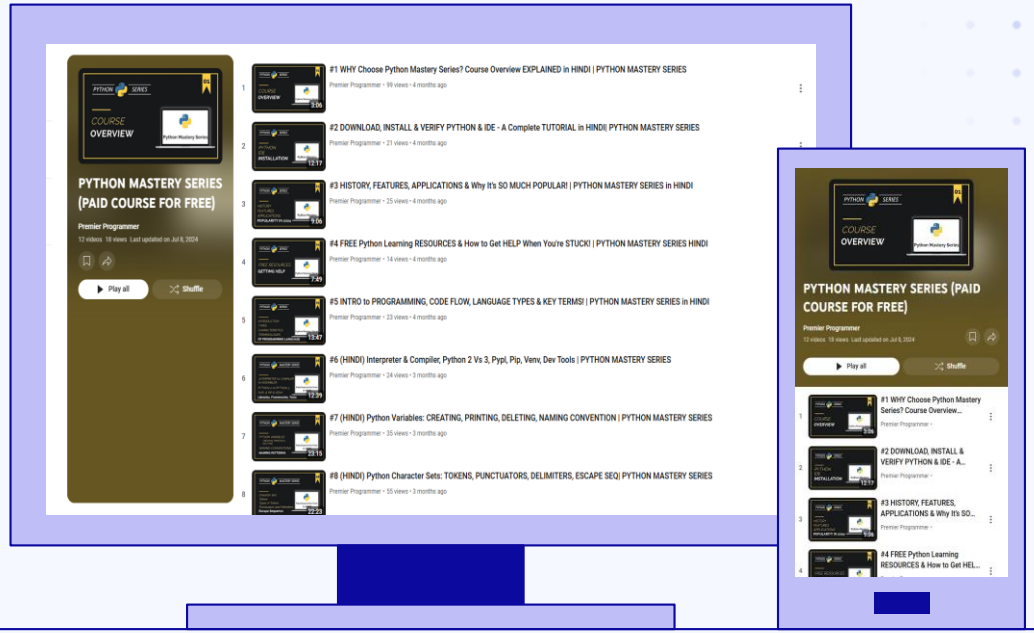
b) print("I will not talk in the class \ n" * 500)

c) print("I will not talk in the class" + 500)

d) print("I will not talk in the class\n * 500”)

WATCH

Level up your coding with each episode in this focused Python series.



Next Video!

Indentation

Comments & Docstrings

Scopes and rules