

Table of the Contents

Module 1

- [What is Python](#)
- [Getting Started](#)
- [How To Use Print\(\) Function in Python](#)

Module 2

- [Variables](#)
- [Types of Data](#)

Module 3

- [Operators](#)
- [Input and Output](#)

Module 4

- [if statement](#)
- [for loop](#)

Module 5

- [while loop](#)

What is Python

- Python is a high-level, general-purpose programming language.
- It is known for its clear syntax, readability, and versatility.
- Python is widely used for [web development](#), [data science](#), [machine learning](#), and [automation](#).

Getting Started

- Install Python: Download and install it from <https://www.python.org/downloads/>.
- Choose a text editor: A program to write code, like [Visual Studio Code](#), [Jupyter Notebook](#), [PyCharm](#), or even a simple text editor like [Notepad](#).
- Text editor for Android: [Pydroid 3 - IDE for Python 3](#)
 - [Video: How to: Install Jupyter Notebook on an Android device](#)
- Interactive mode: Experiment with Python directly in your terminal or command prompt using the python command.

Important: Python source code files always use the [.py](#) extension.

How To Use Print() Function in Python

- It is used to display text to the console, or to a file. The print() function can take one or more arguments, and it can be used to format text in a variety of ways.

Example #1:

```
message = 'Python is fun'  
  
# print the string message  
print(message)
```

Output:

```
Python is fun
```

Example #2:

```
# Print a string:  
print("Hello, World!")  
  
# Print a number:  
print(10)  
  
# Print a variable:  
x = 5  
print(x)  
  
# Print multiple objects on the same line:  
print("Hello", "World")  
  
# Print multiple objects on separate lines:  
print("Hello")  
print("World")  
  
# Print with a custom separator:  
print("Hello", "World", sep=", ")  
  
# Print with a custom ending character:  
print("Hello", "World", end="!")
```

See also:

- [Video: How to print multiple lines](#)
- [Video: 100 times "hello world" without loop](#)

[Learn more...](#)

Comments

Comments

- Comments are important for making code more readable and understandable, especially for other programmers who may need to read or modify the code.
- Comments in Python are non-executable lines of code and ignored by the Python interpreter when the code is executed.
- **Single-line comments:** These comments start with the hash symbol (#) and extend to the end of the line.

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

[Learn more...](#)

Variables

- Storage containers for data (numbers, text, etc.).

What is a variable

- A variable is a named storage location in a computer's memory that is used to hold data or values. It allows programmers to store and manipulate data within a program.

Purpose: Variables provide a way to store and manage data that can be used and manipulated throughout a program. They make programs more flexible and allow for dynamic data storage.

Assignment statement: in Python is used to assign a value to a variable. Its primary purpose is to store and manage data within a program.

Imagine variables as labeled boxes:

- You have boxes for storing different things (numbers, words, etc.).
- Each box has a name (label) to identify what's inside.
- You can put things in, take them out, and change what's inside.

Example #3: Storing a name

```
name = "Muhammad Hamza"  
print(name)
```

Example #4: Tracking a score:

```
score = 0  
score = score + 10 # adds 10 to the score  
print(score)
```

Example #5: Remembering a favorite color

```
favorite_color = "blue" #stores "blue" in variable
print(favorite_color)
```

Example #6: Calculating the area of a rectangle

```
length = 10
width = 5

# calculates the area
area = length * width
print(area)
```

Key Points:

- **Choose meaningful names:** Use names that describe what the variable stores (e.g., pizza_slices instead of x).
- **Assign values using =:** The equals sign is used to put a value into a variable.
- **Change values:** You can update a variable's value later in your code.
- **Use variables in calculations and operations:** Variables can be used just like regular numbers or text in expressions.
- **Think of variables as placeholders:** They hold information that can change as your program runs.

See also:

- [Variables in Python](#)

[Learn more...](#)

Types of Data

- Numbers (integers, floats), strings (text), booleans (True/False)

String (str)

- Stores text or characters.
- Enclosed in single or double quotes.

Example #7:

```
name = "Ahmad"
greeting = 'Hello, world!'
favorite_song = "Let It Go" # Double quotes can also contain single quotes
```

Integer (int)

- Stores whole numbers (without decimals).
- Used for counting and representing quantities.

Example #8:

```
age = 25
num_pets = 3
lucky_number = 7
```

Float (float)

- Stores decimal numbers.
- Used for representing precise measurements or values with fractions

Example #9:

```
height = 1.70 # meters
price = 19.99
distance = 3.14159 # Pi
```

Boolean (bool)

- Stores True or False values.
- Used for representing logical conditions or decisions.

Example #10:

```
is_hungry = True
is_raining = False
has_finished = True
```

Key Points

- Each data type has specific purposes and operations.
- Python automatically determines the data type when you assign a value.
- You can check the data type using the `type()` function:

```
print(type(name)) # Output: <class 'str'>
print(type(age)) # Output: <class 'int'>
print(type(height)) # Output: <class 'float'>
print(type(is_hungry)) # Output: <class 'bool'>
```

See also:

- [Video: Use of type\(\) function in Python](#)

[Learn more...](#)

Operators

- Perform calculations and comparisons (e.g., +, -, *, /, ==, !=).

1. Arithmetic Operators:

- Used for performing basic mathematical calculations.
- Operators: + (addition), - (subtraction), * (multiplication), / (division), // (floor division), % (modulo), ** (exponentiation)

Example #11:

```
result = 10 + 5 # Addition
difference = 15 - 7 # Subtraction
product = 4 * 6 # Multiplication
quotient = 12 / 3 # Division
integer_quotient = 17 // 4 # Floor division
remainder = 25 % 4 # Modulo
square = 5 ** 2 # Exponentiation
```

2. Comparison Operators:

- Used to compare values and return a Boolean result (True or False).
- Operators: == (equal to), != (not equal to), > (greater than), < (less than), >= (greater than or equal to), <= (less than or equal to)

Example #12:

```
is_equal = 7 == 7 # True
is_greater = 12 > 9 # True
is_less_or_equal = 5 <= 5 # True
```

[Learn more..](#)

Input and Output

- Get user input with input() and display output with print().

Input in Python:

- **The input() function:**
 - Takes a prompt as an argument (optional)
 - Reads user input from the keyboard
 - Returns the input as a string

- **Example #13:**

```
name = input("Enter your name: ")
age = int(input("Enter your age: ")) # Convert string input to integer
```

Output in Python:

- The **print()** function:
 - Prints values to the console
 - Can display multiple values, separated by commas
 - Can format output using string methods
- **Examples:**

```
print("Hello, world!")
print("Your name is", name, "and you are", age, "years old.")
print("The answer is:", 42)
```

[Learn more ...](#)

Control Flow

Conditional Statements

if statement

- The if statement in MATLAB is a conditional statement that allows you to execute a block of code only if a certain condition is met.
- Make decisions using **if**, **elif**, and **else** statements.

The general **syntax** of the if statement is as follows:

```
if condition
    statements
```

The **condition** can be any logical expression. If the condition is evaluated to **true**, the block of **statements** is executed. Otherwise, the block of **statements** is skipped.

Here is a simple example of an if statement in MATLAB:

Example #14:

```
x = 10
```

```
if x > 5:  
    print('x is greater than 5.')
```

This code will print the message `x is greater than 5.` to the console.

You can also use `elif` statements to check for multiple conditions. The general **syntax** of the **elif statement** is as follows:

```
elif condition  
    statements  
end
```

If the `condition` for the if statement is evaluated to `false`, the python interpreter will check the `condition` for the first `elif` statement. If the condition for the `elif` statement is evaluated to `true`, the corresponding block of `statements` is executed. Otherwise, the python interpreter will check the `condition` for the next `elif` statement, and so on.

Here is an example of an if statement with an `elif` statement:

Example #15:

```
x = 3  
  
if x > 5:  
    print('x is greater than 5.')
```

```
elif x < 5:  
    print('x is less than 5.')
```

This code will print the message "x is less than 5." to the console.

You can also use an `else` statement to check for all other conditions. The general syntax of the `else` statement is as follows:

```
else  
    statements  
end
```

If all of the conditions for the if and `elseif` statements are evaluated to `false`, the block of `statements` in the `else` statement is executed.

Here is an example of an if statement with an `elif` statement and an `else` statement:

Example #16:


```
x = 2

if x > 5:
    print('x is greater than 5.')
elif x == 5:
    print('x is equal to 5.')
else:
    print('x is less than 5.')
```

This code will print the message "x is less than 5." to the console.

[Learn more...](#)

Loops

- Repeat actions using **for** and **while** loops.

for loop

- A for loop in Python is a programming statement that repeats a block of code a certain number of times.

Example #17:

```
for i in range(5):
    print(i)
```

Example #18:

```
for i in range(5):
    print("Python")
```

See also:

- [Video: Calculate the sum of the first N natural numbers](#)
- [Video: String as an iterable using for loop](#)

[Learn more...](#)

while loop

- A while loop in python is a control flow statement that repeatedly executes a block of code until a specified condition is met.

Example #19: Counting Up to a Number:

```
count = 1 # Start counting at 1
while count <= 10: # Keep counting as long as we're less than or equal to 10
    print(count) # Print the current number
    count += 1 # Add 1 to the count for the next round
```

[Learn more...](#)

Multiple Choice (Select the best answer)

Which of the following is the correct syntax for the print statement in Python?

- ☐ print ("text")
- ☐ println ("text")
- ☐ echo ("text")

Which of the following statements will print the value of the variable x?

- ☐ print(x)
- ☐ print "x"
- ☐ println(x)
- ☐ echo x

What will be the output of the following code?

```
print("Hello, world!")
```

- ☐ Hello
- ☐ world
- ☐ Hello, world!
- ☐ There will be no output

What is the purpose of the sep argument in the print function?

- ☐ To specify the separator between multiple values printed on the same line.
- ☐ To specify the end character for the printed line.
- ☐ To specify the file to which the output should be printed.
- ☐ To specify the format of the output.

What is the purpose of the end argument in the print function?

- ☐ To specify the separator between multiple values printed on the same line.
- ☐ To specify the end character for the printed line.
- ☐ To specify the file to which the output should be printed.
- ☐ To specify the format of the output.

What is the primary purpose of comments in Python code?

- ☐ To execute instructions for the computer

2. ☐ To temporarily disable lines of code
3. ☐ To make the code more readable and understandable for humans
4. ☐ To create errors for debugging

Which of the following is the correct syntax for a single-line comment in Python?

1. ☐ `// This is a comment`
2. ☐ `/* This is a comment */`
3. ☐ `# This is a comment`
4. ☐ `{ This is a comment }`

What is the difference between `==` and `=` in Python?

1. ☐ `==` is the comparison operator, `=` is the assignment operator
2. ☐ `==` is the assignment operator, `=` is the comparison operator
3. ☐ They are the same operator
4. ☐ There is no difference

Which operator checks if two values are greater than or equal to each other?

1. ☐ `>`
2. ☐ `<=`
3. ☐ `>=`
4. ☐ `<`

What is the result of `5 <= 5`?

1. ☐ True
2. ☐ False
3. ☐ 5
4. ☐ None of the above

What is the output of `x = 10; x //= 3`?

1. ☐ 3
2. ☐ 3.33
3. ☐ 3.5
4. ☐ 4

Which operator is used to raise a number to a power?

1. ☐ `**`
2. ☐ `^`
3. ☐ `pow()`
4. ☐ `**` and `^` are both correct

What is the output of `45 // 7`?

1. ☐ 5.0
2. ☐ 6
3. ☐ 5

4. ☐ 6.428571428571429

What is the difference between == and = in Python?

1. ☐ == is the comparison operator, = is the assignment operator
2. ☐ == is the assignment operator, = is the comparison operator
3. ☐ They are the same operator
4. ☐ There is no difference

What is the result of the expression 3 + 5 * 2 in Python?

1. ☐ 16
2. ☐ 13
3. ☐ 11
4. ☐ 26

Which data type is used to represent decimal numbers in Python?

1. ☐ int
2. ☐ float
3. ☐ complex
4. ☐ str

Which of the following is an example of a boolean value in Python?

1. ☐ "True"
2. ☐ 1
3. ☐ 3.14
4. ☐ False

Which scalar data type is used to represent textual data in Python?

1. ☐ str
2. ☐ char
3. ☐ text
4. ☐ string

What is the default type of a numerical literal without a decimal point in Python?

1. ☐ int
2. ☐ float
3. ☐ complex
4. ☐ bool

Which of the following is NOT a scalar data type in Python?

1. ☐ Integer
2. ☐ Float
3. ☐ String
4. ☐ List

What is the output of type(42)?

1. ☐ int
2. ☐ float
3. ☐ str
4. ☐ None

What is the result of 3 + 4.5?

1. ☐ 7
2. ☐ 7.5
3. ☐ Error
4. ☐ None of the above

How do you create a string in Python?

1. ☐ Using single quotes ('')
2. ☐ Using double quotes ("")
3. ☐ Both a and b
4. ☐ None of the above

What is the result of 5 > 3 and not (2 == 2)?

1. ☐ True
2. ☐ False
3. ☐ Error
4. ☐ None of the above

What is the correct syntax for a for loop in Python?

1. ☐ for (int i = 0; i < 10; i++):
2. ☐ for i in range(10):
3. ☐ for i = 0 to 9:
4. ☐ for i in 10:

What will be the output of the following code?

```
for i in range(5):  
    print(i * 2)
```

1. ☐ 0 1 2 3 4
2. ☐ 2 4 6 8 10
3. ☐ 10 8 6 4 2
4. ☐ 0 2 4 6 8

What is the output of the following code?

```
count = 0  
while count < 3:  
    print(count)  
    count += 1
```

1. ☐ 0 1 2
2. ☐ 0 1
3. ☐ 1 2 3
4. ☐ The code will run indefinitely.

Which of the following correctly represents the syntax of an if statement in Python?

1. ☐ if condition { block of code }
2. ☐ if(condition) { block of code }
3. ☐ if condition: block of code

What is the purpose of the else block in an if statement?

1. ☐ To execute a code block when the if condition is True
2. ☐ To execute a code block when the if condition is False
3. ☐ To create an infinite loop
4. ☐ To define a function

What will be the output of the following code?

```
x = 10
y = 5
if x < y:
    print("x is greater than y")
```

1. ☐ "x is greater than y"
2. ☐ "x is less than y"
3. ☐ Nothing will be printed
4. ☐ An error will occur

Review Questions

- What is a variable?
- How do you display the text "Hello, world!" in the console?
- How do you ask the user to enter their name and store it in a variable?
- What are the operators used for addition, multiplication, floor division, and exponentiation in Python and provide an example of each operation?
- What is the difference between the division operator (/) and the floor division operator (//)?
- What is the modulus operator (%) used for?
- What are the operators used to compare values (e.g., greater than, less than, equal to)?
- How do you create a variable in Python?
- What are the common data types in Python?

Control Flow:

- How do you create a for loop that iterates from 1 to 10 using the range() function?
- What is the purpose of an if statement?

- What is the difference between a for loop and a while loop in Python?

References and Bibliography

- Automate the Boring Stuff with Python - Practical Programming for Total Beginners by Al Sweigart [website](#) | [ebook](#) (free to read)
- Python One-Liners by Christian Mayer teaches you how to read and write "one-liners": concise statements of useful functionality packed into a single line of code. | [website with free one-liner explainer videos](#)
- [Python for Data Analysis, 3E](#) | [ebook](#) (free to read)
- [Python Programming And Numerical Methods: A Guide For Engineers And Scientists](#)
- [Google's Python Class](#)
- [Python Code Examples – Sample Script Coding Tutorial for Beginners](#)
- [Python Tutorial - W3Schools](#)
- [Python for beginners - Training | Microsoft Learn](#)