

BUREAUTIQUE

Python Programming

Prepared by:

Nassim KADDOURI

Academic Year: 2024-2025

NUMIDIA INSTITUTE OF TECHNOLOGY

Chapter 1

Introduction to Python

What is Python?

Python is a popular programming language known for its simplicity and readability. It is used for various tasks, from web development and data analysis to artificial intelligence and automation. Python is a great language for beginners because its **syntax** (the rules for writing code) is easy to learn and understand.

Here is a simple example of a Python program:

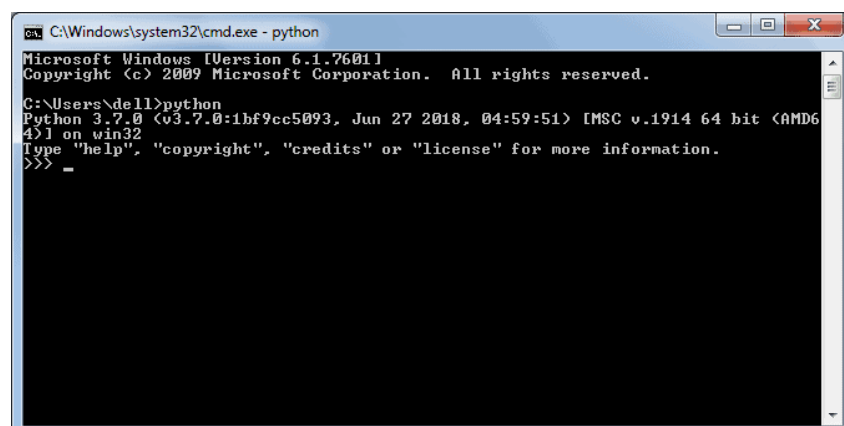
```
# A simple program to greet the user
name = input("Enter your name: ")
print("Hello, " + name + "!")
```

In this example, the program asks for your name and then greets you.

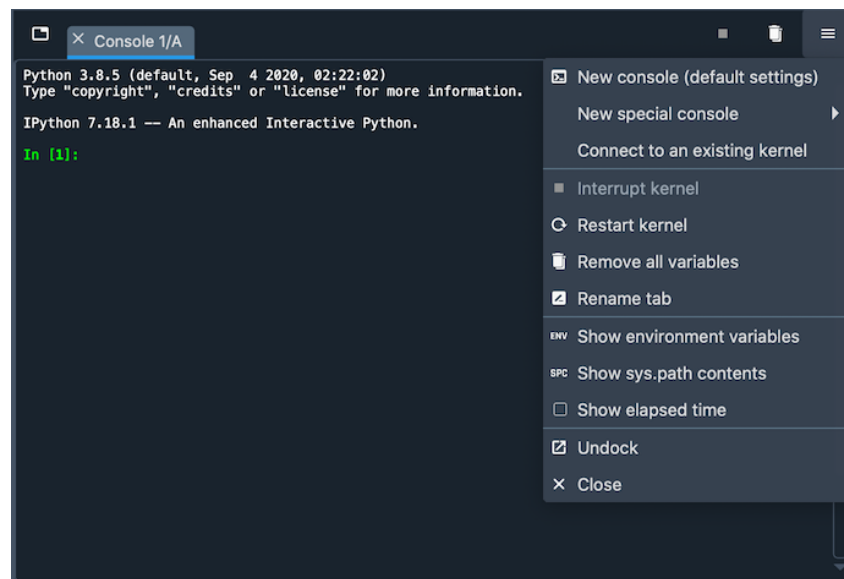
What is a Console?

The console, also known as the command line or terminal, is a text-based interface used to run programs and display output.

Here is an example of what a console might look like on windows:



Here is an example of what a console might look like on Spyder:



When you write a Python program, you can run it in the console to see the results.

1.1 My First Python Code!

- **Example:**

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

- **Output:**

```
Hello , World!
```

- **Description:**

- The ritual for programmers when they start coding with a new language is to output **Hello, World!** on the **console**.

1.2 Comments in Python

- **Syntax:**

```
# This is a comment
```

- **Description:**

- Comments start with a **#** and are ignored by the **Python interpreter** and automatically goes to the next line.
- Comments are used to organize our work and allow it to be understood by readers.
- Comments don't output anything

1.3 Outputs

- **Syntax:**

```
print(value)
```

- **Description:**

- The `print()` function is used to output a value to the console.

- **Example usage:**

```
print("Hello")  
print(42)
```

- **Output:**

```
Hello, World!  
42
```

4. Variables and Variable Types

- **Syntax:**

- For `int` (whole number) value:

```
x = 5
```

- For `str` (string, text) value:

```
#We can use both " " or ' ' for str
name = "Alice"
surname = 'Johnson'
```

- For `float` (decimal number) value:

```
#We use a . and not a ,
pi = 3.14
```

- **Description:**

- Variables are used to store data.
- As opposed to C, where variable types must be explicitly declared (e.g., `int`, `float`, `char`), Python does not require type declaration, making it more flexible.

- **Example usage:**

```
x = 42                                # Integer
print(x)

pi = 3.14159                          # Float
print(pi)

name = "Bob"                          # String
print(name)

is_student = False                    # Boolean
print(is_student)
```

- **Output:**

```
42
3.14159
Bob
False
```

- **Important Note:**

- To print the type of a **value/variable** we use the function `type()`:
Syntax:

```
print(type(value))
```

Example Usage:

```
x = 5
#print type of a variable
print(type(x))
#print type of a value
print(type(1.44))
```

Output:

```
<class 'int'>
<class 'float'>
```

5. Operations on Values and Variables

- **Arithmetic Operations:**

- **Addition:** $x + y$

```
result = x + y
print(result)
```

- **Subtraction:** $x - y$

```
result = x - y
print(result)
```

- **Multiplication:** $x * y$

```
result = x * y
print(result)
```

- **Division:** x / y (always returns x float)

```
result = x / y
print(result)
```

- **Euclidean Division:** `x // y` (returns an integer)

```
result = x // y
print(result)
```

- **Modulus (remainder):** `x % y`

```
result = x % y
print(result)
```

- **Power:** `x ** y`

```
result = x ** y
print(result)
```

- **Example Usage:**

```
x = 10
y = 3
print(x + y)  # Addition
print(x - y)  # Subtraction
print(x * y)  # Multiplication
print(x / y)  # Division
print(x // y) # Floor Division
print(x % y)  # Modulus
print(x ** y) # Exponentiation
```

- **Output:**

```
13
7
30
3.3333333333333335
3
1
1000
```

6. Input

- **Syntax:**

```
x = input("Ask user to input something: ")
```

Or alternatively:

```
print("Ask user to input something: ")  
x = input()
```

- **Description:**

- The `input()` function reads user input from the console. It displays a message and waits for the user to type something and press Enter. The input is always stored as a string.

- **Example usage:**

```
name = input("Enter your name: ")  
print("Hello, " + name)
```

- **Output:**

```
Enter your name: Alice  
Hello, Alice
```

- **Important Note:**

- The value read using the `input()` function is always a string. To use it as a different data type, such as an integer, you need to convert it using type casting.
- If you don't convert your value, you won't be able to use the arithmetic operations such as `+`.
- **Example:**

```
x = input("Enter a number: ")  
y = input("Enter another number: ")  
result = x + y  
print(result)  
# or alternatively  
# print(x+y)
```


– **Output:**

```
Enter a number: 5
Enter another number: 3
53
```

- To cast a string to an integer, use the `int()` function.
- To cast a string to a float, use the `float()` function.

– **Example Usage:**

```
x = int(input("Enter a number: "))
y = int(input("Enter another number: "))
# or alternatively
# x = input("Enter a number: ")
# y = input("Enter another number: ")
# x = int(x)
# y = int(y)
result = x + y
print(result)
```

– **Output:**

```
Enter a number: 5
Enter another number: 3
8
```

Chapter 2

Conditions and Loops in Python

1. Conditions

- **Description:**

- Conditions allow your program to make decisions based on whether certain statements are true or false.
- In Python, we use the keywords `if`, `elif`, and `else` to express conditions.

- **Syntax:**

```
if condition1:
    # code to execute when condition is true
elif condition2:
    # code to execute if another_condition is
    true
else:
    # code to execute if no condition is true
```

- **Comparison Operators:**

- `==` checks if two values are equal.
- `!=` checks if two values are not equal.
- `>` checks if the left value is greater than the right.
- `<` checks if the left value is less than the right.
- `>=` checks if the left value is greater than or equal to the right.
- `<=` checks if the left value is less than or equal to the right.

- **Example Usage:**

```
age = int(input("Enter your age: "))

if age >= 18:
    print("You are an adult.")
elif age > 12:
    print("You are a teenager.")
else:
    # age < 12
    print("You are a child.")
```

- **Output:**

```
Enter your age: 15
You are a teenager.
```

- **Logical Operators:**

Logical operators are used to combine multiple conditions in a single expression. The three primary logical operators in Python are:

- **and** - Returns True only if **both conditions** are true.
- **or** - Returns True if **at least one condition** is true.
- **not** - Reverses the result of a condition, returning True if the condition is false.

```
if condition1 and condition2:
    # code to execute when both conditions are
    true
if condition1 or condition2:
    # code to execute if one of the conditions
    is true
if not condition1:
    # code to execute if condition1 is false
```

- **Example Using Logical Operators:**

```
print("Enter the temperature in degrees: ")
temperature = int(input())

if temperature > 30 and temperature < 40:
    print("It's warm outside.")
elif temperature >= 40:
    print("It's very hot outside!")
else:
    print("The temperature is moderate.")
```

- **Output:**

```
Enter the temperature in degrees: 35
It's warm outside.
```