**Functions:**

* a function is a block of reusable code that performs a specific task.
* Python functions can take parameters as inputs and return values as outputs, although both are optional.
* Functions allow you to organize code into logical units, making it easier to read, understand, and maintain. They promote code reuse and modularity, enabling you to use the same piece of code multiple times without repetition.

Here's how you define a function in Python:

**def:**

\* It's a keyword used to define a function.

**function\_name:**

\* It's the name of the function.

\* Choose a descriptive name that reflects what the function does.

\* Follow Python's naming conventions (lowercase with underscores

for readability).

**parameters:**

\* These are optional inputs that the function can accept.

\* Parameters are variables used to pass data into the function.

\* A function can have zero or more parameters.

**Docstring:**

\* It's a string literal that provides documentation about the function.

\* It's enclosed in triple quotes (""" """).

\* It describes what the function does, its parameters, and return values. \* Good documentation helps other developers understand how to use

the function.

**Function body:**

\* It consists of the code that defines what the function does.

\* It can include any valid Python code, such as variable assignments,

conditional statements, loops, and function calls.

**Return statement:**

\* It specifies the value(s) that the function should return to the

caller.

\* The return statement is optional. If omitted, the function returns

None.

\* You can return single values, multiple values (as a tuple), or even

complex data structures.

• Functions are a fundamental concept in Python programming and are widely used across various domains, including data science, software development, and automation.

• They help in writing clean, modular, and maintainable code.

# Here's a example of a function that calculates the square of a number:

def square(num):

"""calculate the square of number"""

return num \* num

square(5)

25

**classes :**

\* a class is a blueprint for creating objects (instances) that share common attributes

and behaviors.

\* Classes provide a way to organize and structure code in a more modular and reusable manner.

\* They are fundamental to object-oriented programming (OOP) in Python and allow you to model

real-world entities and concepts more effectively.

Let's break down the components:

**class:**

\* It's a keyword used to define a class.

**ClassName:**

\* It's the name of the class. Follow Python's naming conventions

(capitalize the first letter of each word in the name).

**Docstring:**

\* It's an optional string literal that provides documentation about the class.

**Class attributes:**

\* These are variables that are shared by all instances of the class.

\* They are defined directly within the class body and are accessed using the class name

**Constructor method (init):**

\* It's a special method used for initializing objects when they are created.

\* It's called automatically when a new instance of the class is created.

\* The self parameter represents the instance itself.

**Instance attributes:**

\* These are variables specific to each instance of the class.

\* They are initialized within the \_\_init\_\_ method using the self keyword.

**Instance methods:**

\* These are functions defined inside a class that operate on its instances.

\* They take self as the first parameter, which refers to the instance itself.