PYTHON O LEVEL

FUNCTIONS

AKSHAR
COMPUTER
CENTRE

CONTACT US:

DOEACCAKSHARCOMPUTERCENTRE.COM

Functions

- Functions are blocks of organized, reusable code that perform a specific task.
- They allow you to break down a problem into smaller tasks, making your code more modular and easier to manage.
- Functions can take inputs, process them, and return outputs.
- They follow a top-down approach of problem-solving, where you solve the main problem by breaking it down into smaller sub-problems.
- Functions promote modular programming, where you write small, independent modules that can be combined to solve complex problems.

```python
# Example of a simple function
def greet(name):
    return f"Hello, {name}!"

# Calling the function
print(greet("Alice"))  # Output: Hello, Alice!
```

Function Parameter

- Parameters are variables that are defined in the function definition and are used to pass values into a function.
- They allow functions to accept inputs and work with them.

```python
# Example of function with parameters
def add(a, b):
    return a + b

# Calling the function with arguments
result = add(3, 5)
print(result)  # Output: 8
```

Functions

Local Variables

- Variables defined inside a function are called local variables.
- They are only accessible within the function where they are defined.

```python
# Example of local variables
def my_func():
    x = 10  # Local variable
    print(x)

my_func()  # Output: 10
# print(x) will result in an error because x is a local variable and is not
accessible outside the function.
```

Return Statement

- The return statement is used to exit a function and return a value back to the caller.
- It can also be used to return multiple values.

```python
# Example of return statement
def square(x):
    return x * x

result = square(4)
print(result)  # Output: 16
```

Functions

Default argument values

- Default argument values are specified in the function definition.
- If the caller does not provide a value for that argument, the default value is used.

```python
# Example of default argument values
def greet(name="World"):
    return f"Hello, {name}!"

print(greet())  # Output: Hello, World!
print(greet("Alice"))  # Output: Hello, Alice!
```

Keyword arguments

- Keyword arguments are passed to a function with the syntax keyword=value.
- They allow you to specify arguments in any order by explicitly naming the parameter.

```python
# Example of keyword arguments
def greet(greeting, name):
    return f"{greeting}, {name}!"

print(greet(name="Alice", greeting="Hi"))  # Output: Hi, Alice!
```

Functions

VarArgs Parameters

- VarArgs parameters allow you to pass a variable number of arguments to a function.
- They are denoted by *args and can be used when the number of arguments that a function needs to accept is unknown.

```
# Example of VarArgs parameters
def sum_values(*args):
    total = 0
    for num in args:
        total += num
    return total

print(sum_values(1, 2, 3, 4))  # Output: 10
```

Now, let's put this all together with a layman example:

Imagine you're baking a cake. Each step of the recipe can be likened to a function. Mixing the ingredients could be one function, baking the cake could be another. You pass ingredients (parameters) to these functions and they perform specific tasks. If you need to adjust the recipe, you can change just one part (function) without rewriting the entire recipe. So, functions help you break down a complex task (like baking a cake) into smaller, manageable steps, making your baking process more efficient and flexible.

Library Functions

A library function, in the context of programming, refers to a pre-defined function that is provided by a library or module. Libraries in programming contain collections of functions and routines that can be used to perform specific tasks without having to write the code for those tasks from scratch. These functions are designed to be reusable and can be called upon in different parts of a program to perform their specific tasks.

There are three main library functions: input () , eval () and print () .

input() :

The input() function is a built-in library function in Python that allows the user to enter input from the keyboard. It prompts the user with a message (if provided) and waits for the user to type something. Once the user presses Enter, it returns the entered value as a string.

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

eval() :

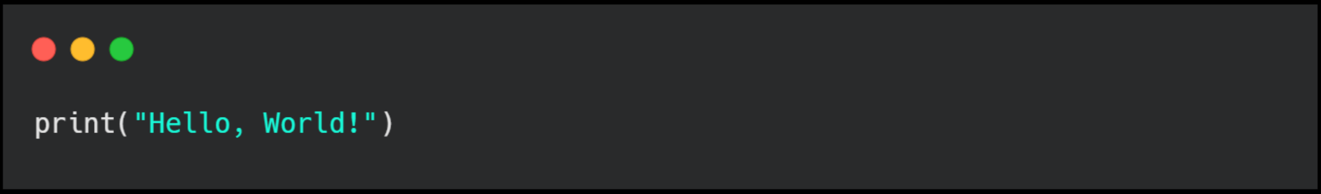
The eval() function is a built-in library function in Python that evaluates a string as a Python expression. It takes a string as input and executes it as a Python expression. This function is powerful but should be used with caution as it can execute arbitrary code.

```python
result = eval("2 + 3")
print(result)  # Output: 5
```

Library Functions

print() :

The print() function is a built-in library function in Python used to display output to the console. It takes one or more arguments and prints them to the standard output (usually the console).



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