1. In Python, what is the difference between a built-in function and a user-defined function? Provide an example of each.

Built-in Functions : Built-in functions are functions that are provided by the Python programming language itself. They are part of the Python standard library and can be used directly without requiring any additional code to be written.

In [3]:
```python
# Eg:In the below line 'print' is the built-in function.
print("Hello, world!")
```

```
Hello, world!
```

User-defined Function : User-defined functions are functions that are created by the programmer to perform specific tasks. User-defined functions are defined using the def keyword, and they can take parameters and return values.

In [4]:
```python
# Eg: below given function it user dedined function for factorial
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)

result = factorial(5)
print(result)
```

```
120
```

2. How can you pass arguments to a function in Python? Explain the difference between positional arguments and keyword arguments.

Positional Arguments: Positional arguments are the most basic way to pass arguments to a function. They are provided in the order that the function's parameters are defined.

Keyword Arguments: Keyword arguments are specified using parameter names followed by an equal sign and the corresponding value. This allows you to provide arguments out of order and explicitly associate values with specific parameters.

3. What is the purpose of the return statement in a function? Can a function have multiple return statements? Explain with an example.

The return statement in a function is used to specify the value that the function should produce as its result. It allows a function to compute a value and send it back to the caller. A function can have multiple return statements, but only one of them will be executed during the function's execution

In [7]:
```python
def divide(a, b):
    if b == 0:
        return "Cannot divide by zero!"  # First return statement
    else:
        result = a / b
        return result  # Second return statement

# Function calls
result1 = divide(10, 2)
print(result1)

result2 = divide(10, 0)
print(result2)
```

```
5.0
Cannot divide by zero!
```

4. What are lambda functions in Python? How are they different from regular functions? Provide an example where a lambda function can be useful.

Lambda functions in Python, also known as anonymous functions, are small, inline functions that are defined without a name. Lambda functions are particularly useful when you need a quick function for a short period of time and don't want to define a full named function using the def keyword.

In [9]:
```python
# Eg:
square = lambda x: x ** 2
squared_value = square(5)
print(squared_value)
```

```
25
```

5. How does the concept of "scope" apply to functions in Python? Explain the difference between local scope and global scope.

Scope in Python refers to the visibility and accessibility of variables within different parts of your code. It defines where a variable can be accessed and modified.

Local Scope: Variables defined within a function are said to have a local scope. These variables are accessible only within the function they are defined in.

Global Scope: Variables defined outside of any function, at the top level of a script or module, have a global scope. These variables can be accessed and modified from anywhere in the code, including inside functions.

6. How can you use the "return" statement in a Python function to return multiple values?

In Python, you can use the return statement in a function to return multiple values by returning them as a tuple, a list, or any other iterable data structure.

In [14]:
```python
# Eg:
def get_coordinates():
    x = 5
    y = 10
    return x, y

result = get_coordinates()
print(result)
```

```
(5, 10)
```

7. What is the difference between the "pass by value" and "pass by reference" concepts when it comes to function arguments in Python?

Pass by Value: In a "pass by value" scenario, when a variable is passed as an argument to a function, the function receives a copy of the variable's value. Any changes made to the parameter inside the function do not affect the original variable outside the function.

Pass by Reference: In a "pass by reference" scenario, when a variable is passed as an argument to a function, the function receives a reference to the original variable. Any changes made to the parameter inside the function directly affect the original variable outside the function.

8. Create a function that can intake integer or decimal value and do following operations: a. Logarithmic function (log x) b. Exponential function (exp(x)) c. Power function with base 2 (2^x) d. Square root

In [17]:

```python
import math

def perform_operations(x):
    try:
        x = float(x)  # Convert the input to a float

        log_result = math.log(x)
        exp_result = math.exp(x)
        power_result = 2 ** x
        sqrt_result = math.sqrt(x)

        return {
            "Logarithmic": log_result,
            "Exponential": exp_result,
            "Power (2^x)": power_result,
            "Square Root": sqrt_result
        }
    except ValueError:
        return "Invalid input. Please enter a valid number."

# Test the function
input_value = input("Enter a number: ")
results = perform_operations(input_value)

if isinstance(results, str):
    print(results)
else:
    for operation, result in results.items():
        print(f"{operation}: {result:.2f}")
```

```
Enter a number: 4.8
Logarithmic: 1.57
Exponential: 121.51
Power (2^x): 27.86
Square Root: 2.19
```

9. Create a function that takes a full name as an argument and returns first name and last name.

In [22]:
```python
def split_full_name(full_name):
    names = full_name.split()
    if len(names) >= 2:
        first_name = names[0]
        last_name = ' '.join(names[1:])
        return first_name, last_name
    else:
        return "Invalid full name. Please provide only first name and a la

full_name = input("Enter your full name: ")
first, last = split_full_name(full_name)
print("First Name:", first)
print("Last Name:", last)
```

```
Enter your full name: Hritik Kadam
First Name: Hritik
Last Name: Kadam
```