



近屿智能 JYI  
Just the Yarest Intelligence

# Python 函数基础

主讲人:

日期:



# 目录

CONTENTS

01

函数



02

返回值



03

参数



04

函数嵌套





近屿智能 JYI  
Just the Yarest Intelligence

01

函数



# 定义



函数是执行特定任务的代码块，使用def关键字定义。



函数定义包含函数名、参数列表，以冒号结束，后跟函数体。



注意：  
函数名要符合标识符的规定，最好见名知道该函数的功能。



# 作用



01

提高代码复用性，避免代码重复，简化程序结构。

03

通过封装功能，减少代码间的耦合，提高代码质量。

02

实现模块化编程，使程序更加清晰，便于维护和扩展。





# 结构



结构:

```
def 函数名():
```

函数体

示例:

```
def hello():
```

```
    """文档字符串"""
```

```
    print ("你好")
```

```
hello ()
```

# 注意点



- 函数名应符合Python标识符规则，简洁明了，体现函数功能。
- 调用函数前必须先定义，确保函数在调用时已存在于当前作用域。
- 函数定义时可包含文档字符串，描述函数功能，便于他人理解。



近屿智能 JYI  
Just the Yarest Intelligence

02

返回值





# 定义

返回值可以用于后续计算或作为其他函数的输入，实现数据传递。

函数通过return语句返回结果给调用者，可以是单个值或多个值。



# return 和 print 的区别



01

return表示函数结束，不继续执行后续代码；print只负责输出信息。

02

return是返回计算值，print是打印结果，两者在函数中的作用不同。

# 示例：

```
def example_with_return():  
    print("这条信息会在 return 之前打印")  
    return "这是返回值"  
    print("这条信息不会被打印，因为它在 return 之后")  
  
# 调用函数并打印返回值  
result = example_with_return()  
print(f"函数返回的结果是: {result}")
```

# 返回值的总结



## 01 没有显式 return 或 return 后没有表达式

如果没有显式return或return后没有表达式，函数返回None。

## 02 单个返回值

如果return后跟一个值，函数直接返回该值给调用者。

## 03 多个返回值

如果return后跟多个值，以元组形式返回给调用者。



# 返回值的总结



## 示例:

```
def no_return():  
    pass # 返回 None
```

```
def single_return():  
    return "single"
```

```
def multiple_returns():  
    return 1, "two", 3.0 # 返回 (1, 'two', 3.0)
```

```
print(no_return())      # 输出: None  
print(single_return())  # 输出: single  
print(multiple_returns()) # 输出: (1, 'two', 3.0)
```



近屿智能 JYI  
Just the Yarest Intelligence

03

参数



# 形参 & 实参



## 实参

实参是调用函数时传入的具体值，与形参一一对应，传递给函数。

## 形参

形参是定义函数时小括号内的变量名称，用于接收调用时传入的实参。



## 示例:

```
def add(a, b): # a 和 b 是形参  
    """返回两个数相加的结果"""  
    return a + b
```

```
# 调用函数, 5 和 3 是实参  
result = add(5, 3)  
print(f"结果是: {result}") # 输出: 结果是: 8
```



# 必备参数 (位置参数)

必备参数必须按照顺序提供，与定义时一致，数量和类型需匹配。

...



## 示例:

```
def greet(first_name, last_name):  
    """打印问候语"""  
    print(f"Hello, {first_name} {last_name}!")
```

# 按照顺序提供实参, 数量和类型匹配

```
greet("Alice", "Smith") # 输出: Hello, Alice Smith!
```

# 默认参数

为参数提供默认值，允许调用时不提供这些参数的值，提高函数的灵活性。





## 示例:

```
def greet(name="World"):
    """打印问候语, 默认向世界问好"""
    print(f"Hello, {name}!")
```

# 不提供实参时使用默认值

`greet()` # 输出: Hello, World!

# 提供实参覆盖默认值

`greet("Alice")` # 输出: Hello, Alice!



# 可变参数



01

允许传递不定数量的参数，接收的是一个元组，增强函数的适应性。



## 示例:

```
def print_args(*args):  
    """打印所有传入的参数"""  
    print("接收到的参数列表:")  
    for arg in args:  
        print(arg)
```

# 调用函数时可以传递不同数量的实参

```
print_args(1, 2, 3) # 输出:
```

```
    # 接收到的参数列表:
```

```
    # 1
```

```
    # 2
```

```
    # 3
```

```
print_args('apple', 'banana', 'cherry') # 输出:
```

```
    # 接收到的参数列表:
```

```
    # apple
```

```
    # banana
```

```
    # cherry
```

# 关键字参数



接收任意数量的关键字参数作为字典，  
允许用户在调用函数时指定参数名称及  
其对应的值。

## 示例:

```
def describe_person(name, age, city):
```

```
    """描述一个人的基本信息"""
```

```
    print(f"Name: {name}")
```

```
    print(f"Age: {age}")
```

```
    print(f"City: {city}")
```

```
# 使用关键字参数调用函数
```

```
describe_person(name="Alice", age=25, city="Beijing")
```





近屿智能 JYI  
Just the Yarest Intelligence

04

# 函数嵌套



# 嵌套定义



01

在一个函数内部定义另一个函数，  
称为内部函数或局部函数，增强代  
码的模块化。

# 嵌套调用



在一个函数中调用另一个已经定义好的函数，实现代码复用和逻辑封装。



## 注意事项

内部函数只能在其外部函数中被调用，遵循编码规范，确保正确的缩进和空行使用。



## 示例:

```
def outer_function():  
    def inner_function():  
        print("这是内部函数")
```

```
    inner_function()
```

```
outer_function() # 输出: 这是内部函数
```

```
def study():  
    print("晚上在学习")
```

```
def course():  
    study()  
    print("Python基础")
```

```
course() # 输出:  
# 晚上在学习  
# Python基础
```



# 练习：编写一个打招呼的函数并调用它

01  
定义一个名为greet的函数，接收一个参数name，打印问候语。

02  
如果没有提供参数，则默认向“World”问好，体现默认参数的使用。

03  
通过调用greet()和greet("Alice")，展示默认参数和指定参数的调用方式。

# 练习

## 题目描述

假设你正在为一家图书馆开发一个简单的图书管理系统。你需要编写几个函数来帮助管理员管理书籍信息。具体要求如下：

添加书籍：编写一个名为 `add_book` 的函数，该函数接收书名（`title`）、作者（`author`）和数量（`quantity`）作为关键字参数，并将这些信息存储在一个字典中。如果数量未提供，默认设置为1。

显示所有书籍：编写一个名为 `display_books` 的函数，该函数没有参数，用于打印当前所有书籍的信息。

借阅书籍：编写一个名为 `borrow_book` 的函数，该函数接收书名（`title`）作为默认参数，并减少对应书籍的数量。如果书籍不存在或库存不足，则打印相应提示信息。

归还书籍：编写一个名为 `return_book` 的函数，该函数接收书名（`title`）作为位置参数，并增加对应书籍的数量。如果书籍不存在，则添加该书籍并设置数量为1。



YOUR LOGO



近屿智能 JYI  
Just the Yarest Intelligence

谢谢大家

2024

