Python函数

目录

Contents

学习目标

Learning Objectives

1. 快速体验函数的使用
2. 了解函数的作用

函数

函数：是组织好的，可重复使用的，用来实现特定功能的代码段。

```
name = "itheima"
length = len(name)
print(length)
```

```
test (1)
D:\dev\python\python3
7
```

为什么随时都可以使用len()统计长度？

因为，len()是Python内置的函数：

- 是提前写好的
- 可以重复使用
- 实现统计长度这一特定功能的代码段

我们使用过的：input()、print()、str()、int()等都是Python的内置函数

函数的快速体验

接下来，让我们实际的体验一下函数的使用。

让我们在PyCharm中完成一个案例需求：

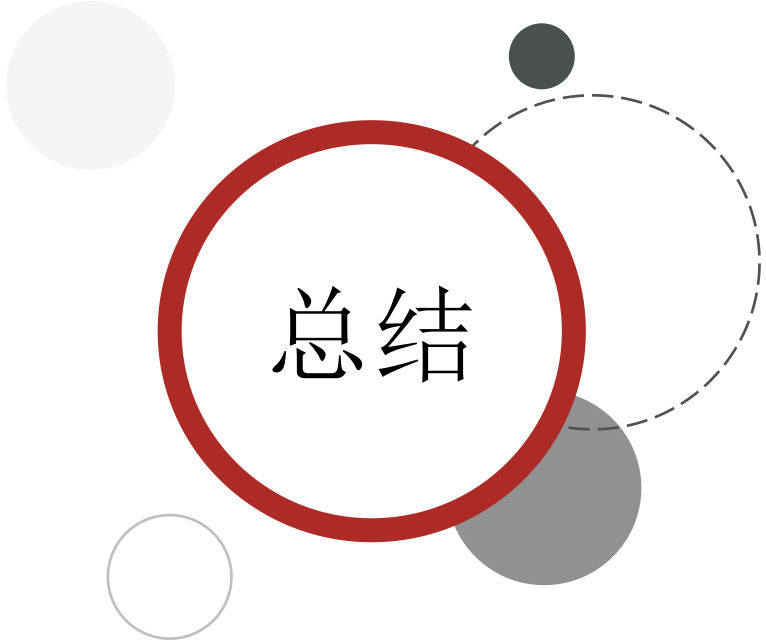
不使用内置函数len()，完成字符串长度的计算。

> 体验代码，会出现未学习到的语法，同学们只需要关心效果即可，语法后面会详细讲解。

为什么要学习、使用函数呢？

为了得到一个针对特定需求、可供重复利用的代码段

提高程序的复用性，减少重复性代码，提高开发效率



1. 函数是：

组织好的、可重复使用的、用来实现特定功能的代码段

2. 使用函数的好处是：

- 将功能封装在函数内，可供随时随地重复利用
- 提高代码的复用性，减少重复代码，提高开发效率

目录

Contents

学习目标

Learning Objectives

1. 掌握函数的基础定义语法

函数的定义

函数的定义：

```
def 函数名(传入参数):
    函数体
    return 返回值
```

函数的调用：

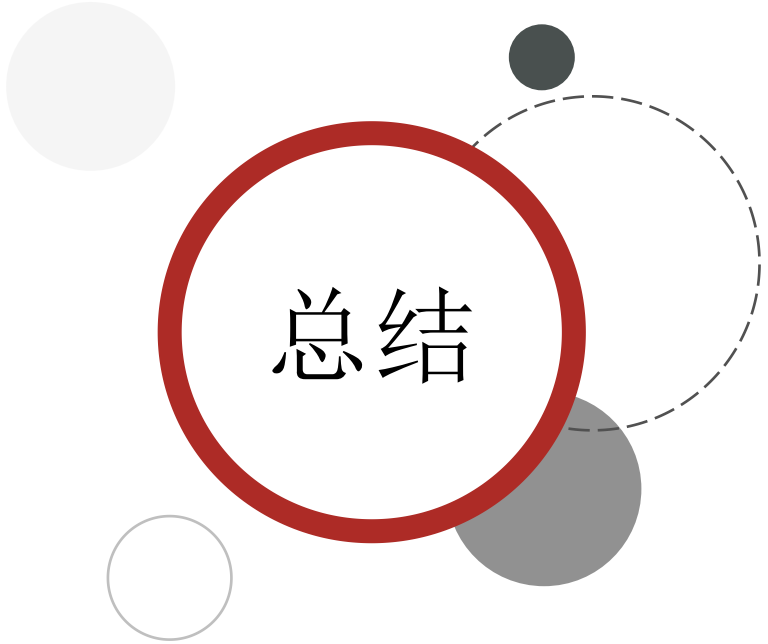
```
函数名(参数)
```

注意事项

① 参数如不需要，可以省略（后续章节讲解）

② 返回值如不需要，可以省略（后续章节讲解）

③ 函数必须先定义后使用



1. 函数的定义语法

```
def 函数名(传入参数):
    函数体
    return 返回值
```

2. 函数使用步骤：
   - 先定义函数
   - 后调用函数

3. 注意事项：
   - 参数不需要，可以省略
   - 返回值不需要，可以省略

练习

练习案例：自动查核酸

定义一个函数，函数名任意，要求调用函数后可以输出如下欢迎语：

```
欢迎来到黑马程序员！
请出示您的健康码以及72小时核酸证明！
```

目录

Contents

学习目标

Learning Objectives

1. 掌握函数传入参数的使用

函数的传入参数

传入参数的功能是：在函数进行计算的时候，接受外部（调用时）提供的数据

有如下代码，完成了2个数字相加的功能：

```
def add():
    result = 1 + 2
    print(f"1 + 2的结果是:{result}")
```

函数的功能非常局限，只能计算1 + 2。

有没有可能实现：每一次使用函数，去计算用户指定的2个数字，而非每次都是1 + 2呢？

可以的，使用函数的传入参数功能，即可实现。

函数的传入参数 - 传参定义

基于函数的定义语法：

```
def 函数名(传入参数):
    函数体
    return 返回值
```

可以有如下函数定义：

```
def add(x, y):
    result = x + y
    print(f"{x} + {y}的结果是:{result}")
```

实现了，每次计算的是x + y，而非固定的1 + 2

x + y的值，可以在调用函数的时候指定。

函数的传入参数 - 语法解析

语法解析：

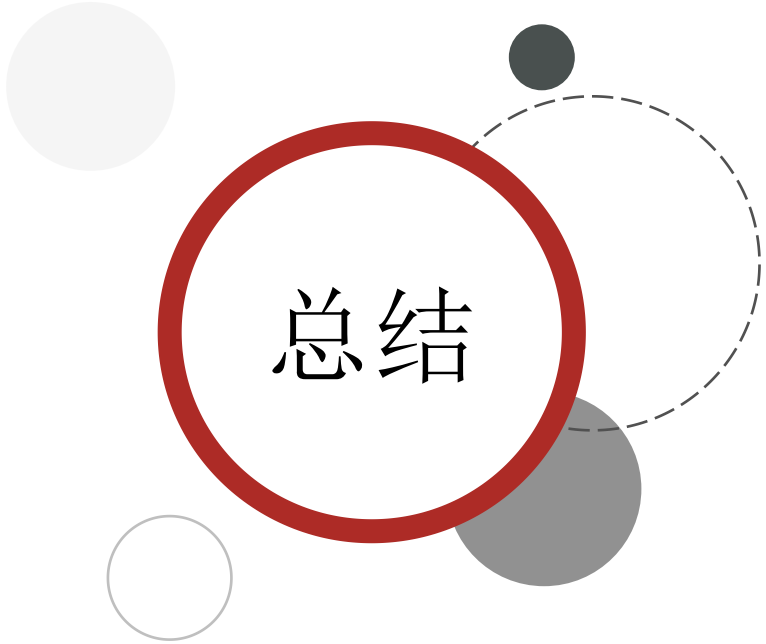
```
# 定义函数
def add(x, y):
    result = x + y
    print(f"{x} + {y}的结果是:{result}")

# 调用函数
add(5, 6)
```

- 函数定义中，提供的x和y，称之为：形式参数（形参），表示函数声明将要使用2个参数
  - **参数之间使用逗号进行分隔**
- 函数调用中，提供的5和6，称之为：实际参数（实参），表示函数执行时真正使用的参数值
  - **传入的时候，按照顺序传入数据，使用逗号分隔**

函数的传入参数

传入参数的数量是不受限制的。

- 可以不使用参数
- 也可以仅使用任意N个参数



1. 函数的传入参数的作用是？

在函数运行的时候，接受外部传入的数据

2. 使用方式

```
def add(x, y):
    result = x + y
    print(f"{x} + {y}的结果是:{result}")
```

3. 注意事项

- 函数定义中的参数，称之为形式参数
- 函数调用中的参数，称之为实际参数
- 函数的参数数量不限，使用逗号分隔开
- 传入参数的时候，要和形式参数一一对应，逗号隔开

练习

练习案例：升级版自动查核酸

定义一个函数，名称任意，并接受一个参数传入（数字类型，表示体温）

在函数内进行体温判断（正常范围：小于等于37.5度），并输出如下内容：

```
欢迎来到黑马程序员！请出示您的健康码以及72小时核酸证明，并配合测量体温！
体温测量中，您的体温是：37.3度，体温正常请进！
```

```
欢迎来到黑马程序员！请出示您的健康码以及72小时核酸证明，并配合测量体温！
体温测量中，您的体温是：39.3度，需要隔离！
```

目录

Contents

目录

Contents

函数的返回值

- 函数返回值的定义
- None类型

学习目标

Learning Objectives

1. 掌握函数返回值的作用
2. 掌握函数返回值的定义语法

什么是返回值

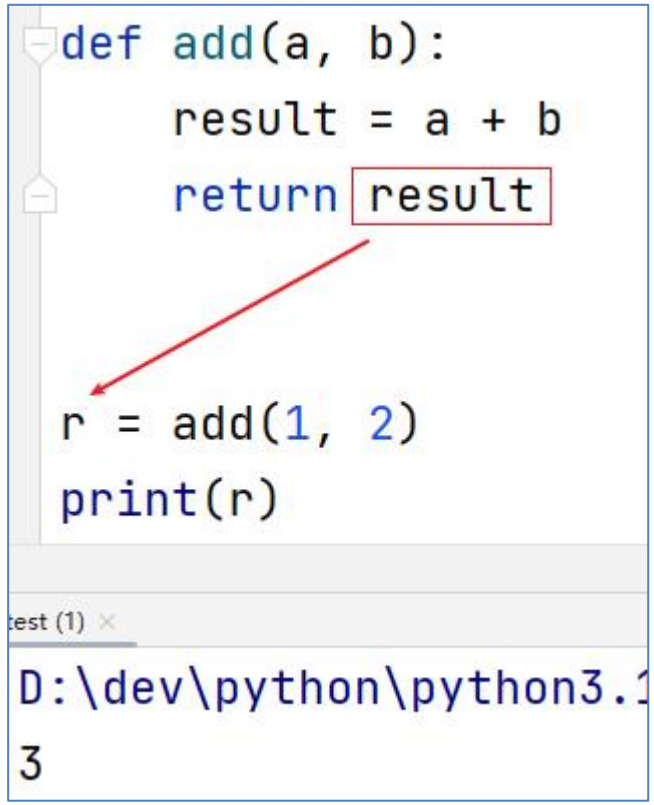
生活中的返回值：

参数

返回值

什么是返回值

程序中的返回值：



如图代码

定义两数相加的函数功能。完成功能后，会将相加的结果返回给函数调用者

所以，变量r接收到了函数的执行结果。

综上所述：

所谓“返回值”，就是程序中函数完成事情后，最后给调用者的结果

返回值的语法

语法格式如图：

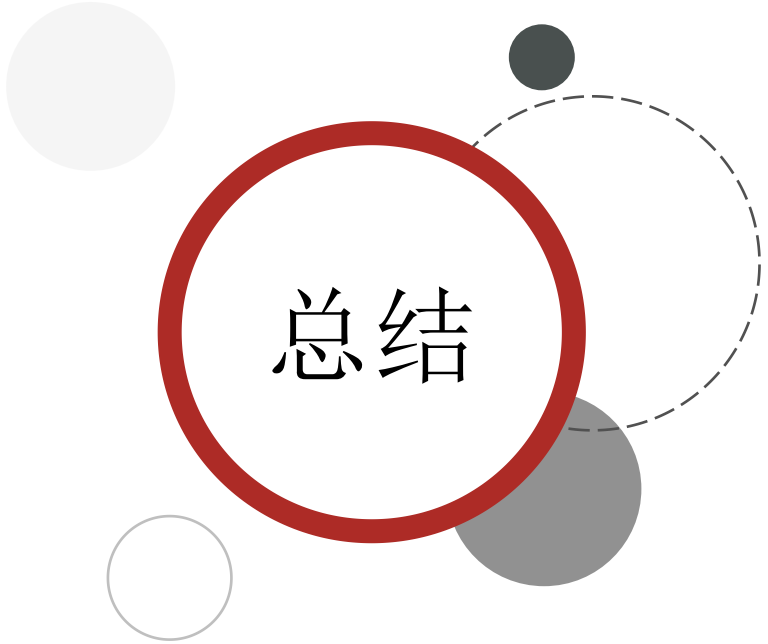
```
def 函数(参数...):
    函数体
    return 返回值


变量 = 函数(参数)
```

如图，变量就能接收到函数的返回值

语法就是：通过return关键字，就能向调用者返回数据

去实践一下吧：

定义一个函数，完成两数相加的功能，并返回结果



1. 什么是函数返回值？

函数在执行完成后，返回给调用者的结果

2. 返回值的应用语法：

使用关键字：return 来返回结果

```
def 函数(参数...):
    函数体
    return 返回值


变量 = 函数(参数)
```

3. 注意：

函数体在遇到return后就结束了，所以写在return后的代码不会执行。

目录

Contents

函数的返回值

- 函数返回值的定义
- None类型

None类型

思考：如果函数没有使用return语句返回数据，那么函数有返回值吗？

实际上是：有的。

Python中有一个特殊的字面量：None，其类型是：<class 'NoneType'>

无返回值的函数，实际上就是返回了：None这个字面量

None表示：空的、无实际意义的意思

函数返回的None，就表示，这个函数没有返回什么有意义的内容。

也就是返回了空的意思。

None类型

演示：

```
def say_hello():
    print("Hello...")

# 使用变量接收say_hello函数的返回值
result = say_hello()
# 打印返回值
print(result)           # 结果None
# 打印返回值类型
print(type(result))     # 结果<class 'NoneType'>
```

None可以主动使用return返回，效果等同于不写return语句：

```
def say_hello():
    print("Hello...")
    return None

# 使用变量接收say_hello函数的返回值
result = say_hello()
# 打印返回值
print(result)           # 结果None
```

None类型的应用场景

None作为一个特殊的字面量，用于表示：空、无意义，其有非常多的应用场景。

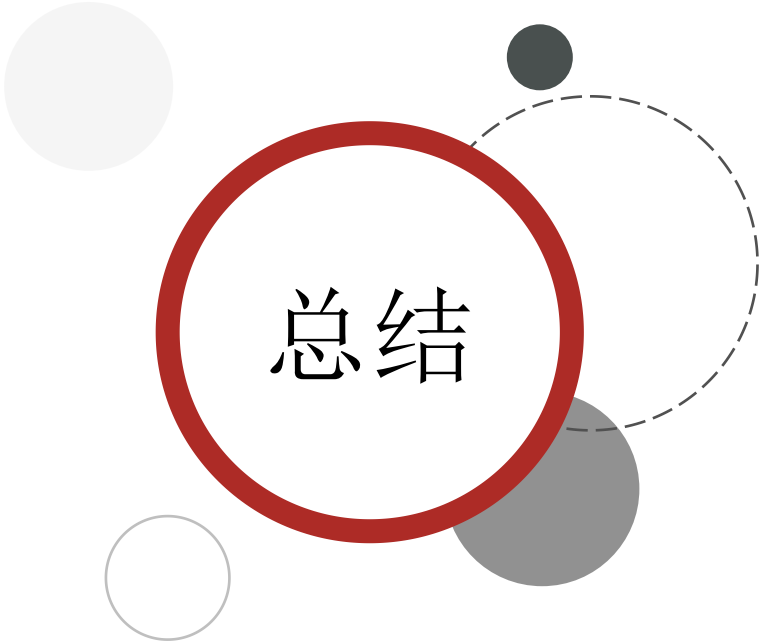
- 用在函数无返回值上

- 用在if判断上
  - **在if判断中，None等同于False**
  - **一般用于在函数中主动返回None，配合if判断做相关处理**

```
def check_age(age):
    if age > 18:
        return "SUCCESS"
    return None

result = check_age(5)
if not result:
    print("未成年，不可进入")
```

- 用于声明无内容的变量上
  - **定义变量，但暂时不需要变量有具体值，可以用None来代替**

```
# 暂不赋予变量具体值
name = None
```



1. 什么是None

None是类型’NoneType’的字面量，用于表示：空的、无意义的

2. 函数如何返回None

- 不使用return语句即返回None
- 主动return None

3. 使用场景

- 函数返回值
- if判断
- 变量定义

目录

Contents

学习目标

Learning Objectives

1. 掌握通过注释对函数进行解释说明

函数的说明文档

函数是纯代码语言，想要理解其含义，就需要一行行的去阅读理解代码，效率比较低。

我们可以给函数添加说明文档，辅助理解函数的作用。

语法如下：

```
def func(x, y):
    """
    函数说明
    :param x: 形参x的说明
    :param y: 形参y的说明
    :return: 返回值的说明
    """
    函数体
    return 返回值
```

通过多行注释的形式，对函数进行说明解释

- 内容应写在函数体之前

在PyCharm中查看函数说明文档

在PyCharm编写代码时，可以通过鼠标悬停，查看调用函数的说明文档

```
def add(x, y):
    """
    两数相加
    :param x: 相加的数字1
    :param y: 相加的数字2
    :return: 返回相加的结果
    """
    return x + y

add(1, 2)
```

鼠标悬停

PEP 8: E305 expected 2 blank lines after class or function definition, found 1

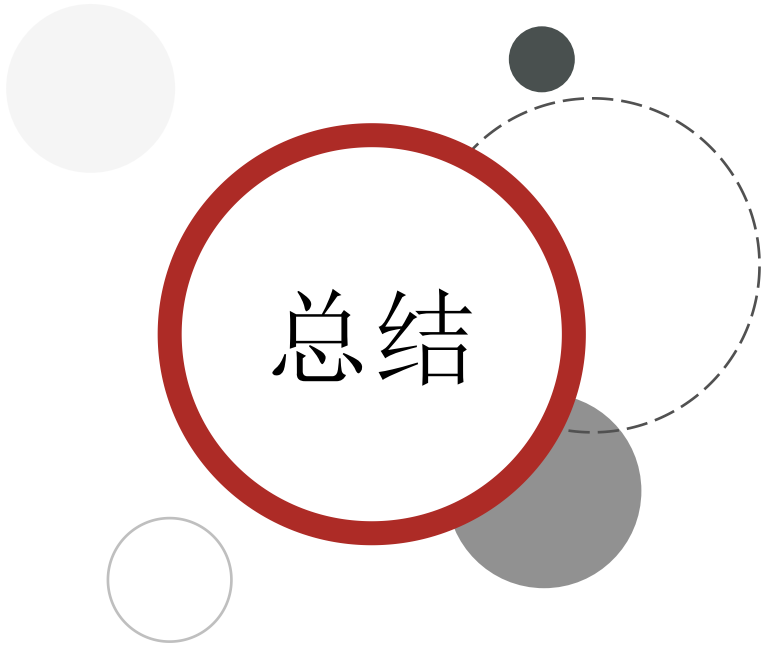
Reformat the file Alt+Shift+Enter More actions... Alt+Enter

```
D:/python-learn/06_函数/test.py
def add(x: {__add__},
        y: Any) -> Any
```

两数相加 :param x: 相加的数字1 :param y: 相加的数字2 :return: 返回相加的结果

Params: x – 相加的数字1
y – 相加的数字2

Returns: 返回相加的结果



1. 函数说明文档的作用是？

对函数进行说明解释，帮助更好理解函数的功能

2. 定义语法

```
def func(x, y):
    """
    函数说明
    :param x: 参数x的说明
    :param x: 参数y的说明
    :return:  返回值的说明
    """
    函数体
    return 返回值
```

- :param 用于解释参数
- :return 用于解释返回值

目录

Contents

学习目标

Learning Objectives

1. 掌握函数的嵌套调用

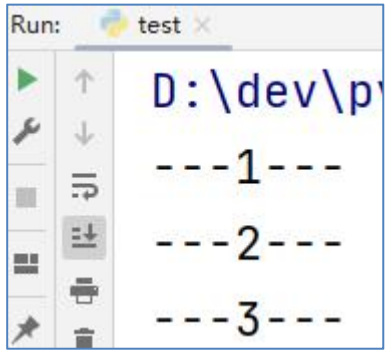
2. 理解嵌套调用的执行流程

什么是函数的嵌套

所谓函数嵌套调用指的是一个函数里面又调用了另外一个函数

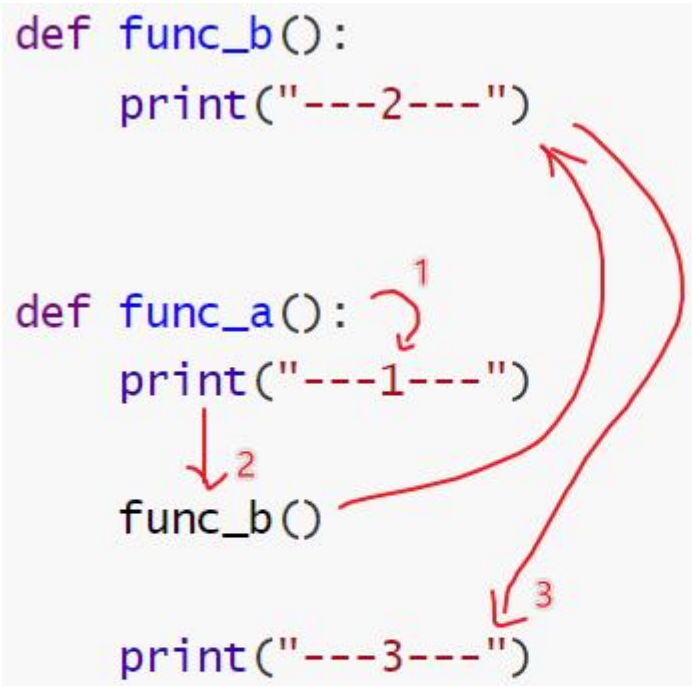
```
def func_b():
    print("---2---")


def func_a():
    print("---1---")

    func_b()

    print("---3---")


# 调用函数func_a
func_a()
```

执行效果：



执行过程

```
def func_b():
    print("---2---")


def func_a():
    print("---1---")
    func_b()

    print("---3---")


# 调用函数func_a
func_a()
```



如果函数A中，调用了另外一个函数B，那么先把函数B中的任务都执行完毕之后才会回到上次 函数A执行的位置

总结

1. 什么是嵌套调用

在一个函数中，调用另外一个函数

2. 执行流程

函数A中执行到调用函数B的语句，会将函数B全部执行

完成后，继续执行函数A的剩余内容

目录

Contents

学习目标

Learning Objectives

1. 知道什么是局部变量
2. 知道什么是全局变量

局部变量

变量作用域指的是变量的作用范围（变量在哪里可用，在哪里不可用）

主要分为两类：局部变量和全局变量

所谓局部变量是定义在函数体内部的变量，即只在函数体内部生效

```
1 def testA():
2       num = 100
3       print(num)
4
5 testA()    # 100
6 print(num)    # 报错：name 'num' is not defined
```

变量a是定义在`testA`函数内部的变量，在函数外部访问则立即报错.

局部变量的作用：在函数体内部，临时保存数据，即当函数调用完成后，则销毁局部变量

全局变量

所谓全局变量，指的是在函数体内、外都能生效的变量

思考：如果有一个数据，在函数A和函数B中都要使用，该怎么办？

答：将这个数据存储在一个全局变量里面

```
# 定义全局变量a
num = 100

def testA():
    print(num)    # 访问全局变量num，并打印变量num存储的数据


def testB():
    print(num)    # 访问全局变量num，并打印变量num存储的数据


testA()    # 100
testB()    # 100
```

global关键字

思考：`testB`函数需要修改变量num的值为200，如何修改程序？

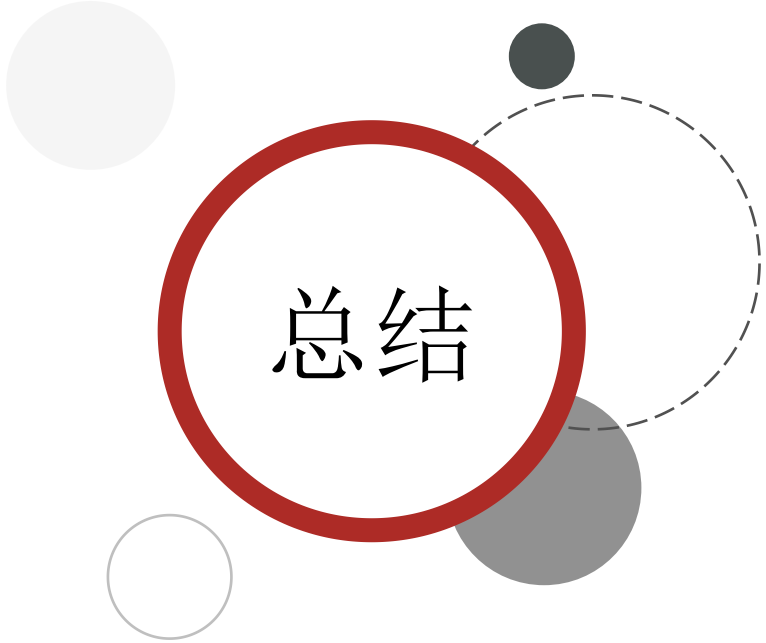
```
num = 100

def testA():
      print(num)


def testB():
      num = 200
      print(num)


testA()                          # 结果：100
testB()                          # 结果：200
print(f'全局变量num = {num}')     # 结果：全局变量num = 100
```

`testB`函数内部的 num = 200 是定义了一个局部变量

global关键字

☆ 使用 global关键字 可以在函数内部声明变量为全局变量，如下所示

```
num = 100

def testA():
      print(num)


def testB():
      # global 关键字声明a是全局变量
      global num
      num = 200
      print(num)


testA()                          # 结果：100
testB()                          # 结果：200
print(f'全局变量num = {num}')     # 结果：全局变量num = 200
```



1. 什么是局部变量

作用范围在函数内部，在函数外部无法使用

2. 什么是全局变量

在函数内部和外部均可使用

3. 如何将函数内定义的变量声明为全局变量

- 使用global关键字，global 变量

目录

Contents

练习

综合案例：黑马ATM

- 主菜单效果

```
----------------主菜单-----------------
周杰轮，您好，欢迎来到黑马银行ATM。请选择操作：
查询余额  [输入1]
存款      [输入2]
取款      [输入3]
退出      [输入4]
请输入您的选择：
```

- 查询余额效果

```
------------------查询余额--------------------
周杰轮，您好，您的余额剩余：5000000元
```

- 存、取款效果

```
----------------存款-----------------
周杰轮，您好，您存款50000元成功
周杰轮，您好，您的余额剩余：5050000元
```

```
----------------取款-----------------
周杰轮，您好，您取款50000元成功
周杰轮，您好，您的余额剩余：4950000元
```

练习

综合案例：黑马ATM

- 定义一个全局变量：money，用来记录银行卡余额（默认5000000）
- 定义一个全局变量：name，用来记录客户姓名（启动程序时输入）
- 定义如下的函数：
  - **查询余额函数**
  - **存款函数**
  - **取款函数**
  - **主菜单函数**
- 要求：
- 程序启动后要求输入客户姓名
- 查询余额、存款、取款后都会返回主菜单
- 存款、取款后，都应显示一下当前余额
- 客户选择退出或输入错误，程序会退出，否则一直运行

传智教育旗下高端IT教育品牌