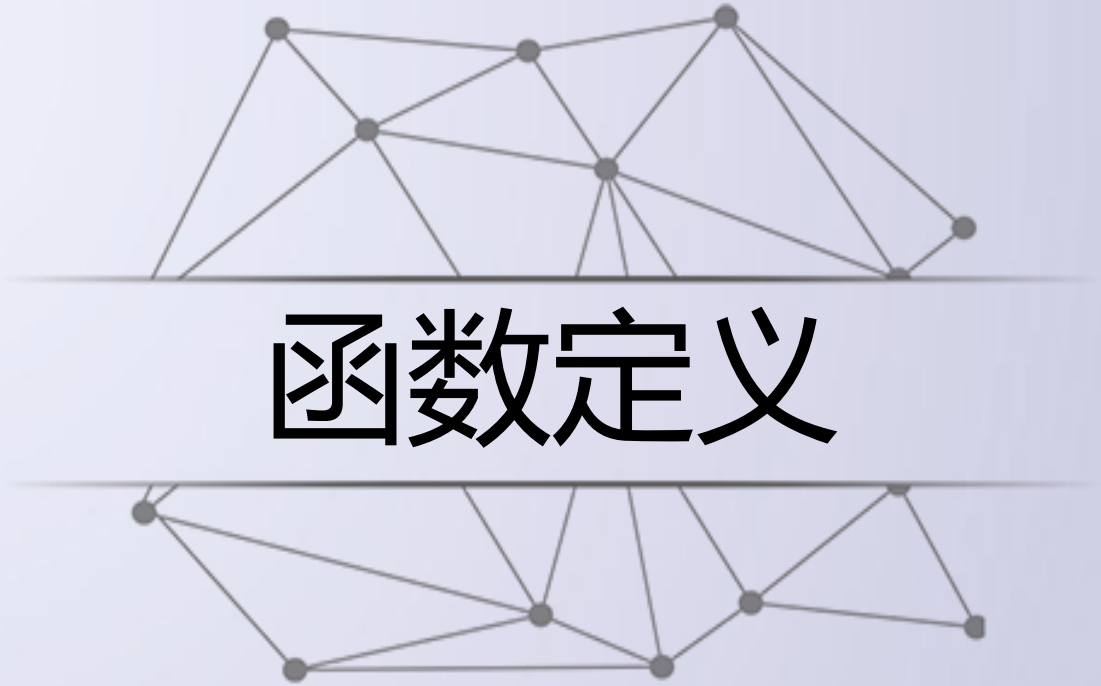

函数定义

礼 欣

北京理工大学

本章重点

- 理解函数在程序设计中的作用
- 理解Python中函数调用和参数传递的过程
- 掌握使用函数减少代码重复性、增加程序模块化的方法
- 理解递归的概念并掌握递归的使用

python 语言程序设计

6.1 函数定义

- 函数：完成特定功能的一个语句组，通过调用函数名来完成语句组的功能。
- 为函数提供不同的参数，可以实现对不同数据的处理
- 函数可以反馈结果

python 语言程序设计

6.1 函数定义

- 分类：
  - 自定义函数：用户自己编写的
  - 系统自带函数： Python内嵌的函数（如abs()、eval()）、Python标准库中的函数（如math库中的sqrt()）、图形库中的方法（如myPoint.getX()）等

6.1 函数定义

- 使用函数目的：
  - 降低编程的难度
  - 代码重用
- 函数定义：使用def语句

```
def <name>(<parameters>):
    <body>
```

6.1 函数定义

- 函数名<name>：可以是任何有效的Python标识符
- 参数列表<parameters>：是调用函数时传递给它的值（可以由多个，一个，或者零个参数组成，当有多个参数时，各个参数用逗号分隔）
  - 参数个数大于等于零
  - 多个参数由逗号分隔

6.1 函数定义

- 形式参数：定义函数时，函数名后面圆括号中的变量，简称“形参”。形参只在函数内部有效
- 实际参数：调用函数时，函数名后面圆括号中的变量，简称 “实参”。

python 语言程序设计

6.1 函数定义

- 函数体<body>：函数被调用时执行的代码，由一个或多个语句组成。
- 函数调用的一般形式：

<name>(parameters>)

python 语言程序设计

6.1 函数定义

- 例:

```
def add1(x):
    x = x + 1
    return x
```

- 函数功能：将传给它的数值增1，返回增加后的值
- return语句：结束函数调用，并将结果返回给调用者
- return语句是可选的，可出现在函数体的任意位置
- 没有return语句，函数在函数体结束位置将控制权返回给调用方

6.1 函数定义

- 上例调用过程及运行结果：

```
>>> add1(1)
2
```

- 函数接口：返回值和参数
- 函数传递信息的主要途径：
  - 通过函数返回值的方式传递信息
  - 通过参数传递信息

6.2 示例程序: 生日快乐

- 例子1：编写一个程序打印“Happy Birthday”的歌词
- 标准的歌词：

Happy Birthday to you!

Happy Birthday to you!

Happy Birthday, dear <insert-name>

Happy Birthday to you!

6.2 示例程序: 生日快乐

- 方法1：使用四个print语句
  - 给Mike唱生日快乐歌的程序代码：

```
print("Happy Birthday to you!")
print("Happy Birthday to you!")
print("Happy Birthday, dear Mike!")
print("Happy Birthday to you!")
```

python 语言程序设计

6.2 示例程序: 生日快乐

- 运行结果：

```
>>>
Happy Birthday to you!
Happy Birthday to you!
Happy Birthday, dear Mike!
Happy Birthday to you!
>>>
```

python 语言程序设计

6.2 示例程序: 生日快乐

- 方法2：使用函数来打印歌词的第一、二、四行
- 定义函数happy()：

```
>>> def happy():
        print("Happy birthday to you!")
```

- 调用happy()结果：


```
>>> happy()
Happy birthday to you!
```

6.2 示例程序: 生日快乐

- 定义函数实现为Mike打印生日歌的歌词：

```
def singMike():
    happy()
    happy()
    print("Happy birthday, dear Mike!")
    happy()
```

6.2 示例程序: 生日快乐

- 调用singMike()结果：

```
>>> singMike()
Happy birthday to you!
Happy birthday to you!
Happy birthday, dear Mike!
Happy birthday to you!
```

python 语言程序设计

6.2 示例程序: 生日快乐

- 例子2：写出给Mike和Lily唱生日歌的程序
  - 给Lily唱生日歌的程序

```
def singLily():
    happy()
    happy()
    print("Happy birthday, dear Lily!")
    happy()
```

python 语言程序设计

6.2 示例程序: 生日快乐

- 完整代码：

```
singMike()
print()
singLily()
```

- print()用来在输出两段歌词之间打印一个空行

6.2 示例程序: 生日快乐

- 运行结果：

```
>>>
Happy birthday to you!
Happy birthday to you!
Happy birthday, dear Mike!
Happy birthday to you!

Happy birthday to you!
Happy birthday to you!
Happy birthday, dear Lily!
Happy birthday to you!
```

6.2 示例程序: 生日快乐

- 例子3：简化程序，编写通用函数唱生日歌

```
def sing(person):
    happy()
    happy()
    print("Happy birthday, dear", person + "!")
    happy()
```

- person参数：此变量在函数被调用时初始化

6.2 示例程序: 生日快乐

- sing()函数只需在函数调用的时候提供名字作为参数
- 给Mike和Lily打印歌词：

```
>>> sing("Mike")
Happy birthday to you!
Happy birthday to you!
Happy birthday, dear Mike!
Happy birthday to you!
>>> sing("Lily")
Happy birthday to you!
Happy birthday to you!
Happy birthday, dear Lily!
Happy birthday to you!
```

6.2 示例程序: 生日快乐

- 综合例子：利用sing()，为Mike、Lily和Elmer三个人唱生日歌
- 完整代码：

```
def happy():
    print("Happy birthday to you!")
def sing(person):
    happy()
    happy()
    print("Happy birthday, dear", person + "!")
    happy()
def main():
    sing("Mike")
    print()
    sing("Lily")
    print()
    sing("Elmer")
main()
```