《Python》课程资料

第一章 Python 简介

1.1 Python 的优势和应用领域

1.1.1 Python 是什么

Python 是一种解释型、面向对象、动态数据类型的高级程序设计语言。

1.1.2 为什么要学 Python

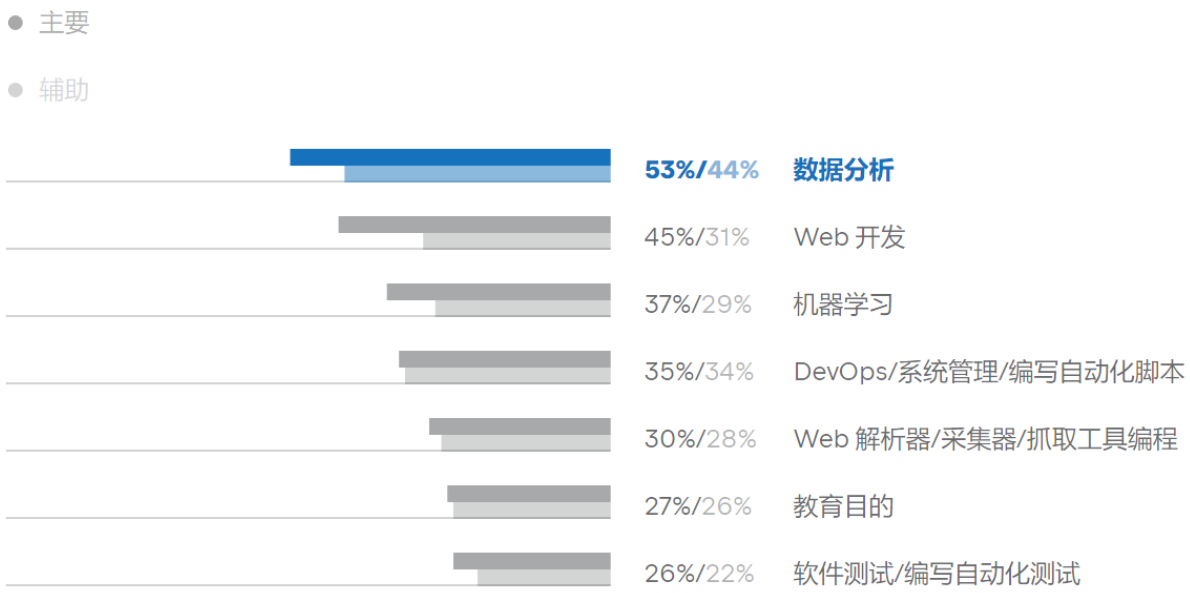


图 1-1 Python 作为主要语言与作为辅助语言的使用

1.1.3 如何学 Python

从“不知道怎么办”到“大概知道哪里可以找答案；

告别复制粘贴，回到复制粘贴；

理解了某一问题的本质，就会解决所有类似的问题。

1.2 Python 的安装和环境配置

Pycharm + Python 的四部曲：

a) 下载安装 Python 3

b) 下载安装 Pycharm

c) 新建一个 Python 环境

d) 新建一个项目并配置环境

关注公众号：【数模加油站】，免费领取更多数模相关资料

第二章 Python 基础

2.1 基础语法

2.1.1 标识符与保留字

a) 第一个字符必须是字母表中字母或下划线 _ 。

b) 标识符的其他的部分由字母、数字和下划线组成。

c) 标识符对大小写敏感。

d) 在 Python 3 中，可以用中文作为变量名，非 ASCII 标识符也是允许的了。

e) 保留字有：'False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield'

2.1.2 注释

a) Python 中单行注释以 # 开头

b) 多行注释可以用多个 # 号，还有 ''' 和 """：

注释示例

```
# 第一个注释
# 第二个注释

'''
第三注释
第四注释
'''

"""
第五注释
第六注释
"""
print ("Hello, Python!")
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

2.1.3 行与缩进

python 最具特色的就是使用缩进来表示代码块，不需要使用大括号 {} 。缩进的空格数是可变的，但是同一个代码块的语句必须包含相同的缩进空格数。实例如下：

用于构成条件判断的代码块缩进

```
if a==1:
    print (a)
    print ("Ture")
    print ("a 是等于 1 的")

print ("Go! ")
```

用于构成条件判断的代码块缩进示例 2

```
# 缩进
a = 1
b = 2
if b == 1:
    print(a)
    print(b)
    print("b=2")
```

2.1.4 输入和输出

print 默认输出是换行的，如果要实现不换行需要在变量末尾加上 end="":

输出示例

```
x="a"
y="b"
# 换行输出
print( x )
print( y )
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
print('---------')
# 不换行输出
print( x, end=" " )
print( y, end=" " )
print()
```

2.2 基本数据类型

2.2.1 标准数据类型

a) Python 中的变量不需要声明。每个变量在使用前都必须赋值，变量赋值以后该变量才会被创建。

b) 在 Python 中，变量就是变量，它没有类型，我们所说的"类型"是变量所指的内存中对象的类型。

c) 等号（=）用来给变量赋值。

d) 等号（=）运算符左边是一个变量名,等号（=）运算符右边是存储在变量中的值。

常见的几个标准数据类型示例

```
counter = 100
# 整型变量
miles   = 1000.0
# 浮点型变量
Var = True
# 布尔型变量
name     = "run"
# 字符串
print (counter)
print (miles)
print (Var)
print (name)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

2.2.2 数据类型转换

有时候，我们需要对数据内置的类型进行转换，数据类型的转换，一般情况下你只需要将数据类型作为函数名即可。Python 数据类型转换可以分为两种：

a) 隐式类型转换 - 自动完成

b) 显式类型转换 - 需要使用类型函数来转换

数据类型转换

```
a = 100   # 整型
print(type(a))

b = 100.0   # 浮点数
print(type(b))

c = True   # bool
print(type(c))

d = 'd'   # 字符和字符串
e = "eeee"
print(type(d))
print(type(e))

# 数据类型的转换
ab = a + b
print("ab 的数据类型", type(a), type(b), type(ab))
ab = a + int(b)
print("ab 的数据类型", type(a), type(b), type(ab))
ab = int(a + b)
print("ab 的数据类型", type(a), type(b), type(ab))
d = '0'
ad = a + int(d)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
print(ad)


print("str(a)=", str(a), type(str(a)))


h = "100.5"
bh = b + float(h)    # unsupported operand type(s) for +: 'float' and 'str'
print(bh, type(bh))
```

2.3 字符串基本操作

2.3.1 字符串的定义

字符串是由字符组成的序列，可以使用单引号（'）或双引号（"）括起来。例如：name = ‘Alice’。

2.3.2 字符串的索引和切片

字符串中的每个字符都有一个索引，可以使用索引访问特定位置的字符。索引从 0 开始。例如：print(name[0]) 将输出第一个字符‘A’。切片是指通过指定起始索引和结束索引来提取子字符串。例如：print(name[1:3]) 将输出‘li’。

字符串的索引和切片示例

```
#name=-101234
name = "Alice"


# 索引  切片


print(name[0])
print(name[1:3])    # 结束位置的索引并不会输出
print(name[1:4])    # 输出的长度是 结束的索引 - 开始的索引


# -1 负数索引
print(name[-2])
print(name[-4:-2])
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

2.3.3 字符串的拼接

字符串的**拼接**：使用加号（+）可以将两个字符串拼接在一起。例如：greeting = “Hello” + name。

字符串的拼接示例
拼接 greeting = "Hello" print(greeting + name) print(greeting + " " + name) # 1 + 1 = 2; 1 x 2 = 2 print("1" + "1") print("1" * 2) print("Alice" * 5) # str = "iloveu" print(str[-3:-1] * 2 + str[2:4])

2.3.4 课后练习

1、索引和切片操作练习

请编写代码，从字符串”大家好我是练习时长两年半的蔡徐坤”中提取出蔡徐坤的练习时长
info = ”大家好我是练习时长两年半的蔡徐坤” print(info[9:12])

2、拼接操作练习

请编写代码， 用最简洁的方式用 print()函数画出点划线、双虚线和直线各一条

关注公众号：【数模加油站】，免费领取更多数模相关资料

```
print("_."*20)
print("="*40)
print("——"*20)
```

2.4 运算符

2.4.1 算术运算符

运算符	描述
+	加 - 两个对象相加
-	减 - 得到负数或是一个数减去另一个数
*	乘 - 两个数相乘或是返回一个被重复若干次的字符串
/	除 - x 除以 y
%	取模 - 返回除法的余数
**	幂 - 返回 x 的 y 次幂
//	取整除 - 往小的方向取整数

算术运算符示例

```
a = 2.00000000000000000001
b = 2
print("a + b = ", a + b)
print("a - b = ", a - b)
print("a * b = ", a * b)
print("a / b = ", a / b, type(a), type(a / b))  # 数据类型转换
print("a ** b = ", a ** b)
print("a // b = ", a // b)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
print("a % b = ", a % b)
```

2.4.2 比较运算符

运算符	描述
==	等于 - 比较对象是否相等
!=	不等于 - 比较两个对象是否不相等
>	大于 - 返回 x 是否大于 y
<	小于 - 返回 x 是否小于 y。
>=	大于等于 - 返回 x 是否大于等于 y。
<=	小于等于 - 返回 x 是否小于等于 y。

比较运算符示例

```
a = 2.00000000000000000001
b = 2
# 比较运算符
print("a > b :", a > b)
print("a < b :", a < b)
print("a >= b :", a >= b)
print("a <= b :", a < b)
print("a == b :", a == b)
print("a != b :", a != b)
```

2.4.3 赋值运算符

运算符	描述

关注公众号：【数模加油站】，免费领取更多数模相关资料

=	简单的赋值运算符
+=	加法赋值运算符
-=	减法赋值运算符
*=	乘法赋值运算符
/=	除法赋值运算符
%=	取模赋值运算符
**=	幂赋值运算符
//=	取整除赋值运算符

赋值运算符示例

```
a = 1
a = a + 1
print(a)
a += 1    # a = a + 1  等效的
print(a)
a -= 1    # a = a - 1  等效的
print(a)
a *= 2    # a = a * 2  等效的
print(a)
a /= 3    # a = a / 3  等效的
print(a)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

2.4.4 逻辑运算符

逻辑运算符	描述
and	布尔"与" - 如果 x 为 False，x and y 返回 x 的值，否则返回 y 的计算值。
or	布尔"或" - 如果 x 是 True，它返回 x 的值，否则它返回 y 的计算值。
not	布尔"非" - 如果 x 为 True，返回 False 。如果 x 为 False，它返回 True。

逻辑运算符示例

```
# 逻辑运算符 and or not
a = 1
b = 3
print(a == 1 and b == 2)
print(a == 1 or b == 2)
print(not b == 2)
# Ture 1    False 0
print(1 and 0)
print(1 or 0)
print(not 0)
```

2.4.5 成员运算符

成员运算符	描述
in	如果在指定的序列中找到值返回 True，否则返回 False。

关注公众号：【数模加油站】，免费领取更多数模相关资料

not in	如果在指定的序列中没有找到值返回 True，否则返回 False。

成员运算符示例

```
# 成员运算符 in  not in
string = "hello alice"
char = "o a"
print(char in string)
```

2.4.6 课后练习

输入：
完成一个加法器，通过用户输入两个数字。
输出：
告诉用户他们相加的和是多少。

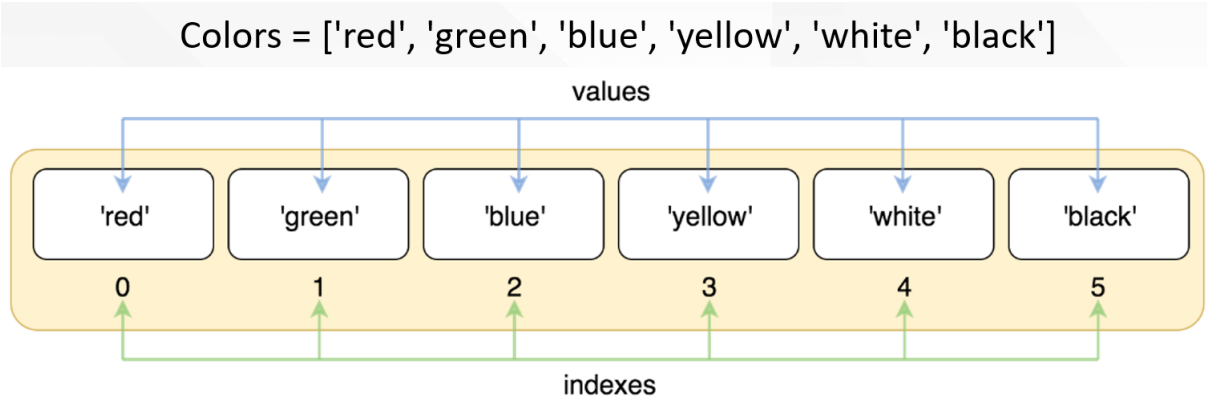
```
a = input("请输入第一个数字")
b = input("请输入第二个数字")


print("输入的两个数字之和是：", int(a)+int(b))
```

2.5 复合数据类型

2.5.1 列表

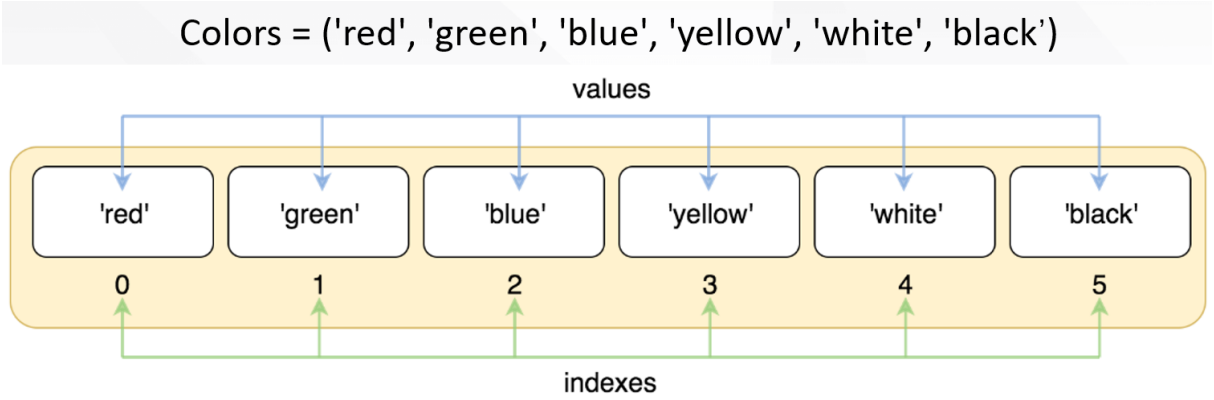
列表是最常用的 Python 数据类型，它可以作为一个方括号内的逗号分隔值出现。列表的数据项不需要具有相同的类型。创建一个列表，只要把逗号分隔的不同的数据项使用方括号括起来即可。

关注公众号：【数模加油站】，免费领取更多数模相关资料



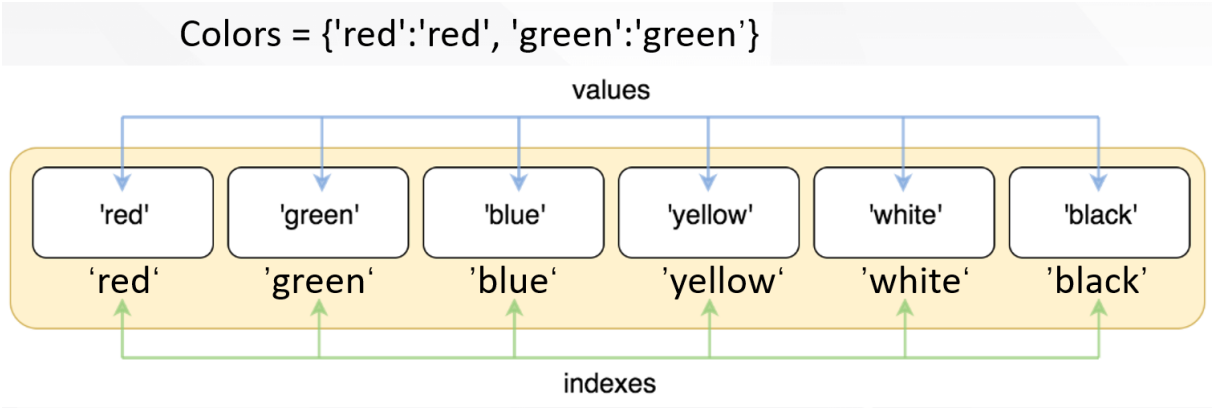
2.5.2 元组

Python 的元组与列表类似，不同之处在于元组的元素不能修改。元组使用小括号 ()，列表使用方括号 []。元组创建很简单，只需要在括号中添加元素，并使用逗号隔开即可。



2.5.3 字典

字典是另一种可变容器模型，且可存储任意类型对象。字典的每个键值 key=>value 对用冒号 : 分割，每个对之间用逗号(,)分割，整个字典包括在花括号 {} 中。



2.5.4 课后练习

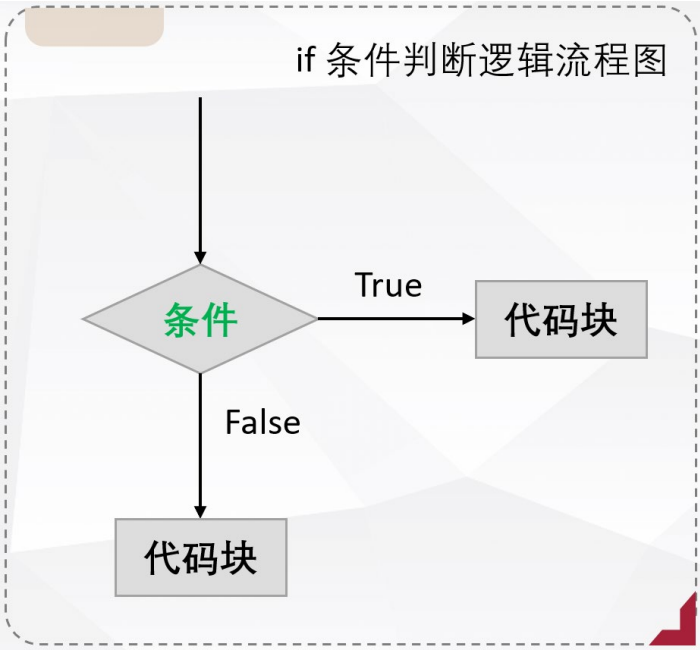
输入：

关注公众号：【数模加油站】，免费领取更多数模相关资料

一个可能包含重复元素的列表。 输出： 剔除了重复元素的列表。
list_example = ["hello", 1, 2, 3, 4, 5, 6, 7, 7, 7, 8, 8, "hello"] str() int() print(list(set(list_example)))

2.6 条件语句和循环语句

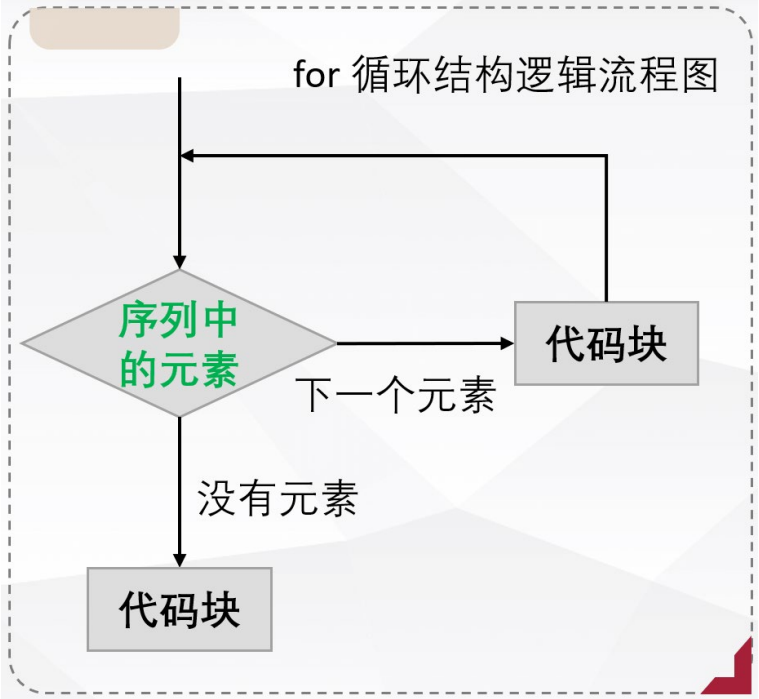
2.6.1 if 条件判断



If 示例
if a > 0: print("a 是正数") elif a == 0: print("a 是 0") else: print("a 是负数") print("a 的正负性判断完成")

关注公众号：【数模加油站】，免费领取更多数模相关资料

2.6.2 for 循环



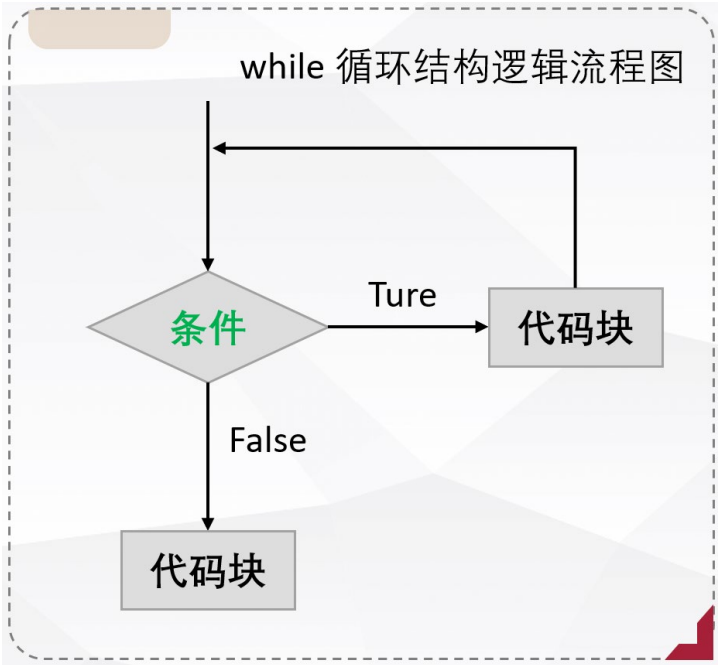
for 示例

```
for i in range(5):    # [0, 1, 2, 3, 4]
    print(i)
print("i 输出的结果是", i)
print("i%2", i%2)

for i in range(3):
    print(i)

for i in "saiouhbwiushadf":
    print(i)

for i in ["red", "blue", "green", "black"]:
    print("当前的颜色是", i)

num = 0
for i in range(5):
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
        for j in range(6):
            num += 1
            print("这是第", num, "次循环", i, j, i * j)
```

2.6.3 while 循环



while 示例

```
num = 0
while num < 100:
    num += 1
    print("这是第", num, "次循环")

    if num == 50:
        print("终止")
        break
```

2.6.4 猜数字

猜数字小游戏实现

```
target = 99
predict = 0
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
while int(predict) != target:    # "99"  ！ = 99
    predict = input("请进行一次猜测")
    if int(predict) > target:    # '>' not supported between instances of 'str' and 'int'
        print("猜大了！ ")
    elif int(predict) < target:
        print("猜小了！ ")
    else:
        print("两者相等")

print("恭喜猜中！ ")
```

2.6.5 课后练习

描述：

不停地接受用户输入，除非输入是 0，否则自动判断该数字是否为 3 的倍数。

输入：

一个数字。

输出：

是否为 3 的倍数。

```
while True:
    a_str = input("请输入一个整数")
    a_int = int(a_str)

    if a_int == 0:
        break

    if a_int % 3 == 0:
        print(a_str, "是 3 的倍数")
    else:
        print(a_str, "不是 3 的倍数")
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

2.7 函数

2.7.1 函数的定义

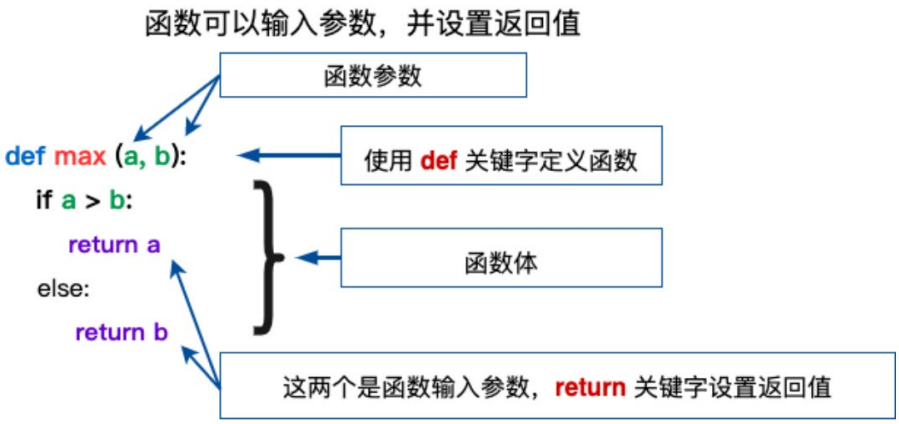
a) 函数代码块以 def 关键词开头，后接函数标识符名称和圆括号 ()。

b) 任何传入参数和自变量必须放在圆括号中间，圆括号之间可以用于定义参数。

c) 函数的第一行语句可以选择性地使用文档字符串—用于存放函数说明。

d) 函数内容以冒号 : 起始，并且缩进。

e) return [表达式] 结束函数，选择性地返回一个值给调用方，不带表达式的 return 相当于返回 None。



2.7.2 课后练习

描述：

判断一个数字是否为质数。

输入：

一个数字。

输出：

是否为质数。

```
def isprime(num):
    '''
    判断是否为质数
    :param num:
    :return:
    '''
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
        for i in range(2, num):
            if num % i == 0:
                # print("找到的第一个因数是：", i)
                return False
        return True



print(isprime(25))
```

2.8 文件操作

2.8.1 文件读取和写入

Python open() 方法用于打开一个文件，并返回文件对象。在对文件进行处理过程都需要使用到这个函数，如果该文件无法被打开，会抛出 OSError。注意：使用 open() 方法一定要保证关闭文件对象，即调用 close() 方法。open() 函数常用形式是接收两个参数：文件名(file)和模式(mode)。

模式	描述
r	以**只读**方式打开文件。文件的指针将会放在文件的开头。这是默认模式。
r+	打开一个文件用于**读写**。文件指针将会放在文件的**开头**。
w	打开一个文件只用于写入。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。
w+	打开一个文件用于读写。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。
a	打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的**结尾**。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。

关注公众号：【数模加油站】，免费领取更多数模相关资料

2.8.2 文件对象及方法

函数	描述
file.close()	关闭文件。关闭后文件不能再进行读写操作。
file.read([size])	从文件读取指定的字节数，如果未给定或为负则读取所有。
file.readline([size])	读取整行，包括 "\n" 字符。
file.readlines([sizeint])	读取所有行并返回列表，若给定 sizeint>0，返回总和大约为 sizeint 字节的行，实际读取值可能比 sizeint 较大，因为需要填充缓冲区。
file.write(str)	将字符串写入文件，返回的是写入的字符长度。
file.writelines(sequence)	向文件写入一个序列字符串列表，如果需要换行则要自己加入每行的换行符。

文件写入示例

```
    with open(file="sample.txt", mode="a", encoding="utf-8") as file_sample:   # w 仅读取   a 追加
        file_sample.write("大家\t 好！ \n")
```

2.8.3 课后练习

描述：将 100 以内的质数都存到一个文件夹中，每行只能写十个数字。

输入：

无。

输出：

文件。

```
    with open(file="prime.txt", mode="w") as file:
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
count = 0
for num in range(2, 100):
    if isprime(num):
        # 存进文件
        print(num)
        count += 1
        file.write(str(num) + ", ")    # write() argument must be str, not int
        if count % 10 == 0:
            file.write("\n")
```

第三章 数据处理与可视化

3.1 模块安装与导入

pip 是 Python 包管理工具，该工具提供了对 Python 包的查找、下载、安装、卸载的功能。

pip 指令示例。
pip --version pip install some-package-name pip uninstall some-package-name pip list

3.2 数值计算 Numpy

pip 安装指令
pip install numpy

3.2.1 数组的创建

示例
arr = np.array([1, 2, 3, 4, 5])

关注公众号：【数模加油站】，免费领取更多数模相关资料

```
print(arr)
print(type(arr))    # numpy.ndarray

arr = np.array([[1, 2, 3], [2, 3, 4], [3, 4, 5], [4, 5, 6]])
print(arr)
print(type(arr))    # numpy.ndarray
```

3.2.2 索引和切片

示例

```
print(arr[0])
print(arr[0:3])
print(arr[1][2])
```

3.2.3 运算

示例

```
print([1, 2, 3] + [4, 5, 6])
print(np.array([1, 2, 3]) + np.array([4, 5, 6]))
print(np.array([1, 2, 3]) * np.array([4, 5, 6]))
```

3.2.4 数组形状操作

示例

```
arr = np.array([[1, 2, 3], [2, 3, 4], [3, 4, 5], [4, 5, 6]])
print(arr.shape)
new_arr = arr.reshape(2, 6)
print(new_arr, "新的数组的形状是", new_arr.shape)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
    new_arr_T = new_arr.transpose()
    print(new_arr_T, "新的数组的转置的形状是", new_arr_T.shape)
```

3.2.5 进阶使用

线性代数、统计的示例

```
    arr1 = np.array([1, 2, 3])
    arr2 = np.array([4, 5, 6])
    arr1_dot_arr2 = np.dot(arr1, arr2)
    print(arr1_dot_arr2)


    arr = np.array([[5, 2, 3], [2, 7, 4], [99, 4, 5], [14, 15, 6]])
    print("数组的平均值", np.mean(arr))
    print("数组的平均值", arr.mean())
    print("数组的最大值", np.max(arr))
    print("数组的最小值", arr.min())
    print("数组的标准差", arr.std())
    print("数组的和", arr.sum())
    print("数组的排序", np.sort(arr.reshape(-1)))   # numpy 的函数
    print(arr[(arr > 10) | (arr < 5)])   # 筛选      & 按位 与   等效于在每一位上进行一次 and
                                                    # 筛选      | 按位 或   等效于在每一位上进行一次 or
```

3.2.6 npy 保存和导入

示例

```
    np.save("arr", arr)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

3.2.7 课后练习

描述：

用 numpy 创建一个 4x4 随机数组，只保留 10 以内的数，并计算出所有元素的和。

```
import numpy as np
np.random.seed(123444)  # 固定随机种子
print(np.random.rand())  # 0-1 之间随机浮点数
arr = np.random.randint(0, 100, 16).reshape(4, 4) # 0-100 之间的随机整数
print(np.sum(arr[arr <= 10]))
```

3.3 表格处理 Pandas

pip 安装指令

```
pip install pandas
```

3.3.1 excel 读取

示例

```
import numpy as np
import pandas as pd
df = pd.read_excel("鸢尾花训练数据.xlsx", "Sheet1", engine="openpyxl")
# print(df.head(10))
print(type(df))


arr = np.array([1, 3, 4])


data = {'样本号': [1, 2, 3], '萼片长(cm)': [8.9, 2.1, 4.5], '类型_num': [0, 0, 1]}
datadf = pd.DataFrame(data)
print(datadf)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
# 基础信息
print(df.head())
print(df.info())
```

3.3.2 缺失值处理

示例

```
df = df.dropna()
```

3.3.3 条件筛选

示例

```
df_1 = df[df['类型_num'] == 1]
print(df_1.head())
print(df_1.info())



lb = df['花瓣宽(cm)'].mean() - 3 * df['花瓣宽(cm)'].std()
ub = df['花瓣宽(cm)'].mean() + 3 * df['花瓣宽(cm)'].std()

selected_df = df[(df['花瓣宽(cm)'] >= lb) & (df['花瓣宽(cm)'] <= ub)]
print(selected_df)
```

3.3.4 课后练习

描述：

自己创建一个 dataframe，包含：姓名、身高、体重、成绩。输出第一名的学生，计算平均分并输出低于 60 分的同学名字

```
import numpy as np
import pandas as pd
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
data = {'姓名': ["张三", "李四", "王五", "老六", "赵七"],
        '身高': [175 for i in range(5)],
        '体重': [50 for i in range(5)],
        '成绩': np.random.randint(40, 90, 5)}
df = pd.DataFrame(data)
print(df)
print(df[df['成绩'] == max(df['成绩'])])
print("本班的平均分是", np.mean(df['成绩']))
new_df = df[df['成绩'] < 60]
print("本班这次考试不及格的同学有：", new_df["姓名"])
```

3.4 可视化 Matplotlib

pip 安装指令
pip install matplotlib

3.4.1 Plot 绘制

画一个 sin 函数
x = np.linspace(0, 10, 10) # print(x) y = np.sin(x) plt.plot(x, y) plt.title("y = sin(x)") plt.xlabel("x") plt.ylabel("y") plt.show()

关注公众号：【数模加油站】，免费领取更多数模相关资料

3.4.2 Scatter 绘制

散点图绘制
plt.scatter(x, y, marker='*', c='r', label="数据点") plt.legend() plt.show()

3.4.3 中文支持

Plt 图像的中文显示问题
from matplotlib.pylab import mpl mpl.rcParams['font.sans-serif'] = ['SimHei'] mpl.rcParams['axes.unicode_minus'] = False

3.4.4 多张图片

Subplot 绘制示例
fig, axes = plt.subplots(1, 2) axes[0].scatter(x, y, marker='*', c='r', label="数据点") axes[0].set_xlabel("x1") axes[0].set_ylabel("y") axes[0].set_title("数据点") axes[1].plot(x2, y2, linestyle='--', label="拟合结果") axes[1].set_xlabel("x2") axes[1].set_title("拟合结果") fig.legend() fig.show()

关注公众号：【数模加油站】，免费领取更多数模相关资料

3.4.5 课后练习

描述：

https://matplotlib.org/stable/plot_types/index.html

自己尝试一个图像的绘制。

```
import matplotlib.pyplot as plt
import numpy as np


plt.style.use('_mpl-gallery')


# make data
np.random.seed(1)
x = 4 + np.random.normal(0, 1.5, 200)


# plot:
fig, ax = plt.subplots()


ax.hist(x, bins=8, linewidth=0.5, edgecolor="white")


ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 56), yticks=np.linspace(0, 56, 9))


plt.show()
```

第四章 项目实战

4.1 简单算法

使用二分法求解超越方程　e^x = pi 的解

```
import numpy as np
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
def f(x):
    # f(x) = e^x - pi
    return np.e ** x - np.pi


print(f(10))
# f(x1) > 0 f(x2) < 0    (x1, x0, x2)
# f(x0) == 0
resolution = 0.00001
# np.abs(f(x0) - 0) <= resolution



def search_x(x1, x2):
    print("调用了一次 search_x")
    x0 = (x1 + x2) / 2
    if np.abs(f(x0) - 0) <= resolution:
        print("二分法找到的解", x0)
    elif f(x1) * f(x0) < 0:
        #  解在  (x1, x0)
        search_x(x1, x0)
    elif f(x0) * f(x2) < 0:
        #  解在  (x0, x2)
        search_x(x0, x2)


print("方程的根", np.log(np.pi))
search_x(0, 10)
```

4.2 简单建模

曲线拟合:

一个成年人饮酒后血液中的酒精含量随时间的变化是：

关注公众号：【数模加油站】，免费领取更多数模相关资料

时间(h)　　[0.25, 0.5, 0.75, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

酒精含量(mg/100ml)　[30, 68, 75, 82, 82, 77, 68, 68, 58, 51, 50, 41, 38, 35, 28, 25, 18, 15, 12, 10, 7, 7, 4]

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import polyfit

time = [0.25, 0.5, 0.75, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]
alcohol = [30, 68, 75, 82, 82, 77, 68, 68, 58, 51, 50, 41, 38, 35, 28, 25, 18, 15, 12, 10, 7, 7, 4]
y = [np.log(a) for a in alcohol]

alcohol_tup = alcohol[alcohol.index(max(alcohol)):]
time_tup = time[alcohol.index(max(alcohol)):]
y_tup = y[alcohol.index(max(alcohol)):]

k, b = polyfit(time_tup, y_tup, 1)
print(k, b)   # 拟合结果


def model(t):
    a = np.e ** (k * t + b)
    return a


time0 = np.linspace(time_tup[0], 16, 1000)
predy = model(time0)
```


关注公众号：【数模加油站】，免费领取更多数模相关资料

```
plt.scatter(time, alcohol, label="sample point")
plt.plot(time0, predy, c='r', label="fitting results")
plt.title("alcohol change with time")
plt.ylabel("alcohol [mg/100ml]")
plt.xlabel("time [h]")
plt.legend()
plt.show()
```


关注公众号：【数模加油站】，免费领取更多数模相关资料