

Python 程序设计

5.1 Python 程序设计简介

本章内容包括程序设计的基本概念与分类,重点介绍 Python 编程基础,包括语言基础、控制结构、函数与模块和 Python 数据库编程,具体见图 5.0。

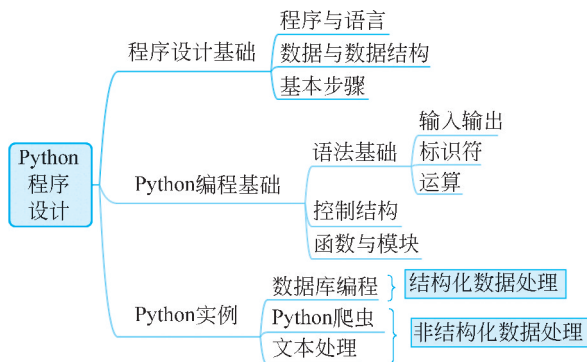


图 5.0 Python 程序设计

5.1.1 程序设计概述

1. 程序

所谓程序,是计算机为完成某一个任务遵循一定规则和算法思想组织起来并执行的一系列代码(也称为指令序列)。通常,计算机程序需要描述两部分内容:一是描述问题涉及的每个对象及它们之间的关系;二是描述处理这些对象的规则。其中前者涉及数据结构的内容,而后者则是求解问题的算法。因此,对于程序的描述,可以用经典的公式来表示:

$$\text{程序} = \text{算法} + \text{数据结构}$$

2. 程序设计与程序设计语言

所谓程序设计,就是根据计算机要完成的任务,设计解决问题的数据结构和算法,然后编写相应的程序代码,并测试该代码运行的正确性,直到能够得到正确的运行结果为止。程序设计应遵循一定的方法和原则,良好的程序设计风

格是程序具备可靠性、可读性、可维护性的基本保证。

程序设计语言是编写程序所使用的计算机语言。随着计算机技术的发展,程序设计语言经历了从机器语言、汇编语言到高级语言的发展历程。

高级语言的出现为计算机的应用开辟了广阔的前景。高级语言有很多种,虽然它们的特点各不相同,但编程解决问题的过程和一些基本的程序设计规则和方法却是相似的。因此在学习某种语言后,应该具有将其中共性的思想和方法迁移到其他语言环境中进行问题求解的能力。

3. 标识符与变量

标识符是用来对变量、函数、类等数据对象命名的有效字符串序列。通常,标识符只能由字母、数字和下画线组成,且第一个字符必须为字母或下画线,不能以数字开头,标识符中不能出现标点符号或运算符。语言本身的关键字,也称为保留字,如变量、函数、类等,不能作为标识符使用。

变量是指其值可以改变的量,每个变量都有一个变量名,对应计算机内存中具有特定属性的一个存储单元。该单元用来存储变量的值,在程序运行期间,这个单元中的值是可以改变的。变量通过变量名访问,变量的命名必须遵循标识符的命名规则。

4. 程序设计的基本步骤

对于初学者来说,往往把程序设计简单地理解为只是编写一个程序,这是不全面的。程序设计反映了利用计算机解决问题的全过程,包含多方面的内容,而编写程序只是其中的一个方面。使用计算机解决实际问题,通常先要对问题进行需求分析并建立数学模型,然后考虑数据的组织方式和算法,并用某一种程序设计语言编写程序,最后上机调试程序,使之运行后产生预期的结果。

5.1.2 Python 语言基础

Python 语言是一种解释型、面向对象、动态数据类型的高级程序设计语言。从 20 世纪 90 年代初诞生至今,已逐渐成为最受欢迎的程序设计语言之一。Python 语言简单易学,具有强大的数据处理能力,并且是一种通用的程序设计语言,既适合作为程序设计的入门语言,也适合用来作为解决数据分析等各类问题的通用工具。

1. 输入与输出

(1) 输入函数 `input()`。

`input()` 函数用于获得用户输入的数据,其基本格式如下:

```
变量 = input("提示字符串")
```

其中,变量和提示字符串均可省略。`input()` 函数将用户输入以字符串返回。如果需要输入整数或小数,则需要使用 `eval()`、`int()` 或 `float()` 函数进行转换。

(2) 输出函数 `print()`

Python 中使用 `print()` 函数完成基本输出操作。`print()` 函数的基本格式如下:

```
print([obj1,...][,sep=""][,end=""],[,file=sys.stdout])
```

2. Python 数据类型

Python 是动态类型语言,变量不需要显式声明数据类型,对其直接进行赋值即可使用,Python 语言的解释器会根据变量的赋值自动确定其数据类型。通过内置的 `type()` 函数,可以测试一个变量的数据类型。

Python 中的数据类型主要包括数字(整数、浮点数、复数)、字符串(文本)、列表(有序的元素集合)、元组(不可变的列表)、集合(无序且不重复的元素集合)、字典(键值对的集合)以及布尔值(True 或 False)。比较特殊的是,Python 的空值(None)用于表示缺失或未定义的值,在函数无返回值、变量初始化、占位符、默认参数值以及条件语句等多个场景中都发挥着重要作用,为 Python 编程提供了灵活性和明确性。Python 的数据类型总结见表 5.1。

表 5.1 Python 的数据类型总结

数据类型	描述	示例	应用场景
整数 (int)	表示整数值,可以是正数、负数或零	<code>x=10</code>	数学运算、计数、索引等
浮点数 (float)	表示带有小数部分的数值	<code>y=3.14</code>	浮点数计算、科学计算、金融应用等
复数 (complex)	由实数部分和虚数部分组成的数值	<code>z=2+3j</code>	电气工程、信号处理、物理学等
字符串 (str)	表示文本数据,用单引号或双引号括起来	<code>name="Alice"</code>	文本处理、用户界面、文件操作等
列表 (list)	表示可变序列,可以存储多个元素,用方括号括起来	<code>numbers=[1,2,3,4,5]</code>	数据收集、数据处理、算法实现等
元组 (tuple)	表示不可变序列,用圆括号括起来	<code>colors=('red','green','blue')</code>	数据保护、函数返回多个值、解构赋值等
字典 (dict)	表示键值对的集合,用花括号括起来	<code>person={'name':'Bob','age':30}</code>	数据存储、数据检索、配置信息等
集合 (set)	表示无序不重复元素的集合,用花括号括起来	<code>fruits={'apple','banana','orange'}</code>	去重、集合运算、成员检查等
布尔 (bool)	表示逻辑值 True 或 False	<code>is_valid=True</code>	判断条件、逻辑运算、控制流程等
空值 (NoneType)	表示空值或缺少值	<code>x=None</code>	初始化变量、函数返回值缺失等

在数学运算中,True 和 False 分别对应于 1 和 0。`bool()` 是布尔型的转换函数,可以将其他数据类型转换为布尔型。Python 也支持八进制、十六进制或二进制来表示整型。浮点型(float)也称小数,可以直接用十进制或科学记数法表示。浮点数通常都有一个小数点和一个可选的后缀 e(大写或小写,表示科学记数法)。在 e 和指数之间可以用“+”

或“-”表示正负,正数“+”号可以省略。float()是浮点型的转换函数,可以将其他数据类型转换为浮点型。

在 Python 中,字符串可以用单引号、双引号或三引号括起,但必须配对,其中三引号既可以是三个单引号,也可以是三个双引号。字符串类型比较是按照字符的编码值的大小进行的。

成员运算符是用来判断一个元素是否属于一个序列的,对于字符串类型来说,就是判断一个字符(也可以是一个子串)是否出现在一个字符串中。成员运算符用“in”或“not in”表示,返回值是布尔值 True 或 False。

3. Python 切片运算

Python 切片运算的作用是通过指定下标或索引范围来获得一个序列的一组元素,可以快速地提取列表、字符串、元组等序列类型中的子集,实现数据的快速访问和处理。切片运算使得代码简洁易读,同时提供了丰富的功能,如逆序访问、步长控制等,为程序员提供了便利和效率。Python 中常用的切片运算见表 5.2。

表 5.2 Python 中常用的切片运算

切片运算	描 述	示 例
列表切片	从列表中提取指定范围的元素	lst=[1,2,3,4,5] print(lst[1:4]) # 输出[2,3,4]
字符串切片	从字符串中提取指定范围的字符	s="Hello,World!" print(s[1:5]) # 输出"ello"
元组切片	从元组中提取指定范围的元素	t=(1,2,3,4,5) print(t[2:]) # 输出(3,4,5)

字符串中的字符按顺序编号,最左边字符的序号为 0,最右边字符的序号比字符串的长度小 1。Python 还支持在字符串中使用负数从右向左进行编号,最右边字符(即倒数第 1 个字符)的序号为-1。字符在字符串中的序号也称为下标或索引,可以通过索引获取字符串中的字符。表 5.3 包括了正向切片、逆向切片、步长切片以及反转序列等常见的切片操作。

表 5.3 Python 常见的切片操作

切片运算	描 述	示 例
正向切片	从左往右提取指定范围的元素	lst = [1, 2, 3, 4, 5] print(lst[1:4]) # 输出 [2, 3, 4]
逆向切片	从右往左提取指定范围的元素	lst = [1, 2, 3, 4, 5] print(lst[-3:-1]) # 输出 [3, 4]
步长切片	按照指定步长提取元素	lst = [1, 2, 3, 4, 5] print(lst[:2]) # 输出 [1, 3, 5]
反转序列	反转整个序列	s = "hello" print(s[::-1]) # 输出 "olleh"

4. Python 列表与元组

列表是 Python 中重要的内置数据类型,是一个数据的有序序列,列表中数据的类型可以各不相同。列表中的每个数据称为元素,数据在列表中的序号也称为下标或索引。列表中的元素用一对方括号“[”和“]”括起来,元素之间用逗号分隔。与字符串一样,列表也有两种索引方式,元素序号如果从左向右则从 0 开始依次递增,如果从右往左则以 -1 开始,依次递减。

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。列表可以进行截取、组合等,使用下标索引来访问列表中的元素,同样也可以使用切片的形式截取列表中的一系列元素。使用加法运算符可以进行列表的连接操作,使用乘法运算符可以使列表重复多次,使用 in 运算符可以判断一个元素是否在列表中。

Python 的元组与列表类似,元组中的元素用一对圆括号“(”和“)”括起来。不同之处在于元组的元素不能被修改、删除,也不能往元组中新增元素。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。当元组中只包含一个元素时,需要在元素后面添加逗号,否则括号会被当作运算符使用。元组是一个序列,所以可以访问元组中指定位置的元素,也可以用切片截取其中的一系列元素,方法与列表相同。

与列表一样,元组也可以使用连接运算符(+)、复制运算符(*)、in/not in 运算符。元组的相关函数有 len()、max()、min()、sum()、sorted(),相关方法有 count()、index(),函数 tuple()可以将其他序列转换为元组。

5.1.3 Python 程序控制结构

1. 分支结构

单分支结构只有一个条件判断;双分支结构包含两个代码块,一个对应条件为真时的执行,另一个对应条件为假时的执行;多分支结构适用于需要基于多个条件做出不同选择的情况,例如根据学生的成绩判断其等级。

2. 循环结构

在 Python 中,循环语句用于重复执行特定代码块,直到满足一定条件为止。主要的循环语句包括 while 循环(条件型循环)和 for 循环(计数型循环),这两种循环语句的区别见表 5.4。

表 5.4 Python 循环语句对照

特点	条件型循环 (while)	计数型循环 (for)
运行次数	取决于条件表达式真假,直到条件为假结束循环	取决于序列中元素个数,遍历完所有元素结束循环
使用场景	不确定循环次数,条件可能在循环体内改变	确定循环次数,遍历容器类型数据(如列表、元组)

续表

特点	条件型循环（while）	计数型循环（for）
动态性	需要考虑条件变化,潜在风险是无限循环	固定循环次数,无须考虑条件变化
示例	<pre>count = 0 while count < 5: print(count) count += 1</pre>	<pre>numbers = [1, 2, 3, 4, 5] for num in numbers: print(num)</pre>

条件型循环（while 循环）是一种基于条件表达式的循环结构。在 while 语句中,执行循环体的次数取决于条件表达式的值是否为真。只要条件为真,循环体将一直执行下去。当条件为假时,循环停止。这种循环适用于不确定循环次数的情况,因为循环次数由条件的真假来控制。

计数型循环（for 循环）是一种基于给定序列元素的循环结构。在 for 语句中,变量会被依次赋值为序列中的每个元素,然后执行循环体。这种循环适用于明确知道循环次数的情况,因为循环次数取决于序列中元素的个数。

在 Python 中可以用 continue 和 break 语句来控制循环结构程序的执行。continue 语句用于跳过当前循环体中的剩余语句,并继续进行下一轮循环的执行。break 语句用于终止整个循环,即使循环体中还有语句没有被执行。

在 Python 中,for 和 while 语句也可以和 else 子句一起使用。else 中的语句会在循环正常执行完的情况下被执行,即 while 循环不是通过 break 语句跳出循环的。

5.1.4 Python 函数与模块

Python 语言为用户提供了众多模块,一个模块其实就是一个 Python 文件,一个模块内往往包含了许多功能函数,用户只要将模块导入自己的程序中,就可以使用这些模块中的函数,这是一种代码的重用方式,它减少了程序员编写程序的代码量。Python 还允许用户自己定义一些程序自身所需的函数,这为编写程序提供了一种方便的手段。

1. 常用模块中的函数

在调用模块中的函数之前,先要使用 import 语句导入相应的模块,然后就可以访问模块中的任何函数,其方法是在函数名前加上模块名。

2. 用户自定义函数

在 Python 中,自定义函数是由用户自己编写的函数,可以根据需要定义函数名、参数和函数体,实现特定的功能,并在程序中进行调用。自定义函数的语法见表 5.5,具体要求。

- def 为自定义函数的关键字。函数的首行与末行之间是描述函数操作的语句序列,称为“函数体”。
- 函数名要符合标识符命名规则。
- 形参列表指明了从调用语句传递给被调用函数的变量,各变量名之间用逗号分

隔。若无参数,形参两旁的括号也不能省略。

- 函数体中 Return 语句的作用是把该语句后的表达式作为函数的返回值。若缺少该语句,函数以 None 作为返回值。

表 5.5 Python 自定义函数介绍

语 法	描 述	示 例
def 函数名(参数列表):	定义一个函数,参数可以为空或多个	<pre>def greet(name): print(f"Hello, {name}!")</pre>
返回值	使用 return 语句返回函数结果	<pre>def add(a, b): return a + b</pre>
默认参数	为函数参数指定默认值	<pre>def greet(name='Alice'): print(f"Hello, {name}!")</pre>
关键字参数	使用关键字参数传递参数值	<pre>def greet(name): print(f"Hello, {name}!") greet(name='Bob')</pre>

对于函数的参数,在函数定义中的参数称为形参,在函数调用时使用的参数称为实参。形参是变量,作为函数要处理的数据的输入口,用以被调用时接收实参的数据。实参是要传送给函数的数据,可以是常量、变量或表达式。定义函数时可以为形参提供默认值,在调用函数时如果没有提供相应的实参,则使用默认值。

3. 匿名函数

匿名函数是指没有函数名的简单函数,只可以包含一个表达式,不允许包含其他复杂的语句,表达式的结果就是函数的返回值。Python 定义匿名函数的关键字是 lambda,基本格式如下:

```
lambda <参数列表>:<表达式>
```

匿名函数是一个函数对象,可以把匿名函数赋值给一个变量,再利用变量来调用该函数。可以将匿名函数作为普通函数的返回值返回,也可以将匿名函数作为序列的元素。

5.1.5 Python 结构化数据处理

在 Python 中,结构化数据处理主要指对表格化数据的操纵和分析,这通常通过 pandas 库实现。Pandas 提供了两种核心数据结构:Series 和 DataFrame,它们分别用于处理一维和二维数据。这些数据结构支持复杂的标签操作、缺失数据处理、数据清洗、转换、聚合以及分组计算等任务。Pandas 也支持对日期和时间序列数据的特殊处理,使其成为数据分析领域的首选工具之一。此外,它与 Matplotlib 等可视化工具集成,能够方便地进行数据探索和结果展示。

Python 结构化数据处理与数据库编程紧密相关,因为在数据驱动的应用中,经常需从数据库中提取、处理和分析结构化数据。在实际应用中,Pandas 常与数据库连接库配合使用,使得从数据库中读取数据到 DataFrame 中进行加工变得无缝,以及将处理后

的数据写回数据库也变得直接。Python 提供了对大多数数据库的支持,使用 Python 中相应的模块,可以连接到数据库,进行查询、插入、更新和删除等操作,可以方便地设计满足各种应用需求的数据库应用程序。

以 Access 数据库为例,使用 Python 进行数据库编程的基本步骤如下。

(1) 导入 pyodbc 模块库。pyodbc 是一个 Python 的第三方模块库,提供了使用程序访问数据库的功能。

(2) 使用连接字符串建立与数据库的连接,指定要对其进行操作的数据库文件,返回 connection 连接对象。对于 Access 数据库,连接字符串为包含其所在的路径的数据库文件名以及所使用的 Access 驱动程序名。

(3) 调用 cursor() 函数创建游标对象 cursor。

(4) 使用游标对象 cursor 的 execute() 函数执行相应的数据库 SQL 命令,包括对数据的插入、删除、修改和查询等。

(5) 获取游标对象执行的结果集。

(6) 提交数据库事务。

(7) 关闭与数据库的连接。

5.1.6 Python 非结构化数据处理

Python 非结构化数据处理可分成两类:①需要或可以转换为结构化数据的,如通过爬虫技术把网页的半结构化数据转换成结构化的形式存储或输出;②不需要或不能转换为结构化数据的,如文本的内容分析。

1. 网络爬虫应用实例

1) 网页数据的组织形式

网页是用超文本标记语言(HyperText Markup Language,HTML)编写的文本文件,超文本的意义在于,HTML 允许在文本中嵌入一些标签(Tag),以指示浏览器如何对文本进行操作。在 HTML 中定义了若干元素,用于表示文档结构、规定信息格式、指定操作功能等。HTML 元素一般由起始标签和结束标签组成,它们都必须用一对尖括号 < > 括起来。

为了能够从网页源代码中爬取数据,需要分析数据所在标签的特征,以便 Python 调用相关的模块库进行处理。

2) 利用 requests 库爬取网页

浏览器端访问网页实际上是向服务器端发起一个请求(Request),请求访问服务器上的某个 HTML 网页文件;服务器接收到请求后将会产生一个响应(Response),将这个网页文件的内容返回给浏览器端。

requests 是 Python 的一个第三方库,提供了使用程序请求访问网页的功能,简单易用。调用一次 requests 库的 get() 函数就相当于向服务器发起了一次请求,服务器会返回一个响应,requests 库会把该响应的信息封装到一个 Response 响应对象中。

Response 对象的 text 属性和 content 属性都表示响应的内容,具体就是网页的全部

内容。两者的主要区别: text 属性是以字符串的形式展现, content 属性则以二进制形式展现。当请求的是普通的 HTML 格式的网页时, 通常使用 text 属性; 如果请求的是图片、视频或其他非文本格式的网页文件, 需使用 content 属性。

3) 利用 BeautifulSoup4 库解析网页文档

通过 requests 库爬取了网页的内容后, 若从网页的 HTML 源代码中抽取所需的数据, 则需要使用另外一个第三方库 BeautifulSoup4。它能够根据 HTML 语法建立解析树, 进而高效解析其中的内容。

解析 HTML 文档树有两种方法: 一是遍历文档树, 依次访问文档树上的节点; 二是搜索文档树, 基于目标数据所在标签的特征直接进行查找, 能够快速定位到目标数据所在标签附近, 然后再利用局部的上下级节点关系, 对局部结构进行标签遍历从而获取目标数据, 效率显然更高。

2. 中文文本分析实例

文本分析是对文本的表示及其特征项的提取, 它把从文本中抽取出的特征词进行量化来表示文本信息, 目的是从文本数据中提取出符合需要的、感兴趣的和隐藏的信息。

文本内容是非结构化的数据, 要从大量的文本中提取出有用的信息, 需要将文本从无结构的原始状态转换为便于计算机处理的数据。典型的文本分析过程主要包括分词、特征提取、数据分析、结果呈现等。

中文分词是将连续的字序列按照一定的规范重新组合成词序列的过程, 即将一个汉字序列切分成一个一个单独的词。Python 中文分词模块 jieba 采用的是基于词典的分词方法, 也称基于字符匹配的分词方法, 即在分析句子时与词典中的词语进行对比, 词典中出现的就划分为词。

词云是目前常用的关键词可视化形式, 它能直接抽取文本中的关键词, 并将其按照一定顺序和规律整齐美观地呈现在屏幕上。关键词是从文本的文字描述中提取的语义单元, 可反映出文本内容的重点。用词云可视化文本数据可以帮助人们快速地了解文本的内容和特征等信息。词云通常使用字体的大小和颜色表示关键词的重要程度或出现频次。wordcloud 是 Python 中非常优秀的词云展示第三方库, 以词为基本单位, 对文本中出现频率较高的关键词予以视觉化的展示。

5.2 基本练习题与解析

5.2.1 单选题

1. 以下关于 Python 变量的描述, 错误的是()。
- A. 变量的类型是可变的
 - B. 变量是动态定义的
 - C. 变量一经定义, 就不能消除
 - D. 变量名应符合标识符定义

【答案】 C

【解析】 Python 语言是动态类型语言, 程序中的变量是通过赋值而定义的, 在赋值

过程中数据对象的身份指派给变量,定义了该变量,该变量也有了类型。变量通过变量名而具体化,变量名属于标识符。在程序中可以使用 del 命令清除变量。

2. ()在 Python 中是非法语句。

A. $x += y$

B. $x = y = z = 1$

C. $x = (y = z + 1)$

D. $x, y = y, x$

【答案】 C

【解析】

(1) $x += y$ 属于复合赋值结构,等同于 $x = x + y$ 。

(2) $x = y = z = 1$ 属于链式赋值结构,同时给 x, y, z 赋值 1。

(3) $x = (y = z + 1)$ 是非法语句。其中,()中只能是表达式,而不能是赋值语句。

(4) $x, y = y, x$ 属于解包赋值结构,右侧的元组解包后按位置赋值给左侧的变量。

3. Python 不支持的数据类型是()。

A. list

B. char

C. int

D. float

【答案】 B

【解析】 Python 内置支持的基本数据类型有 int、float、str,组合数据类型有 list、tuple、set、dict。

4. `print(12 + 34 * 5 % 6)` 输出()。

A. 8

B. 14

C. 13

D. 15

【答案】 B

【解析】 根据运算符的优先规则。

在表达式 $12 + 34 * 5 \% 6$ 中优先运算的表达式是乘法运算式: $34 * 5$ 得 170。

在表达式 $12 + 170 \% 6$ 中优先运算的表达式是求余运算式: $170 \% 6$ 得 2。

在表达式 $12 + 2$ 中优先运算的表达式是加法运算式: $12 + 2$ 得 14。

print 函数的输出为 14。

5. Python 中优先级最高的运算符为()。

A. /

B. *

C. **

D. //

【答案】 C

【解析】 根据 Python 运算符的优先规则,4 个选项中,乘方运算符**的优先级最高,/、//、* 的优先级相同,按照左运算优先原则计算表达式。

6. 关于 a or b 的运算,描述错误的是()。

A. 若 $a == \text{True}$ $b == \text{True}$,则 $a \text{ or } b == \text{True}$

B. 若 $a == \text{False}$ $b == \text{False}$,则 $a \text{ or } b == \text{False}$

C. 若 $a == \text{True}$ $b == \text{False}$,则 $a \text{ or } b == \text{True}$

D. 若 $a == \text{False}$ $b == \text{True}$,则 $a \text{ or } b == \text{False}$

【答案】 D

【解析】 a or b 是逻辑或运算, a 和 b 中只要有一个是 True,则 a or b 的运算结果即为 True。

7. Python 中可以使用()函数接收用户输入的数据。