Python 科学计算基础

第三章 分支和迭代

2025 年 9 月 5 日

目录

if 语句和 if-else 表达式

分支的语义是根据若干条件是否满足从多个分支中选择一个运行，由 if 语句和 if-else 表达式实现。

if 语句可以有多种形式，以计算整数的绝对值为例说明。根据绝对值的定义，可以有以下三种实现方式：

- 单分支 if 语句：程序 3.1
- 两分支 if 语句：程序 3.2
- if-else 表达式：程序 3.3

多分支 if 语句

if 语句包含的每个分支可以是一条语句，也可以是多条语句组成的语句块。这些分支相对 if 所在行必须有四个空格的缩进。

程序 3.4 利用多分支 if 语句将百分制成绩转换为等级分。

for 语句

迭代的语义是当某一条件满足时反复运行一个语句块，由 for 语句、while 语句和推导式实现。

for 语句包含循环变量、可迭代对象和一个语句块。语句块相对 for 所在行必须有四个空格的缩进。第二章介绍的所有序列类型 (包括 list、tuple、range 和 str 等) 和 set、dict 等类型的对象称为可迭代对象 (iterable)，即可以通过 for 语句访问其包含的所有元素，在每次迭代时循环变量的值等于一个元素。

for 语句

end='x'表示输出以x结尾，默认是换行符

- 程序 3.5 输出 1 到 10 之间的所有自然数的和
- 程序 3.6 输出 100 到 120 之间的所有偶数
- 程序 3.7 输出一个由整数组成的集合中所包含的 3 的倍数
- 程序 3.8 输出一个由整数组成的集合中所包含的 3 的倍数，continue 语句跳过本次循环的剩余语句并开始下一次循环
- 程序 3.9 采用两种方式输出一个通讯录中的每个联系人的姓名和其电话号码
- 程序 3.10 输出一个字符串中的所有字符和其对应的 Unicode 编码值
- 程序 3.11 输出一个列表中的所有元素的最大值和最小值

while 语句

for 语句常用于循环次数已知的情形，而 while 语句也适用于循环次数未知的情形。

while 语句包含一个条件表达式和一个语句块。语句块相对 while 所在行必须有四个空格的缩进。while 语句的运行过程如下：

1. 对条件表达式求值。
2. 若值为 False，则 while 语句运行结束。
3. 若值为 True，则运行语句块，然后跳转到 1。

while 语句

- 程序 3.12 输出 1 到 10 之间的所有自然数的和
- 程序 3.13 输出 100 到 120 之间的所有偶数
- 程序 3.14 输出两个正整数的最大公约数

推导式

list、dict 和 set 等容器类型都提供了一种称为推导式 (comprehension) 的紧凑语法，可以通过迭代从已有容器创建新的容器。

程序 3.15

穷举

穷举 (exhaustive search) 是一种解决问题的基本方法。穷举法的基本思想是：当问题的解属于一个规模较小的有限集合时，可以通过逐一列举和检查集合中的所有元素找到解。

求解 [100,200] 的所有质数

100 到 200 之间的所有质数都是自然数。解决本问题的方法是：列举 100 到 200 之间的所有自然数，逐一检查每个自然数是否是质数。

本问题比我们之前所解决的问题更加复杂。当问题比较复杂时，在编写程序之前应提出一个设计方案，这样便于对解决问题的策略和步骤进行深入而细致的思考，避免错误。此外，还可以在保证正确性的前提下选择最优解决方案，提高程序的运行效率并降低资源使用量。

求解 [100,200] 的所有质数

本问题的设计方案如下：

1. 列举给定区间内的每个自然数 i。
   1.1 对于 i，判断其是否质数。对于每个从 2 到 $i-1$ 的自然数 j：
      1.1.1 检查 i 是否可以被 j 整除。
   1.2 若存在这样的 j，则 i 非质数。否则 i 为质数，输出 i。

求解 [100,200] 的所有质数

根据质数的定义，这个设计方案是正确的，而且每个步骤都易于实现，但是在运行效率上还有改进的余地。

1. 在步骤 1 列举自然数时，只需列出奇数。
2. 在步骤 1.1 查找 i 的因子 j 时，j 的取值范围的上界可以缩小为 $\lceil\sqrt{i}\rceil$。

基于以上改进的设计方案，可以使用嵌套 for 语句写出程序 3.16。

求解 3-sum 问题

3-sum 问题的描述如下：给定一个整数 x 和一个由整数构成的集合 S，从 S 中找一个由三个元素构成的子集，该子集中的三个元素之和必须等于 x。使用穷举法列举 S 的所有由三个元素构成的子集，逐个检查其是否满足条件。

程序 3.17 实现了穷举法求解 3-sum 问题，其中列表 S 表示 S。三重循环的循环变量 i,j 和 k 依次表示组成的子集的三个元素的索引值，它们满足严格递增关系。

求解 subset-sum 问题

subset-sum 问题的描述如下：给定一个整数 x 和一个由整数构成的集合 S，从 S 中找一个子集，该子集中的所有元素之和必须等于 x。使用穷举法列举 S 的所有子集，逐个检查其是否满足条件。设 S 包含 n 个元素，则 S 的每个子集 T 和 n 位二进制数存在一一映射。n 位二进制数的第 k 位为 1 表示第 k 个元素在子集 T 中，为 0 则表示不在。

例如设 $S = \{1, 2, 3, 4\}$，10 的二进制形式是 1010，其对应的子集是 $S = \{1, 3\}$。

程序 3.18 实现了穷举法求解 subset-sum 问题。

实验 3：分支和迭代

本实验的目的是掌握分支和迭代的语句。

在 Blackboard 系统提交一个文本文件 (txt 后缀)，文件中记录每道题的源程序和运行结果。

1. 考拉兹猜想

定义一个从给定正整数 n 构建一个整数序列的过程如下。开始时序列只包含 n。如果序列的最后一个数 m 不为 1 则根据 m 的奇偶性向序列追加一个数。如果 m 是偶数，则追加 $m/2$，否则追加 $3 \times m + 1$。考拉兹猜想 (Collatz conjecture) 认为从任意正整数构建的序列都会以 1 终止。

编写程序读取用户输入的正整数 n，然后在 while 循环中输出一个以 1 终止的整数序列。输出的序列显示在一行，相邻的数之间用空格分隔。例如用户输入 17 得到的输出序列是“17 52 26 13 40 20 10 5 16 8 4 2 1”。

2. 字符串加密

编写程序实现基于偏移量的字符串加密。加密的过程是对原字符串中的每个字符对应的 Unicode 值加上一个偏移量，然后将得到的 Unicode 值映射到该字符对应的加密字符。

用户输入一个不小于-15 的非零整数和一个由大小写字母或数字组成的字符串，程序生成并输出加密得到的字符串。例如用户输入 10 和字符串“Attack at 1600”得到的加密字符串是“K~~kmu*k~*;@::”。

3. 推导式转换为 for 语句

将程序 3.15 中的所有推导式转换为 for 语句。