

第四讲 - Python 流程控制

张建章

阿里巴巴商学院

杭州师范大学

2022-09



- 1 三种基本控制结构
- 2 选择结构
- 3 循环
- 4 break,continue,pass, 嵌套
- 5 循环结构与 else 子句搭配
- 6 课堂实训
- 7 课后练习

- 顺序结构：按照书写顺序依次解释执行；
- 选择结构：按照判断条件选择其中一个分支执行；
- 循环结构：满足指定条件时重复执行某些操作。

```
# 顺序结构，先接收键盘输入，再打印输出
```

```
age = input("Please input your age: ")  
print("Your age is {}".format(age))
```

```
# 选择结构，年龄小于18，禁止购买香烟
```

```
if age < 18:  
    print("Sorry, we can not sell cigarettes to you!")  
else:  
    print("Please input your ID number.")
```

```
# 循环结构，如身份证号不是18位数字，则重新输入
```

```
while True:  
    id_num = input("ID: ")  
    if id_num.isdigit() and len(id_num) == 18:  
        print('正在与公安系统联网核验.....，请稍后')  
        break
```

单选结构

```
# 下面为伪代码，不可直接运行
if condition:
    statements1

other statements
```

当条件判断表达式 (condition) 返回结果为 **True** 时，执行语句块 1 (statements1)，否则 (**False**)，直接顺序往下运行与 **if** 对齐的后续代码，在上例中，如果 condition 计算结果返回 **False**，则直接执行后续代码 (other statements)。

注意：① 判断条件 (condition) 必须是一个能返回布尔值 (或等价于布尔值，如 0,1,None) 的表达式，如，关系运算和逻辑运算表达式；② 语句块 1 (statements1) 可以包含任意多行代码。

```
# 互换两个变量的值，按照从小到大的顺序输出两个值
a = int(input("Please input the first integer: "))
b = int(input("Please input the second integer: "))
print('Before exchange:', a, b)
if a > b: # 条件
    a, b = b, a # 语句块
print('After exchange:', a, b) # 单选结构外的语句

# 寻找180+的男孩子
height = input("Please input your height(cm): ")
if float(height) < 180:
    print("Sorry, you are a good boy, but you know ...")
```

双选结构

```
# 下面为伪代码，不可直接运行
if condition:
    statements1
else:
    statements2

other statements
```

当条件判断表达式 (condition) 返回结果为 **True** 时，执行语句块 1 (statements1)，否则 (**False**) 执行语句块 2 (statements)，最后再顺序执行后续的语句块 (other statements)。

```
# 根据身份证号判断性别，第15-17位数字组成的三位数如果是奇数则为男性
last_four = input("Please input the last four number of your ID:
↪ ")

num = int(last_four[:3])
if num % 2:
    print('Male')
else:
    print('Female')
```

上述条件 `num % 2` 返回 0 或者 1，等价于 `False` 或 `True`。

多选结构

```
# 下面为伪代码，不可直接运行
if condition1:
    statements1
elif condition2:
    statements2
... ..
elif conditionN:
    statementsN
...
else:
    statements0

other statements
```

满足条件 1 (condition1) 执行语句块 1 (statements1)，满足条件 2 (condition2) 执行语句块 2 (statements2)，.....，所有条件都不满足，则直接执行后续代码 (other statements)。


```
# 根据分数输出相应评语
score = float(input("Please input your final score: "))

if 90 <= score < 100:
    print('Great!')
elif 80 <= score < 90:
    print('Good!')
elif 70 <= score < 80:
    print('You could be better!')
elif 60 <= score < 70:
    print('Dangerous!')
elif 0<= score < 60:
    print("It's a pity!")
else:
    print('The score is invalid!')
```

注意：使用多选结构时，要根据实际问题，全面考虑可能的条件判断，如，上例中，百分制下，合法的分数应该在 0 到 100 之间。

选择结构嵌套：条件满足时 (`True`) 所执行的语句块 (statements) 也是一个选择结构。

```
amount = int(input("输入驾驶员每100ml血液酒精的含量(mg): "))
if amount < 20:
    print("驾驶员不构成酒驾")
else:
    if amount < 80:
        print("驾驶员已构成酒驾")
    else:
        print("驾驶员已构成醉驾")
```

应用海伦公式和余弦定理的例子

问题: 给定三条边的长度, 首先判断这三条边是否能够成一个三角形, 如果能, 请判断其构成的是哪种类型的三角形 (锐角、直角、钝角)。

海伦公式: $S = \sqrt{p(p-a)(p-b)(p-c)}$, 其中 $p = \frac{a+b+c}{2}$

余弦定理: $\cos A = \frac{b^2+c^2-a^2}{2bc}$

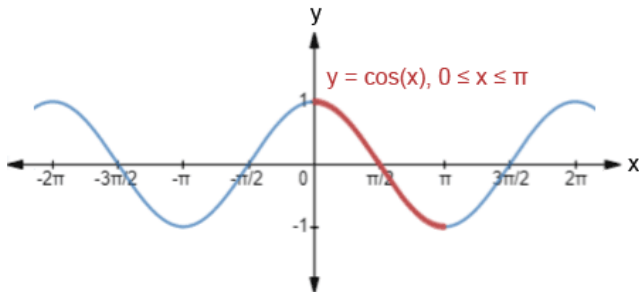


图 1: cosine 函数图像

应用海伦公式和余弦定理的例子 (续)

```
# 将三条边按照长度，从小到大排列
a = 3; b = 12; c = 10
a,b,c = sorted([a,b,c])
# 任意两边之和大于第三边 ( $a < b < c$ )，因此构成三角形
if (a + b) > c:
    # 下面用海伦公式用三边求三角形面积
    s = (a + b + c)/2
    area = float((s*(s-a)*(s-b)*(s-c))**(1/2))
    print("The area of triangle is {:.2f}".format(area))
# 根据“同一个三角形内，长边对大角”，以及余弦定理，计算最大角的余弦值
cos_C = (a**2 + b**2 - c**2)/(2*a*b)
if cos_C == 0:
    print("Right triangle") # 直角
elif cos_C < 0:
    print("Obtuse triangle") # 钝角
else:
    print("Acute triangle") # 锐角
else:
    print("Invalid lengths")
```

课堂练习: 个人所得税计算

题目: 键盘接收年收入 (元为单位), 根据下图税率, 使用选择结构计算应缴所得税额。

级数	全年应纳税所得额	税率(%)
1	不超过 36000 元的	3
2	超过 36000 元至 144000 元的部分	10
3	超过 144000 元至 300000 元的部分	20
4	超过 300000 元至 420000 元的部分	25
5	超过 420000 元至 660000 元的部分	30
6	超过 660000 元至 960000 元的部分	35
7	超过 960000 元的部分	45

图 2: 个人所得税税率

while 循环

```
# 下面为伪代码，不可直接运行
while condition:
    statements

other statements
```

当条件判断表达式 (condition) 返回结果为 **True** 时，执行语句块 (statements)，再次计算 condition，如果结果返回 **True**，则继续执行 statements，循环往复...，否则 (**False**)，跳出循环，顺序往下运行与 **while** 对齐的后续代码 (other statements)。

注意：① 判断条件 (condition) 必须是一个能返回布尔值 (或等价于布尔值，如 0,1,None) 的表达式，如，关系运算和逻辑运算表达式；② 语句块 (statements) 可以包含任意多行代码；③ 选择结构只计算一次 condition，**while** 循环计算多次 condition。

```
# 笨办法求1+2+3+ ...+100的和
sum = 0
number = 1
while number < 101:
    sum += number
    number += 1
print(sum)

# 输出小于10的奇数
a = 1
while a < 10:
    print(a)
    a += 2
print('我就是与while对齐的后续代码^_^')
```

注意: 上述两例中, `while` 循环中的 `statements` 中都对 `condition` 中的变量进行了修改, 才使得 `condition` 可以返回 `False`, 程序跳出 `while` 循环, 从而执行后续代码 (other statements)。

for 循环

```
# 下面为伪代码，不可直接运行
for item in iterables:
    statements
```

通过遍历可迭代对象 (iterables) 中的元素 (item)，来控制语句块 (statements) 执行的次数。典型的可迭代对象有序列 (列表，元组，字符串)、集合、字典、range、zip、enumerate 等。

```
for char in '中华民族伟大复兴':
    print(char)
for char in list('中华民族伟大复兴'):
    print(char)
for char in tuple('中华民族伟大复兴'):
    print(char)
for char in set('中华民族伟大复兴'):
    print(char)
```


内置函数 range 和 enumerate

`range` 和 `enumerate` 是两个经常与 `for` 循环搭配使用的内置函数，典型用法如下例：

```
# 打印出1-9范围内的数字
for i in range(1,10):
    print(i)
# 以3为步长打印出1-9范围内的数字
for i in range(1,10,3):
    print(i)
```

`range(start, end, step)` 返回一个 `range` 对象，使用内置函数 `list` 可将该 `range` 对象转化为列表，列表会包含从 `start` 到 `end-1` 的整数，步长为 `step`。此处的 `start`、`end` 和 `step` 与列表切片所使用的参数非常像，都是有头没有尾、带步长就跳。

```
for idx,num in enumerate(range(48,59)):  
    print(idx,num)
```

内置函数 `enumerate` 接收一个可迭代对象为参数，返回一个 `enumerate` 对象，通过 `list` 函数可将该对象转化为列表，列表中包含的元素均为形如 `(index,element)` 的二元组，`element` 表示可迭代对象中的一个元素，`index` 为该元素在可迭代对象中的索引 (`index` 本质上是从 0 开始的计数器，看下面例子)。

```
for idx, num in enumerate(set([1,2,3])):  
    print(idx,num)
```

集合无序，上例中 `idx` 不表示索引，表示 `num` 是第几个被取出来的。

注意: 在使用 `range` 和 `enumerate` 时，不是必须要在其外层套一个 `list` 函数，套 `list` 函数便于查看他们的内容。

课堂练习

题目 1: 计算 1 1000 范围内 3 的倍数和 7 的倍数的数字之和。

题目 2: 水仙花数是一个 3 位数，它的每个位上的数字的 3 次幂之和等于它本身 (如: $1^3 + 5^3 + 3^3 = 153$)，求所有水仙花数。

`break`、`continue`、`pass` 都是 Python 的保留关键字，用法如下：

- `break`：只能出现在循环结构 (for 循环, while 循环) 中的语句块 (statements) 中，程序一旦运行到 `break`，立刻跳出循环，继续执行后续语句 (other statements)；
- `continue`：只能出现在循环结构 (for 循环, while 循环) 中的语句块 (statements) 中，程序一旦运行到 `break`，立刻结束本次循环，进入下次循环；
- `pass`：可以单独出现在任何一行，主要作用是保证代码格式的规范和完整性，程序一旦遇到 `pass`，什么操作也不执行。

注意：当 `break` 和 `continue` 出现在嵌套循环中时，只作用于它所在的那一层循环。

```
vowel = 'aeiou'

# 打印输出 br
for char in 'brown':
    if char in vowel:
        break
    print(char)

# 打印输出 brwn
for char in 'brown':
    if char in vowel:
        continue
    print(char)

# 打印输出 brwn
for char in 'brown':
    if char in vowel:
        pass
    else:
        print(char)
```

题目: 对英文句子去做去元音化操作 (Disemvoweling), 即, 将单词中非首尾元音字母去除, 这是短信 (SMS) 语言的典型特征, 阅读起来需要的认知努力很小。

```
vowel = 'aeiou'
sent = 'The quick brown fox jumps over the lazy dog'
word_list = []

for word in sent.strip().split():
    char_list = []
    for idx,char in enumerate(word):
        if char in vowel and idx not in [0, len(word) -1]:
            pass
        else:
            char_list.append(char)
    word_list.append(''.join(char_list))
print(' '.join(word_list))
# The qck brwn fx jmps ovr the lzy dg
```

```
# 下面为伪代码，不可直接运行
while condition:
    statements1
else:
    statements2
other statements
```

```
# 下面为伪代码，不可直接运行
for item in iterables:
    statements1
else:
    statements2
other statements
```

在循环未经 `break` 打断的情况下，整个循环完成时执行 `else` 子句下的语句块 (`statements2`)，再执行后续代码 (`other statements`)。

题目: 找出 [2, 100] 中的素数。素数 (质数), 指在大于 1 的自然数中, 除了 1 和该数自身外, 无法被其他自然数整除的数。

```
for i in range(2, 100):  
    for j in range(2, i):  
        if i % j == 0:  
            break  
    else:  
        print(i, ' 是素数')
```


题目：寻找整数 `x` 的最大因子 (除 1 和自身外)。

```
x = input('Please input a integer bigger than 1: ')

for num in range(x-1,1,-1):
    if x % num == 0:
        print('The biggest factor is {}'.format(num))
        break
else:
    print('{} is a prime number'.format(x))
```

题目: 求两个数的最大公约数。

```
a = int(input("Please input the first number: "))
b = int(input("Please input the second number: "))
for i in range(min(a,b), 1,-1):
    if a%i == 0 and b%i == 0:
        print("{}和{}的最大公约数是{}".format(a,b,i))
        break
else:
    print("{}和{}是互素数".format(a,b))
```

题目：最多三次密码输入机会。

```
PASSWORD="123456"
max_times = 3
flag = False
for i in range(3):
    pwd = input("Please input the password: ")
    if pwd == PASSWORD:
        flag = True
        break
    else:
        print("Come on baby, try again, {} times
        ↪ left.".format(max_times - i - 1))
if flag:
    print("Great, Baby!")
else:
    print("Sorry, Baby, See you tomorrow!")
```

题目: 1-100 之间不是 7 的倍数的数字之和。

```
sum = 0
for i in range(1, 101):
    if i%7 == 0:
        continue
    else:
        sum += i
print(sum)
```

题目: 思考下面代码的运行结果。

```
i = 0
sum = 0
while i < 5:
    i += 1
    if i == 3:
        continue
    sum += i
print(sum)
```

题目: 输出字符串中的每个字符, h 除外。

```
for letter in 'Python':  
    if letter == 'h':  
        pass  
    else:  
        print('Current letter is {}'.format(letter))
```

题目: 输出 100 之内的所有素数, 每行 10 个依次输出。

```
tmp_list = []
for i in range(2, 100):
    for j in range(2, i):
        if i % j == 0:
            break
    else:
        tmp_list.append(str(i))
    if len(tmp_list) == 10:
        print(', '.join(tmp_list))
        tmp_list.clear()
```

题目：输出如下图所示的九九乘法表。

```

1x1=1
1x2=2   2x2=4
1x3=3   2x3=6   3x3=9
1x4=4   2x4=8   3x4=12  4x4=16
1x5=5   2x5=10  3x5=15  4x5=20  5x5=25
1x6=6   2x6=12  3x6=18  4x6=24  5x6=30  6x6=36
1x7=7   2x7=14  3x7=21  4x7=28  5x7=35  6x7=42  7x7=49
1x8=8   2x8=16  3x8=24  4x8=32  5x8=40  6x8=48  7x8=56  8x8=64
1x9=9   2x9=18  3x9=27  4x9=36  5x9=45  6x9=54  7x9=63  8x9=72  9x9=81
  
```

```

for i in range(1, 10):
    for j in range(1, i+1):
        print('{x}{y}={z}\t'.format(j, i, i*j), end='')
    print()
  
```

题目: 使用 * 作为填充物打印输出高度为 10 的等腰三角形。

```

      *
     ***
    *****
   ********
  **********
 **********
 **********
 **********
 **********
 **********

```

```
row = 10
for i in range(row):
    for _ in range(row - i - 1):
        print(' ', end='')
    for _ in range(2 * i + 1):
        print('*', end='')
    print()
```


题目：用蒙特卡罗法计算圆周率 π 的值。

题目：用微元法计算定积分 $\int_0^1 x^2$ 的值。

答案参见课程代码。

THE END