

# Python 语言程序设计

# 第四章

## 程序控制结构



4.1 结构化程序的基本结构

01

02

4.2 分支结构

4.3 循环结构

03

04

4.4 break语句和continue语句



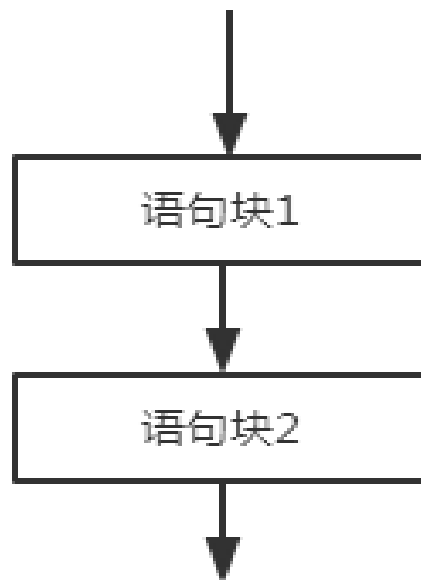
# 4.1 结构化程序的基本结构

# 结构化程序设计的三种结构

1. 顺序结构
2. 分支结构（又称选择结构）
3. 循环结构

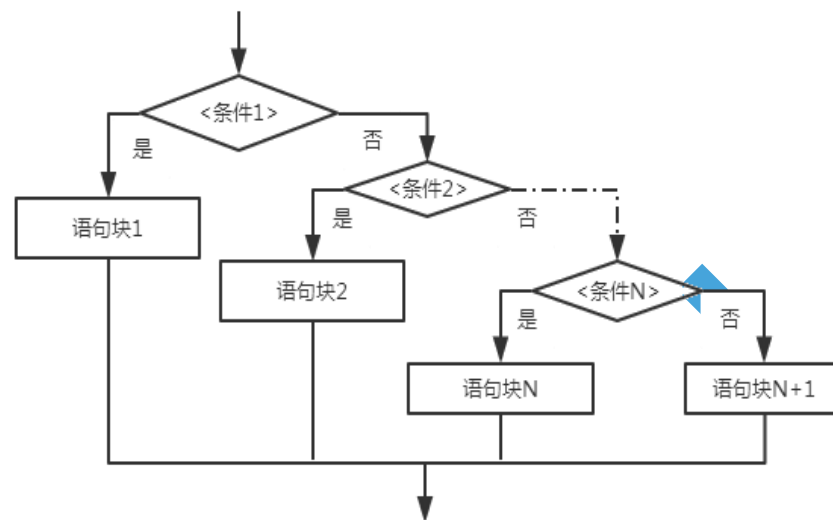
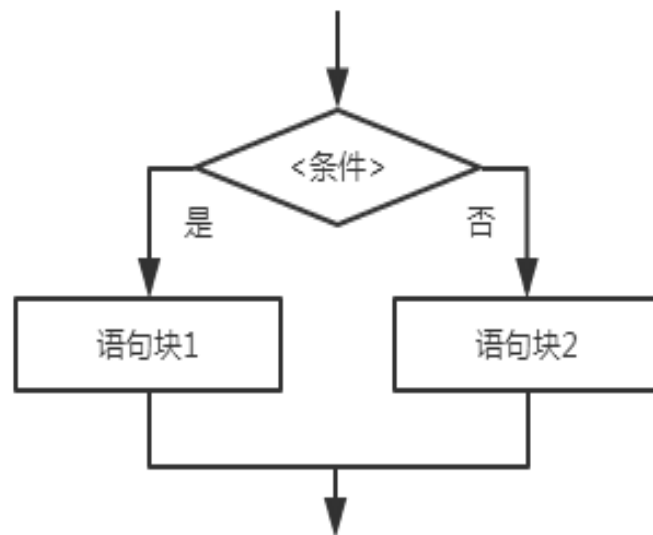
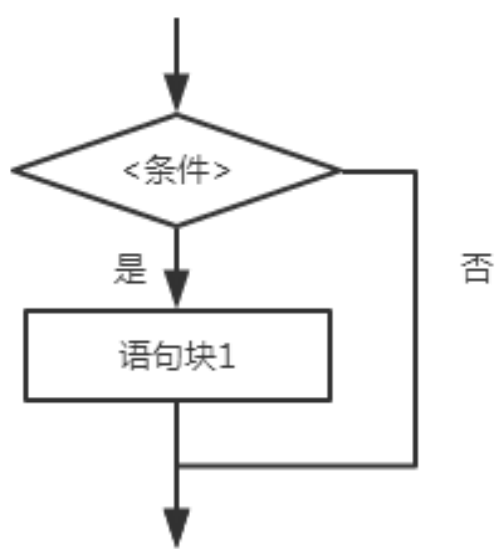
## 4.1.1 顺序结构

- 顺序结构是指程序从第一行语句开始执行，执行到最后一行语句结束，程序中的每条语句都会被执行一次。



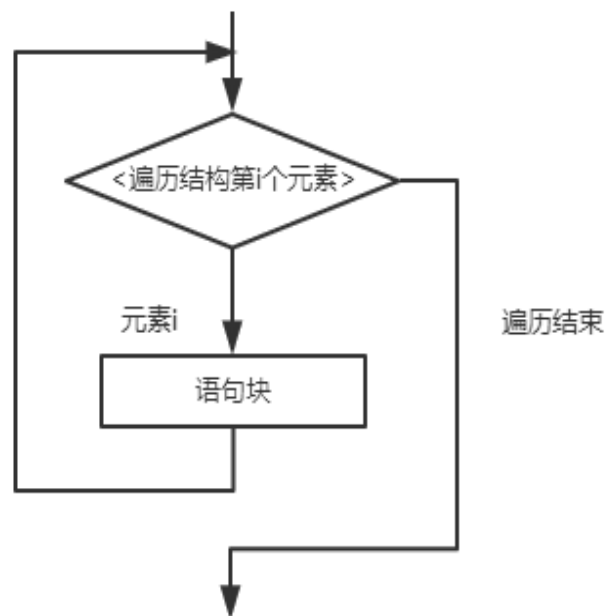
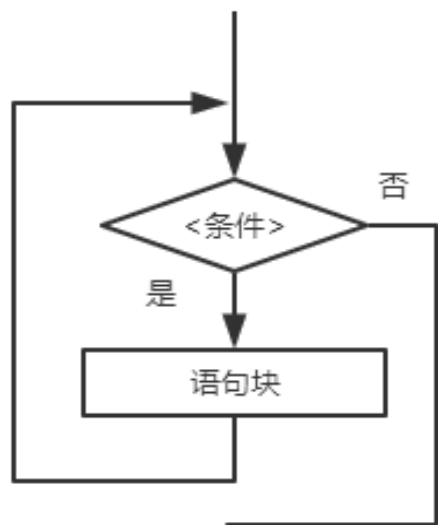
## 4.1.2 分支结构

- 分支结构，也称选择结构，表示程序的处理步骤出现了分支，它需要根据某一特定的条件选择其中的一个分支执行。根据分支的多少，分为**单分支结构**、**双分支结构**、**多分支结构**。



### 4.1.3 循环结构

- 循环结构是程序根据条件判断，在满足条件的情况下反复执行某个语句块的运行方式。Python中根据循环触发条件不同，分为**条件循环**和**遍历循环**。



## 4.2 分支结构



## 4.2.1 单分支结构

- 分支结构一般用来判断某种情况发生或者不发生。单分支if语句的语法格式如下：

if <条件>:  
    <语句块>

- 程序执行到if语句时，如果判断到条件为True，则执行语句块；条件为假，则跳过语句块。

例4.1 输入两个数字，输出其中较大数字。

```
x=eval(input("请输入第一个数字:"))
```

```
y=eval(input("请输入第二个数字:"))
```

```
print("输入的两个数字为：",x,y)
```

```
if x>y:
```

```
    print("较大的是:",x)
```

```
if x<y:
```

```
    print("较大的是:",y)
```

## 4.2 双分支结构

- 双分支结构是使用比较多的一种程序结构，一般用来在两种情况中选择一种执行。双分支if语句的语法格式如下：

```
if    <条件>:  
    <语句块1>  
else:  
    <语句块2>
```

- 当if后的条件为True时，执行语句块1，然后结束分支结构；当条件为False时，执行语句块2，然后结束分支结构。

## 例题4.2 用双分支改写例题4.1

```
x=eval(input("请输入第一个数字:"))
y=eval(input("请输入第二个数字:"))
print("输入的两个数字为: ",x,y)
if x>y:
    print("较大的是:",x)
else:
    print("较大的是:",y)
```

### 4.2.3 多分支结构

- 当判断的条件有多个，结果也有多种情况的时候，可用多分支if语句进行判断。多分支结构语法格式如右：
- 程序执行时，会按照条件n的序列从上往下进行判断，当第一个条件i的值为True，就执行该条件下的语句块，然后整个多分支if结构结束。如果没有任何条件为True,则执行else下的语句块。

```
if <条件1>:  
    <语句块1>  
elif <条件2>:  
    <语句块2>  
elif <条件3>:  
    <语句块3>  
...  
else:  
    <语句块n>
```

## 例4.3 用多分支改写例题4.1

```
x=eval(input("请输入第一个数字:"))
y=eval(input("请输入第二个数字:"))
print("输入的两个数字为: ",x,y)
if x>y:
    print("较大的是:",x)
elif x<y:
    print("较大的是:",y)
else:
    print("这两个数字相等")
```

## 例题4.4

- 输入一个分数，判断它应该得到的学分绩点。规则为：90分以上绩点为4；80~90分绩点为3；70~79分绩点为2；60~69分绩点为1；60以下绩点为0。以下两段代码哪一段满足上述规则。

```
score = eval(input("请输入分数: "))
if score >= 90:
    gpa = 4
elif score >= 80:
    gpa = 3
elif score >= 70:
    gpa = 2
elif score >= 60:
    gpa = 1
else:
    gpa = 0
print("应得学分绩点为: ", gpa)
```

```
score = eval(input("请输入分数: "))
if score < 60:
    gpa = 0
elif score >= 60:
    gpa = 1
elif score >= 70:
    gpa = 2
elif score >= 80:
    gpa = 3
else:
    gpa = 4
print("应得学分绩点为: ", gpa)
```

## 4.2.4 分支结构的嵌套

- 如果一个if分支结构中包含另一个（或多个）if分支，称为分支结构的嵌套。

## 例4.5输入3个数字，利用分支结构进行降序排列输出。

```
a=eval(input("输入第1个数字"))
b=eval(input("输入第2个数字"))
c=eval(input("输入第3个数字"))
print("输入顺序为：",a,b,c)
if a<b:
    a,b=b,a
if a<c:
    print("排序后为：",c,a,b)
else:
    if c>b:
        print("排序后为：",a,c,b)
    else:
        print("排序后为：",a,b,c)
```

## 4.3 循环结构



- 循环结构是根据条件去重复执行某些语句，它是程序设计中一种重要的结构。使用循环控制结构可以减少程序中大量重复的语句，从而编写出更简洁的程序。Python提供了两种不同风格的循环结构，包括遍历循环for语句循环和条件循环while语句循环。
- 一般情况下，for语句循环是按给定的次数进行循环，而while语句循环是在条件满足时执行循环。

### 4.3.1 for循环结构

- 遍历循环可以理解为让循环变量逐一使用一个遍历结构中的每个项目，遍历结构可以是字符串、列表、文件、range()函数等。for语句循环语法格式如下：

```
for <循环变量> in <遍历结构>:  
    <语句块>
```

## 例4.6 求 $1^2+2^2+\dots+10^2$ 之和

```
s=0
for i in range(1,11):
    s=s+i**2
print("数列和为",s)
```

for语句循环中使用else关键字，语法格式如下：

for <循环变量> in <遍历结构>:

    <语句块1>

else:

    <语句块2>

在这种结构中，当for循环正常执行之后，程序会继续执行else语句中的内容。如果for循环因为某种原因没有正常执行完，比如遇到了break语句，则不会执行else语句中的内容。所以通常用else来检验for循环是否正常结束。

例4.7求字符串 "Life is short, YOU need Python!"  
中有多少个字母 "o", 不区分大小写字母。

```
n=0
```

```
str="Life is short, YOU need Python!"
```

```
for i in str:
```

```
    if i=="o" or i=="O":
```

```
        n=n+1
```

```
else:
```

```
    print("计算完毕,字母o的个数为: ",n)
```

### 4.3.2 while语句循环

在明确知道循环的次数，或者明确遍历结构的时候，一般使用for循环。但是更多的时候无法明确遍历结构，或者不确定循环需要进行多少次。这时，就要使用while循环了。while语句循环语法格式如下：

```
while <条件>:  
    <语句块>
```

## 例4.8 猜价格

```
from random import randint
n=randint(10,100)
print("商品价格已经产生，请输入10到100间的价格。")
bingo=False
while bingo==False:
    guess=eval(input("请输入您猜的价格。"))
    if guess>n:
        print("您输入的价格高于指定价格，请继续。")
    elif guess<n:
        print("您输入的价格低于指定价格，请继续。")
    else:
        print("恭喜您猜对了！价格为",guess)
        bingo=True
else:
    print("游戏结束！")
```

## 例4.9 用while循环求 $1^2+2^2+\dots+10^2$ 之和

```
s=0
```

```
i=1
```

```
while i<=10:
```

```
    s=s+i**2
```

```
    i+=1
```

```
else:
```

```
    print("数列和为",s)
```

### 4.3.3 循环的嵌套

- 在循环语句中使用另一个循环语句称为循环的嵌套，也称多重循环。for语句循环和while语句循环都可以互相嵌套。例如：

```
for i in range(1,4):  
    for j in range(1,4):  
        print("i值为",i,";", "j值为",j)
```

上层的i循环称为外层循环，下层的j循环称为内层循环，当外层循环执行一次时，内层循环就要整体循环一遍。

## 例4.10

- 某班级一共有50人，需要组织1次竞赛活动，已知一等奖奖品40元1个、二等奖奖品20元1个、三等奖奖品10元1个，每个人都要有奖品并且一等奖最少、三等奖最多，班费共有1000元，该如何设置各奖级个数。

```
for x in range(25):
    for y in range(50):
        for z in range(50):
            if x<y<z and x*40+y*20+z*10==1000 and x+y+z==50:
                print("一等奖%d个；二等奖%d个；三等奖%d个"%(x,y,z))
else:
    print("计算完毕")
```

## 4.4 break语句和continue语句

## 4.4.1 break语句

- Python提供了一个提前结束循环的语句——break语句。在循环中，执行到break语句时，可以结束本层的循环。一般来说，break语句要放在一个分支结构中，当触发某个条件时，结束循环的运行。

## 例4.11求两个数字的最小公倍数

```
x=eval(input("输入第一个数字"))
y=eval(input("输入第二个数字"))
if x<y:
    x,y=y,x
for i in range(x,x*y+1):
    if i%x ==0 and i%y==0:
        print(x,"和",y,"的最小公倍数为",i)
        break
```

当使用了break时，循环变量i遍历到第一个满足if分支的数字，输出计算结果后循环就结束，程序只能输出一个数字，并且是最小公倍数。

## 4.4.2 continue语句

- continue语句和break语句一样，用在循环结构中，用来结束循环的运行，但是continue语句只能结束本次循环的执行，而不终止循环。即当执行到continue语句时，程序会终止当前循环，并忽略循环中continue之后的语句，然后回到循环语句for或者while，再次判断是否进行下一次循环。

例4.12输入10名同学的分数求及格同学的均值，如果分数低于60，则不计入计算中。

```
s,n=0,0
```

```
for i in range(1,11):
```

```
    score=eval(input("输入成绩"))
```

```
    if score>=60:
```

```
        s=s+score
```

```
        n=n+1
```

```
print("合格人数为:",n)
```

```
print("成绩平均值为:",round(s/n,2))
```

```
s,n=0,0
```

```
for i in range(1,11):
```

```
    score=eval(input("输入成绩"))
```

```
    if score<60:
```

```
        continue
```

```
    s=s+score
```

```
    n=n+1
```

```
print("合格人数为:",n)
```

```
print("成绩平均值为:",round(s/n,2))
```

谢谢观看