分支与循环

孙栩
xusun@pku.edu.cn

目录

contents

关系运算符、逻辑运算符和逻辑表达式

关系运算符和bool类型

六种关系运算符用于数值的比较

- 相等 ==
- 不等 !=
- 大于 >
- 小于 <
- 大于等于 >=
- 小于等于 <=

比较的结果是bool类型，成立则为True，反之为False

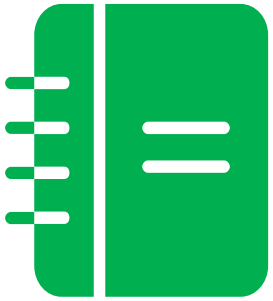
bool类型数据只有两种取值，True或False

关系运算符和bool类型

```
print(3 < 5)                    #>>True
print(4 != 7)                   #>>True
a = 4
print(2 < a < 6 < 8)            #>>True
print(2 < a == 4 < 6)           #>>True
print(2 < a > 5)                #>>False
b = a < 6
print(b)                        #>>True
print(b == 1)                   #>>True
print(b == 2)                   #>>False
b = a > 6
print( b == 0)                  #>>True
a = True
print(a == 1)                   #>>True
```

关系运算符和bool类型

关系运算符也能比较字符串(按字典序，大小写相关)



```
a = "k"
print(a == "k")              #>>True
a = "abc"
print(a == "abc")            #>>True
print(a == "Abc")            #>>False
print("abc" < "acd")         #>>True
print("abc" < "abcd")        #>>True
```

逻辑运算符和逻辑表达式

逻辑运算符有 and or not 三种，操作的结果是True或False

与 (and): 全部为真时，结果才为真; 只要有一个为假，结果就为假

逻辑运算符和逻辑表达式

逻辑运算符有 and or not 三种，操作的结果是True或False

- 与 (and): 全部为真时，结果才为真; 只要有一个为假，结果就为假

- 或 (or): 只要有一个为真，结果就为真；全部为假，结果才为假

逻辑运算符和逻辑表达式

逻辑运算符有 and or not 三种，操作的结果是True或False

与 (and): 全部为真时，结果才为真; 只要有一个为假，结果就为假

或 (or): 只要有一个为真，结果就为真；全部为假，结果才为假

非 (not): 真的结果是假，假的结果是真

逻辑运算符和逻辑表达式

与: exp1 and exp2

- exp1和exp2的值都为True时，结果为True；
- 只要exp1和exp2有一个为False，结果就是False

```
n = 4
(n > 4) and (n < 5)                          # False
(n >= 2) and (n < 5) and (n%2 == 0)          # True
```

什么相当于True或False

- 0，""(空字符串)，[](空表)都相当于False
- 非0的数，非空的字符串和非空列表，都相当于True
- True可以看作1，False可以看作0

```
True == 1          #True
False == 0         #True
```

逻辑运算符和逻辑表达式

或 ： exp1 or exp2

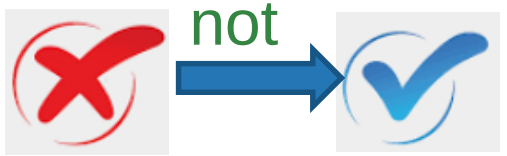
exp1和exp2的值都为False时，结果为False；只要exp1和exp2有一个为True，结果就是True

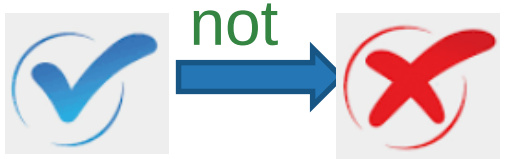
```
 n = 4
(n > 4) or (n < 5)      #True
(n <= 2) or (n > 5)     #False
```

逻辑运算符和逻辑表达式

非： not exp

exp值为**True**时, 结果为**False**
exp值为**False**时, 结果为**True**






```
not (4 < 5)    #False
not 5          #False
not 0          #True
not "abc"      #False
not ""         #True
not []         #True
not [1]        #False
```

各种运算符的优先级

从高到低：

- 算术运算符　`+ - * / // % **`
- 关系运算符　`< > == != <= >=`
- 逻辑运算符　`and or not`

记不得就勤用（ ）

```
print(3 + 2 < 5)      #>>False
print(3 + (2 < 5))    #>>4
```

条件分支语句（if 语句）

条件分支语句

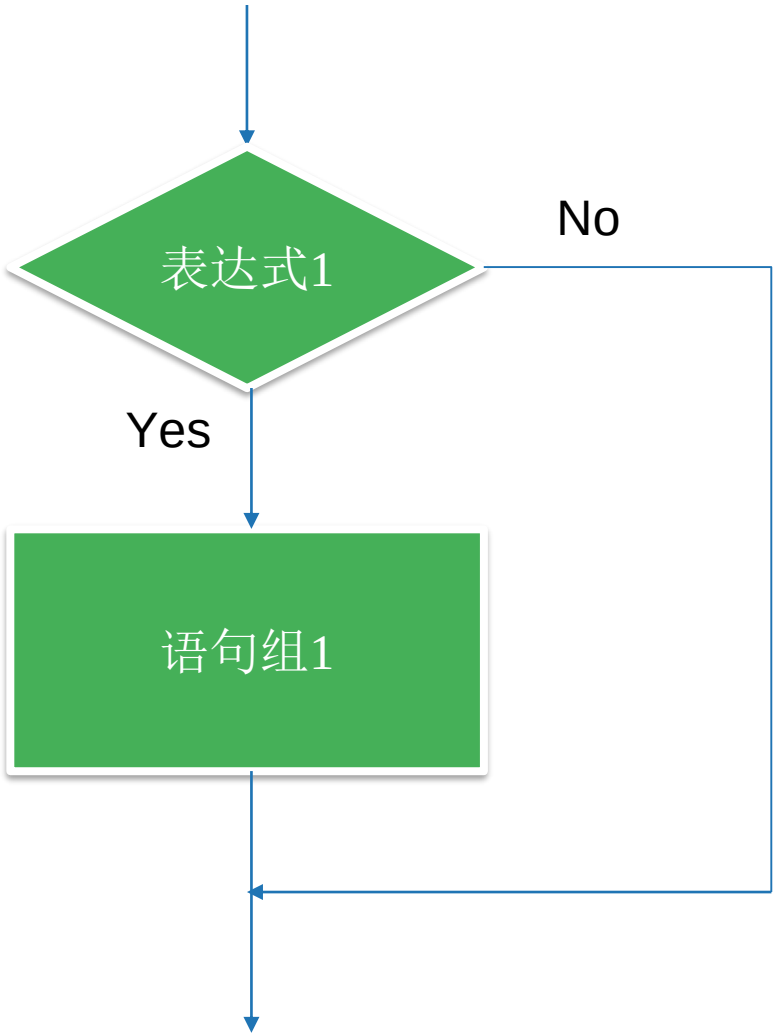
并非所有的程序语句都要被顺序执行到，会希望满足某种条件就执行这部分语句，满足另一条件就执行另一部分语句。这就需要“条件分支语句”

-

```
if 你是男生:
    你应该去男洗手间
if 你是女生:
    你应该去女洗手间
```

条件分支语句

```
if 表达式1:
    语句组1
```

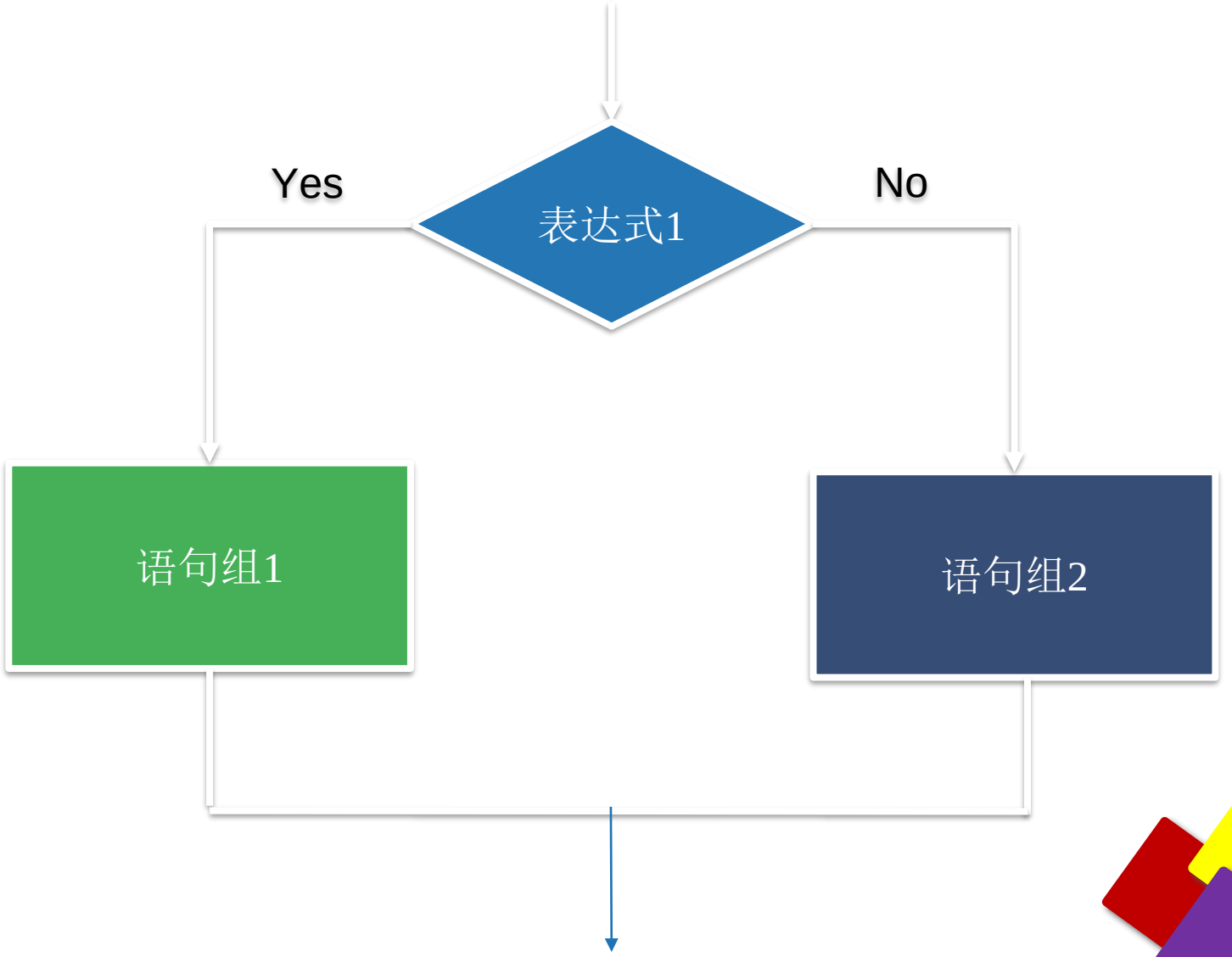


条件分支语句

```
In [93]: a = int(input("请输入一个整数:  "))
    ...: if a % 2 == 0:
    ...:     print("%d 是一个偶数" % a)
    ...:
    ...: print("所输入的数字是%d" % a)

请输入一个整数:  10
10 是一个偶数
所输入的数字是10

In [94]: a = int(input("请输入一个整数:  "))
    ...: if a % 2 == 0:
    ...:     print("%d 是一个偶数" % a)
    ...:
    ...: print("所输入的数字是%d" % a)

请输入一个整数:  5
所输入的数字是5
```

条件分支语句

```
if 表达式1:
    语句组1
else:
    语句组2
```

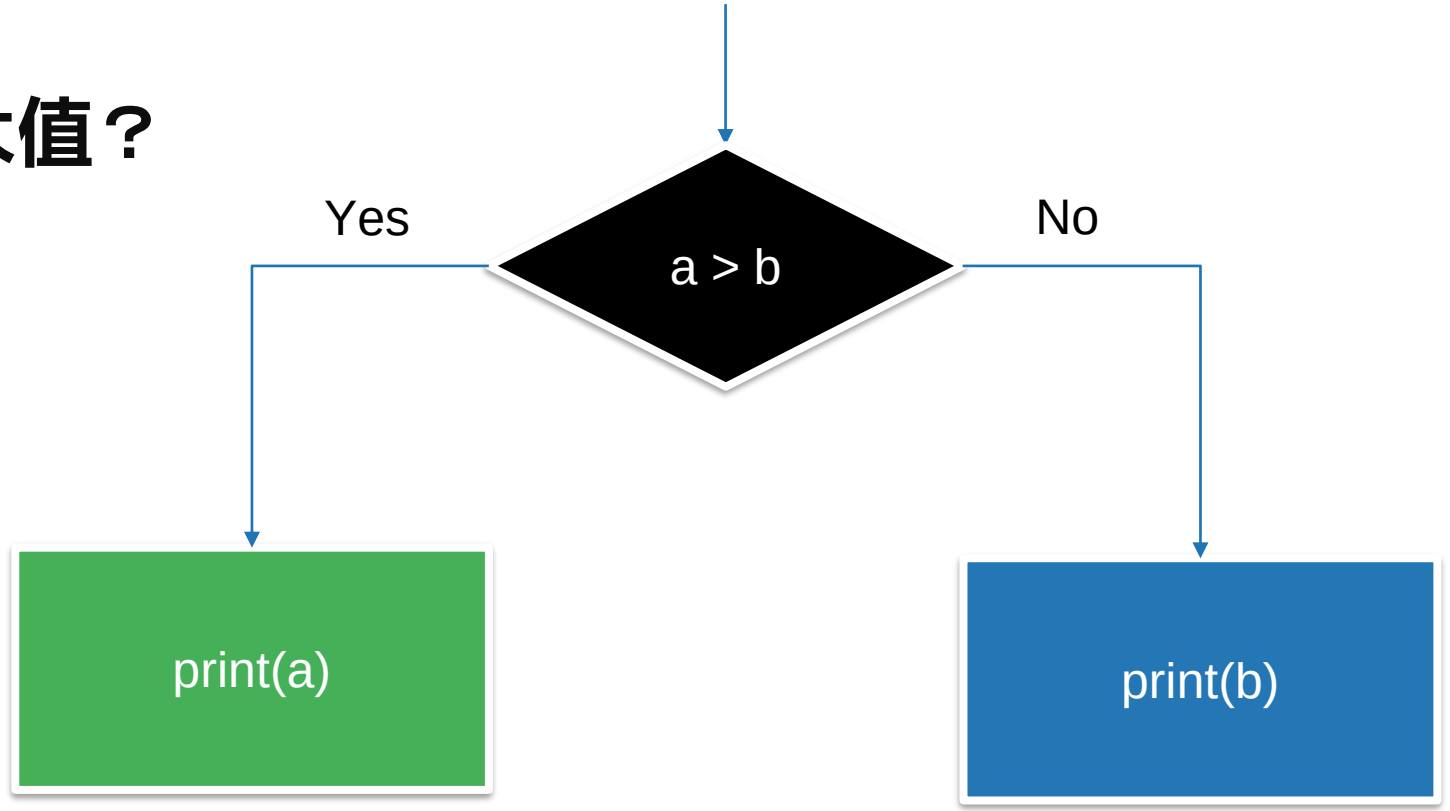


条件分支语句

如何打印出两个数的最大值？

```
a = int(input())
b = int(input())
   if a > b:
        print(a)
   else:
        print(b)
```

输入： 4, 3
输出：4



条件分支语句

```
if 逻辑表达式1:
        语句组1
elif 逻辑表达式2:
        语句组2
  ...  #可以有多种个 elif
elif 逻辑表达式n:
        语句组n
else:
        语句组n+1
```

依次计算逻辑表达式1、逻辑表达式2…只要碰到一个表达式i为真，则执行语句组i（前面为假的表达式对应的语句组不会被执行），且后面的表达式不再计算，后面的语句组也都不会被执行。

若所有表达式都为假，则执行语句组n+1

条件分支语句的缩进

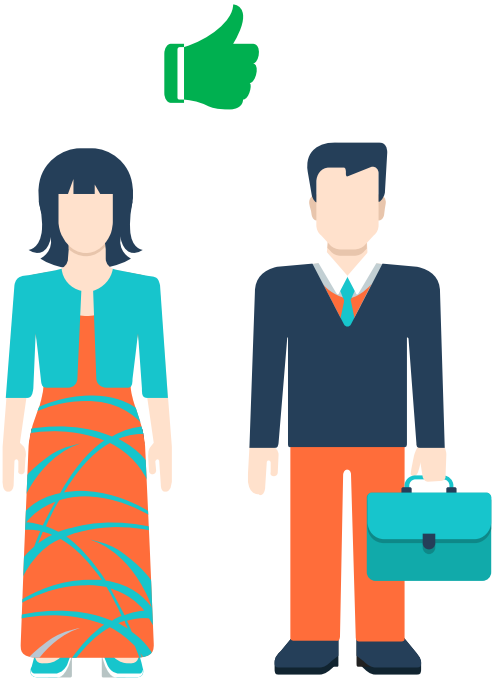
- if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样(对齐)
- 一般编译器会自动缩进

```
if int(input()) == 5:
        print("a",end="")
        print("b")
```

输入：5　　输出：ab
输入：4　　无输出

条件分支语句的缩进

- if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样(对齐)



```
if int(input()) == 5:
    print("a",end="")
print("b")
```

输入：4　　输出：b

条件分支语句的缩进

- if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样(对齐)

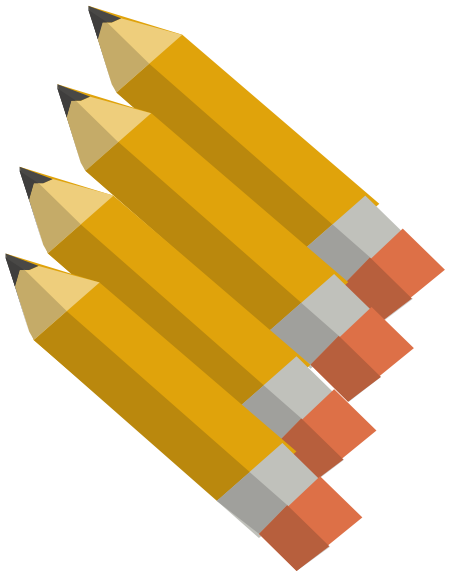
```
if int(input()) == 5:
        print("a",end="")
    print("b")
```

出错！没有对齐的缩进!

if 语句示例

例题：写一个判断整数奇偶性的程序，要求输入一个整数，如果是奇数，就输出“It's odd.”，如果是偶数，就输出“It's even.”。

```
if int(input()) % 2:
        print("It's odd")
  else:
        print("It's even")
```



if 语句嵌套

- 在一条if语句的某个分支(语句组）里，还可以再写if语句。

```
a = int(input())
if a > 0:
    if a % 2:
        print("good")
    else:
        print("bad")
```

输入： 4　　输出：bad
输入： 3　　输出：good
输入： -1　　无输出

程序演示

循环语句

循环语句

有时，需要重复多次执行一系列语句，因此需要循环语句

```
for <variable> in <sequence>:
    <statements 1>
```

依次对 sequence中的每个值,执行 <statements 1>

for循环语句

```
for i in range(5):      #[0,5)中的每一个整数
    print(i)
```

输出

```
0
1
2
3
4
```

```
for i in range(5,9):   #[5,9)中的每一个整数
    print(i)
```

输出

```
5
6
7
8
```

for循环语句

```
for i in range(0, 10, 3):   #步长3
    print(i)
```

输出

```
0
3
6
9
```

```
for i in range(-10, -100, -30):
    print(i)
```

输出

```
-10
-40
-70
```

for循环语句

```
for i in range(0):
    print(i)
```

无输出

```
for i in range(2,2) :
    print(i)
```

无输出

for循环语句

```
a = ['Google', 'Apple', 'IBM', 'Taobao', 'Wechat']
for i in range(len(a)):# len,求列表长度（元素个数）
    print(i, a[i])
```

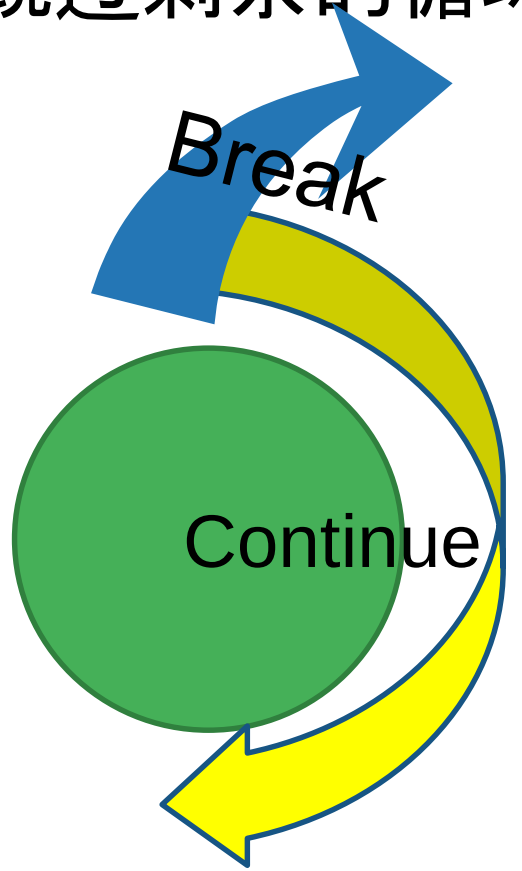
```
0 Google
1 Apple
2 IBM
3 Taobao
4 Wechat
```

len也可以用来求字符串长度

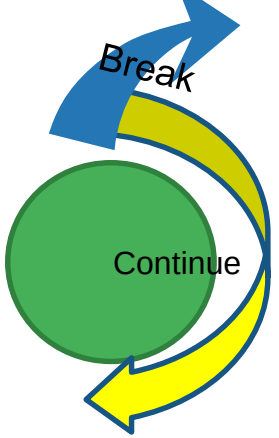
```
print(len("abc"))      #>>3
```

for循环语句

- break：跳出循环
- continue：跳过剩余的循环体，但是不结束循环



for循环语句

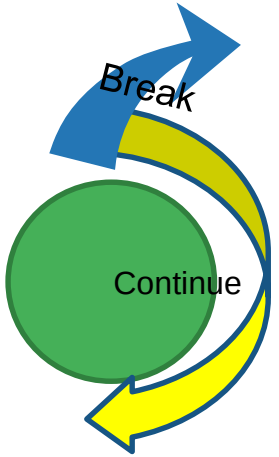


- break: 跳出循环

```
sites = ["Baidu", "Google", "IBM", "Taobao"]  #list
for site in sites:  #对sites中的每个值 site
    if site == "IBM":
        print("OK")
        break               #跳出循环
    print("site: " + site)
print("Done!")
```

```
site: Baidu
site: Google
OK
Done!
```

for循环语句



- continue：跳过剩余的循环体，但是不结束循环

```
for letter in 'Taobao':
    if letter == 'o':           # 字母为 o 时跳过输出
        continue                # 直接跳到下次循环
    print ('当前字母 :', letter)
```

```
当前字母 : T
当前字母 : a
当前字母 : b
当前字母 : a
```

for循环语句示例

➢ 例题：在一行输入若干个整数，输出最大值

```
s = input().split()
maxV = int(s[0])
for x in s:
    maxV = max(maxV,int(x))     #max 是pyton的函数
print(maxV)
```

for循环语句示例

➢ 例题：输入一个正整数n，从小到大输出它的所有因子

```
n = int(input())
    for x in range(1,n+1):
        if n % x == 0:
            print(x," ",end="")
```

15 ↙
1 3 5 15

for循环语句示例

➢ 例题： 输入一个正整数n，从大到小输出它的所有因子

```
n = int(input())
   for x in range(n,0,-1):   #步长-1
        if n % x == 0:
            print(x," ",end="")
```

```
15 ↙
15 5 3 1
```

循环语句嵌套

- 循环可以嵌套，形成多重循环：

```
for i in range(n):
    .....
    for j in range(m):
        .....   #内重循环的执行次数一共是n×m次
```

while循环语句

1. while 逻辑表达式 exp:
2. 语句组1

.....

1) 判断 exp 是否为真，若为真，转2)，若为假，转3)
2) 执行 语句组1，回到 1)
3) 继续往下执行 .....

while循环语句

- 输出小于5的非负数

```
count = 0
while count < 5:
    print (count, " 小于 5")
    count = count + 1
print (count, " 大于或等于 5")
```

```
0 小于 5
1 小于 5
2 小于 5
3 小于 5
4 小于 5
5 大于或等于 5
```

while循环语句

```
while True:
        ........
        if exp:
              break
        .....
```

不停执行，直到 exp为真时跳出循环

while循环语句示例

➢ 例题：输入一个正整数n，从小到大输出它的所有因子

```
n = int(input())
for x in range(1,n+1):
    if n % x == 0:
        print(x," ",end="")
```

或：

```
n = int(input())
x = 1
while x <= n:
    if n % x == 0:
        print(x," ",end="")
    x += 1
```

```
15 ↙
1 3 5 15
```

while循环语句

- 输入三个整数，求它们的最小公倍数

```
s = input().split()
x,y,z = int(s[0]),int(s[1]),int(s[2])
n = 1
while True:
    if n % x == 0 and n % y == 0 and n % z == 0:
        print(n)
        break
    n = n + 1
```

例题：求阶乘的和

给定正整数n，求不大于n的正整数的阶乘的和
（即求1!+2!+3!+...+n!）

输入
输入有一行，包含一个正整数n（1 < n < 12）。

输出
输出有一行：阶乘的和。

样例输入
5

样例输出
153



例题：求阶乘的和

```
n = int(input())
s = 0
for i in range(1,n+1):
    f = 1 #存放i阶乘
    for j in range(1,i+1):
        f *= j   #此操作一共做1+2+3+...+n次
    s += f
print(s)
```

程序演示

Thank You

孙栩
xusun@pku.edu.cn