

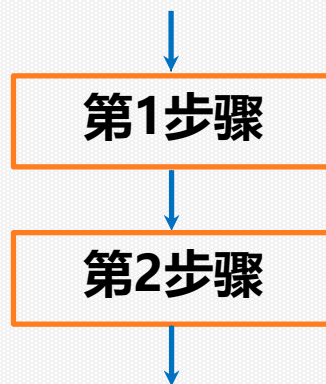


第3章 程序流程控制

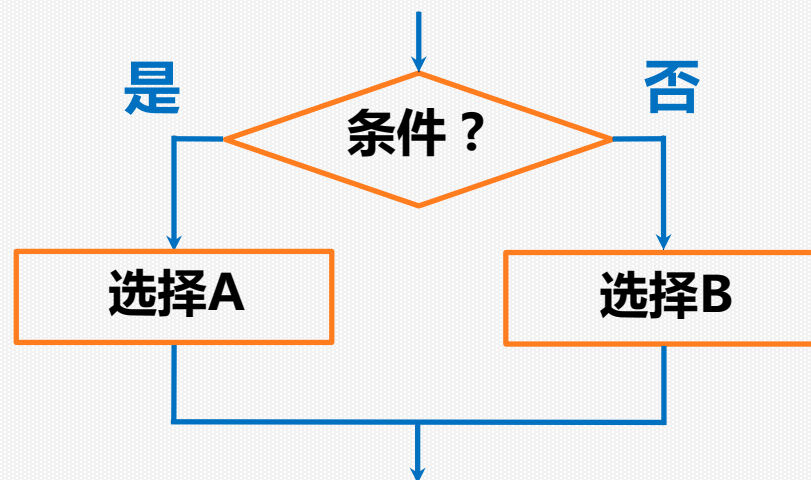


程序的控制结构

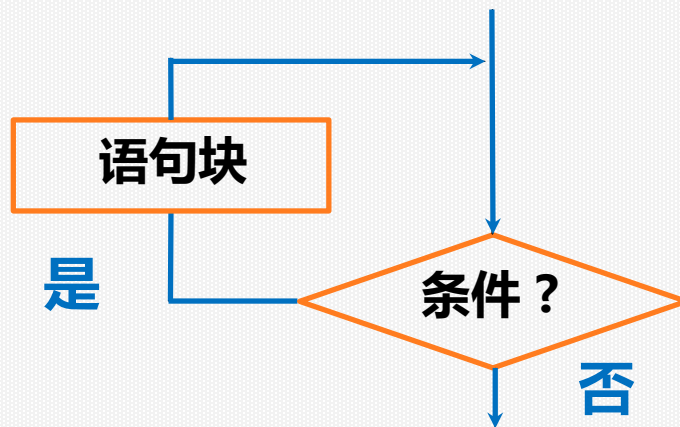
- 顺序结构



- 分支结构



- 循环结构



问题：实现一个猜数字的小游戏

猜数字游戏

```
target = 78 #先给定一个值
```

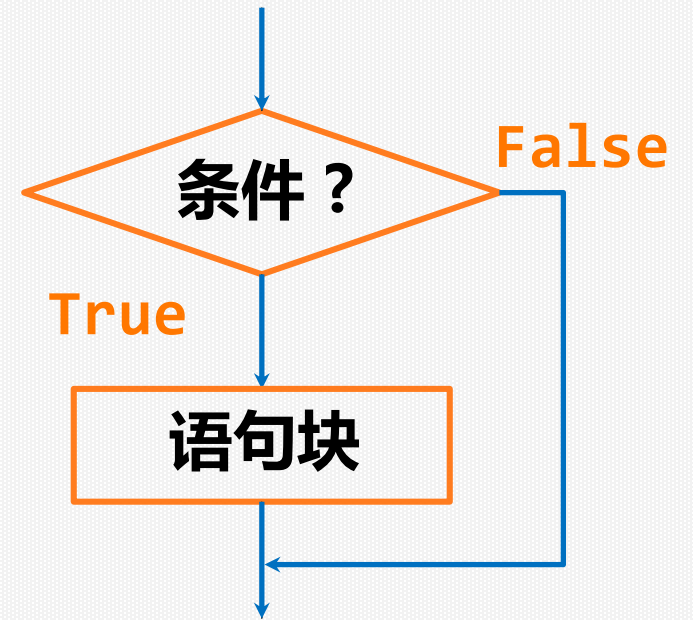
```
guess = int(input("What's your guess:"))
```

```
? ? ?
```

单分支结构

根据判断条件结果而选择不同向前路径的运行方式

if 条件 :
语句块



- 条件表达式后面的 “:” 不可缺少。
- 在写 “语句块” 的时候，务必注意代码缩进，且同一个代码块必须保证相同的缩进量。

猜数字游戏单分支解决方案：猜对输出结果

```
target = 78 #先给定一个值
```

```
guess = int(input("What's your guess:"))
```

```
if guess == target:
```

```
    print("Good guess - you found it!")
```

单分支结构

单分支例题：从键盘输入两个任意整数，将其按照从大到小的顺序输出。

```
a = int(input("请输入整数a: "))  
b = int(input("请输入整数b: "))  
print("输入值 a={},b={}".format(a, b))  
if a < b:
```

```
    a, b = b, a
```

```
print("比较后的值 a={},b={}".format(a, b))
```

两个变量互换

是否需要缩进？

猜数字游戏单分支解决方案：猜对输出结果

```
target = 78 #先给定一个值
guess = int(input("What's your guess:"))
if guess == target:
    print("Good guess - you found it!")
```

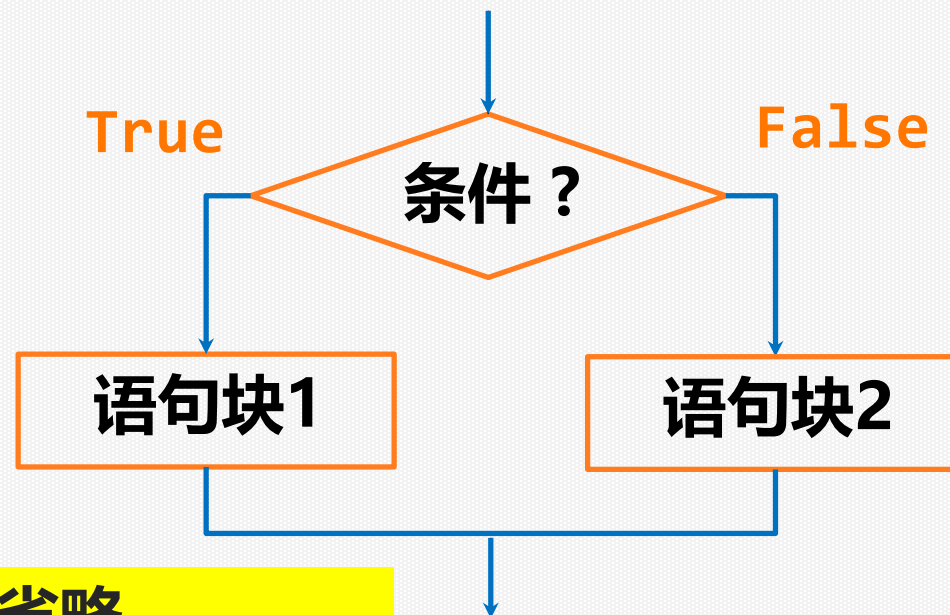
当猜错时也需要提示信息???

双分支结构

根据判断条件结果而选择不同向前路径的运行方式

if 条件 :
语句块1

else :
语句块2



- 冒号不能省略
- 语句块注意缩进
- *else*后面不要加条件

猜数字游戏双分支解决方案：猜对和错都输出提示信息

```
target = 78
guess = int(input("What's your guess:"))
if guess == target:
    print("Good guess - you found it!")
else:
    print("Wrong guess - don't give up!")
```

双分支结构

双分支紧凑形式：适用于简单表达式的二分支结构

表达式1 *if* 条件 *else* 表达式2

```
print("Good") if guess == target else print("Wrong")
```

双分支例题：

- 判断一个字符串是否为回文。

```
s = input("请输入字符串：")  
if (s == s[::-1]):  
    print(s + "为回文串")  
else:  
    print(s + "不是回文串")
```

mum
level

猜数字游戏双分支解决方案：猜对和错都输出提示信息

```
target = 78
guess = int(input("What's your guess:"))
if guess == target:
    print("Good guess - you found it!")
else:
    print("Wrong guess - don't give up!")
```

猜数字游戏if嵌套决方案：当猜错时嵌套一个if语句输出提示信息（太大或太小）

```
target = 78
guess = int( input("What's your guess:") )
if guess != target:
    print("Good guess - you found it!")
else:
    if guess > target:
        print("Too high!")
    else:
        print("Too small")
```



If嵌套例题：

```
num = input("请输入一个正整数： ")  
  
print(num, "中最大的数字是： ", max(num) )
```

学而不思则罔
思而不学则殆

猜数字游戏if嵌套决方案：当猜错时嵌套一个if语句输出提示信息（太大或太小）

```
target = 78
guess = int( input("What's your guess:") )
if guess == target:
    print("Good guess - you found it!")
else:
    if guess > target:
        print("Too high!")
    else:
        print("Too small")
```

多分支结构

if 条件1 :

语句块1

elif 条件2 :

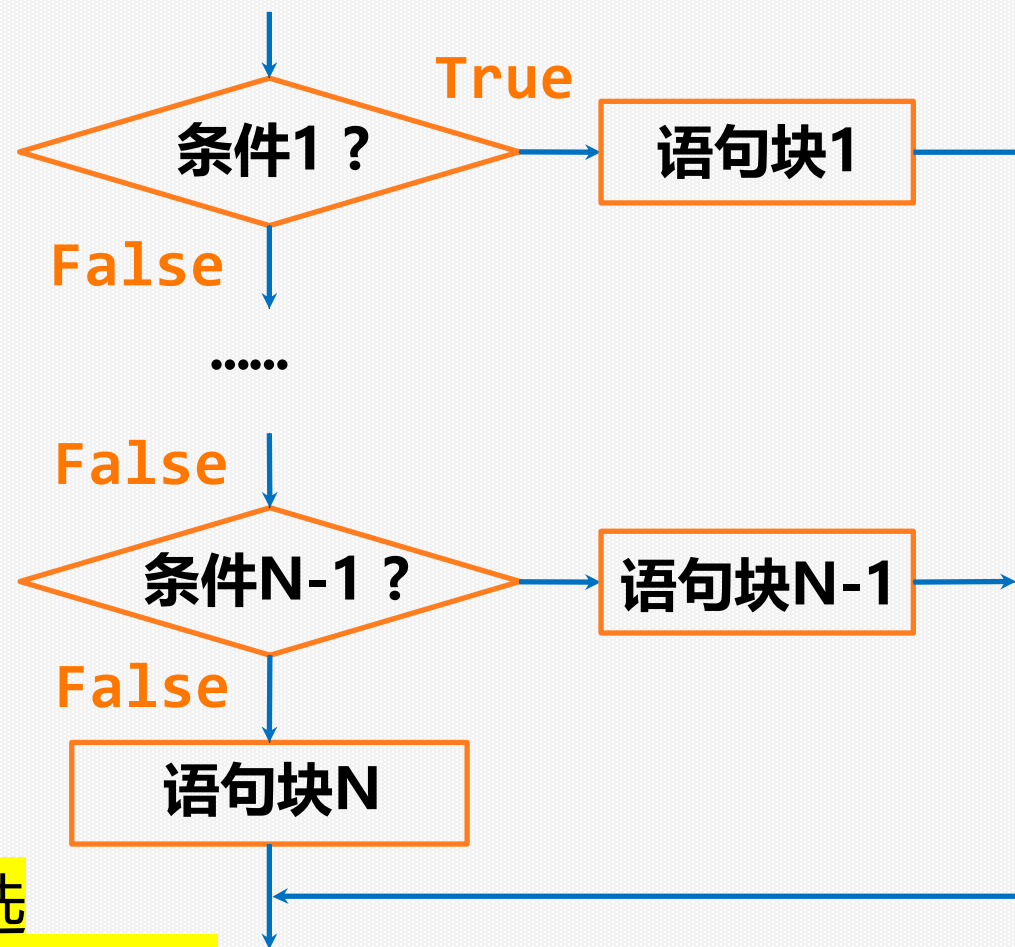
语句块2

.....

else :

语句块N

- else分支为可选
- else后面不可以加条件



猜数字游戏多分支解决方案： 利用多分支分别输出提示信息（猜对，太大或太小）

```
target = 78
guess = int(input("What's your guess:"))
if guess == target:
    print("Good guess - you found it!")
elif guess > target:
    print("Too high!")
else:
    print("Too small")
```

多分支例题：

```
score = eval(input())
if score < 60:
    grade = "F"
elif score < 70:
    grade = "D"
elif score < 80:
    grade = "C"
elif score < 90:
    grade = "B"
else:
    grade = "A"
print("输入成绩属于级别{}".format(grade))
```

score >= 90

score >= 60



猜数字游戏多分支解决方案：利用多分支分别输出提示信息（猜对，太大或太小）

```
target = 78
guess = int(input("What's your guess:"))
if guess == target:
    print("Good guess - you found it!")
elif guess > target:
    print("Too high!")
else:
    print("Too small")
```



重复执行

本周课程作业（第6周）

① 阅读 第三章知识点

课本第3章选择结构P34-43，例题程序复现

② 复习 课件中的实例

可在解释器中输入代码，运行输出结果

③ 完成实验书中的题目（上机课进行）

一 实验准备（p51-57,1-3上机前填到书上，
4上机前已有思路）

二 实操

上机实践 P41-P52 范例的程序复现

提醒：审题（确定需要几个变量，输入什么数据，用什么算法，输出什么结果）；

编写程序（尽量不看答案）；

运行，改错，再运行，测试运行测试；

对照参考答案，反思。

④ 提交 python123中的作业（上机课后）

⑤ 预习（第8周上课前）

range函数、random库的应用、循环结构



练习：

- 1.从键盘输入一个百分制成绩，60分及以上输出“Pass”，否则输出“Fail”。
- 2.输入 PM2.5 的值，输出对应的空气质量。已知 PM2.5 的值与空气质量的对应关系为：

$pm \leq 50$	优
$50 < pm \leq 100$	良
$100 < pm \leq 150$	轻度污染
$150 < pm \leq 200$	中度污染
$200 < pm \leq 300$	重度污染
$pm > 300$	严重污染

【练习1】上周作业（python123）：获得用户的输入当作宽度，以*作为填充符号右对齐输出PYTHON字符串。请完善代码。

```
w = input()
s = "PYTHON"
print("{0:*>{1}}".format(s,w))
```


【练习2】无空隙回声输出：获得用户输入，去掉其中全部空格，将其他字符按输入顺序打印输出。

```
txt = input()
s = txt.replace(' ','')
print(s)
```

【练习3】 BMI ： Body Mass Index 国际上常用的衡量人体肥胖和健康程度重要标准，主要用于统计分析
定义：BMI = 体重 (kg) /身高2(m2)
获取用户输入的体重和身高值，根据右表计算并给出国际和国内的 BMI 分类

```
height, weight = eval(input())
bmi = weight / pow(height, 2)
print("BMI数值为:{:.2f}".format(bmi))
who, nat = "", ""
if bmi < 18.5:
    who, nat = "偏瘦", "偏瘦"
elif 18.5 <= bmi < 24:
    who, nat = "正常", "正常"
elif 24 <= bmi < 25:
    who, nat = "正常", "偏胖"
elif 25 <= bmi < 28:
    who, nat = "偏胖", "偏胖"
elif 28 <= bmi < 30:
    who, nat = "偏胖", "肥胖"
else:
    who, nat = "肥胖", "肥胖"
print("BMI指标为:国际'{0}',国内'{1}'".format(who, nat))
```

分类	国际BMI值	国内BMI值
偏瘦	<18.5	<18.5
正常	18.5 ~ 25	18.5 ~ 24
偏胖	25 ~ 30	24 ~ 28
肥胖	≥30	≥28

【练习4】使用 random 函数库，编写一个程序，输入一个整数做为随机数种子，随机产生一个 0 - 100 之间的整数并输出。

```
import random
n = eval(input())
random.seed(n)
print(random.randint(0,100))
#print(random.randrange(101))
```

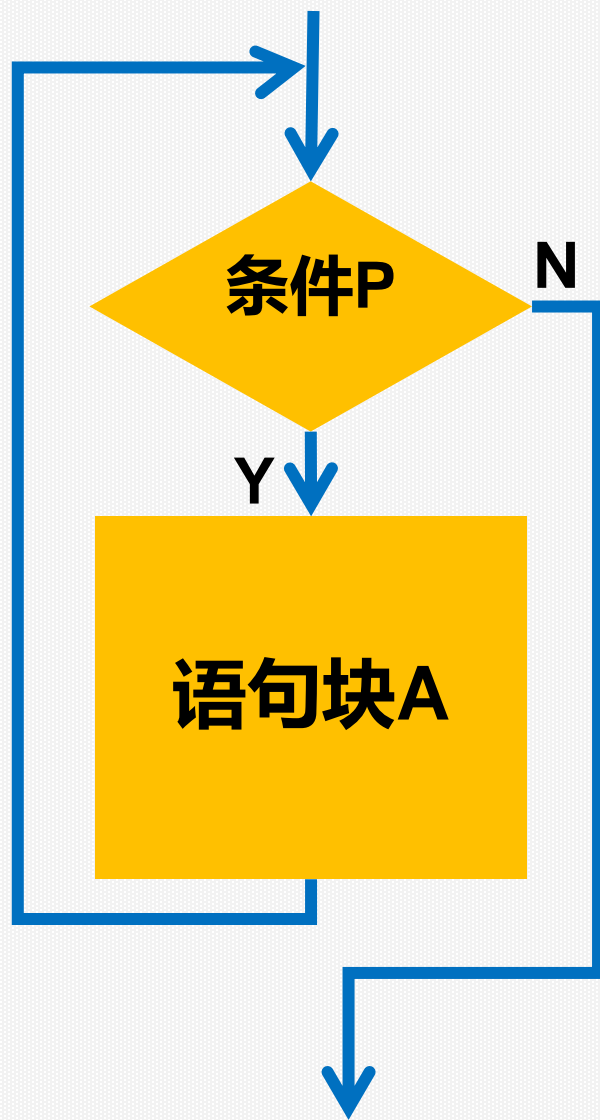
The End

下次课讲循环结构

下面内容仅供预习参考，同学们
是否看以下幻灯片可自行决定，
不做要求

循环结构

某些操作需要反复执行多次



循环的两大要素：

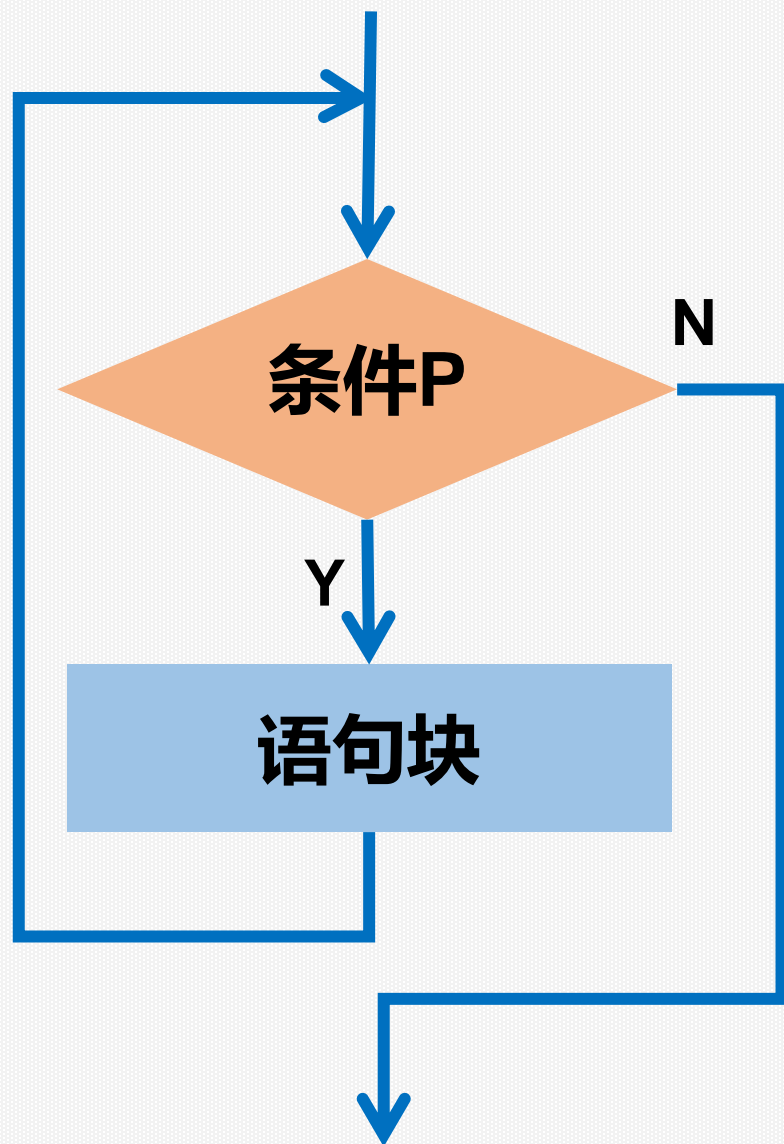
- **语句块A：循环体**
- **条件P：循环控制条件**

python中实现循环结构的语句：

for循环： 遍历循环

while循环： 条件循环

while 循环： 由条件控制的循环



while 条件表达式：
语句块

- **执行过程：** 先计算条件表达式的值，如果值为True则执行循环体语句块一次，然后继续判断条件表达式的值；当条件表达式的值为False的时候，结束循环。

猜数字游戏while循环解决方案： 利用while循环实现猜多次，直到猜对停止

```
import random  
target = random.randint(0,100)
```

```
found = False
```

```
while not found:
```

```
    guess = int(input("What's your guess: "))
```

```
    if guess == target:
```

```
        print("Good guess - you found it")
```

```
        found = True
```

```
    elif guess < target:
```

```
        print("too low")
```

```
    else:
```

```
        print("too high")
```

随机生成一个0到100的整数

found为一个逻辑型变量

循环体

while语句例题：

- 输出5行#号，每行10个。

```
i = 0
```

```
while i < 5:
```

```
    print(' #' * 10)
```

```
    i += 1
```

while循环也可以解决循环次数已知的循环

循环变量 i 控制循环：

- 初始值
- 循环变量的变化
- 循环条件

for循环：遍历循环

for 循环变量 **in** 迭代器：
语句块

迭代器 iterator: 循环控制器

- **for**和**in**都是关键字，由关键字**for**开始的行称为**循环的头部**，语句块称为**循环体**。语句块需要**缩进**，且块中各个语句的缩进量必须相同。
- **执行过程**：每次循环，从迭代器中取出一个值赋给循环变量，而后执行一次循环体；当完整遍历所有元素后结束循环。

```
for i in range(5):  
    print(i)
```

```
for c in "Hello" :  
    print(c)
```

```
for i in [1,2,3,4]:  
    print(i)
```

```
for i in (1,2,3,4):  
    print(i)
```

for语句例题:

- 输入一串字符, 统计其中的大写字母, 小写字母和数字各有多少个? 。

```
mystr = input("Enter a string: ")
count_upper = count_lower = count_digit = 0
for s in mystr:
    if s.isupper(): count_upper += 1
    if s.islower(): count_lower += 1
    if s.isdigit(): count_digit += 1
print("大写字符: ", count_upper)
print("小写字符: ", count_lower)
print("数字字符: ", count_digit)
```


for语句例题:

- 累加/累乘问题: $1+2+3\ldots+100$ 。

```
total = 0
for i in range(1, 101):
    total += i
print(total)
```

so easy ! ! !



1. $2+4+6\ldots100$

2. $1+1/2-1/3+1/4\ldots 1/n$

3. $1*2*3\ldots5$

4. $1+2!+\ldots n!$

5. $x/1!-x^3/3!+ x^5/5!-x^7/7!$

6. 求1至100中奇数和偶数的和

猜数字游戏while循环解决方案：利用while循环实现猜多次，直到猜对停止

```
import random
target = random.randint(0,100)
found = False
while not found:
    guess = int(input("What's your guess: "))
    if guess == target:
        print("Good guess - you found it")
        found = True
    elif guess < target:
        print("too low")
    else:
        print("too high")
```

**规定最多猜5次？
若5次以内猜对提前结束循环；**

**可以用for循环实现5次循环
但怎样提前结束循环呢？**

break 和 continue 语句

break 和 continue

- **break跳出并结束当前整个循环，执行循环后的语句**
- **continue结束当次循环，继续执行后续次数循环**
- **break和continue可以与for和while循环搭配使用**

break 和 continue 语句示例

```
for i in range(1, 11):  
    if i % 3 == 0:  
        break  
    print(i, end='  ')
```

输出: 1 2

当i是3的倍数的时候，执行break语句。
break语句的作用是**立刻结束整个for循环**，
因此输出只有1和2两个数字。

```
for i in range(1, 11):  
    if i % 3 == 0:  
        continue  
    print(i, end='  ')
```

输出: 1 2 4 5 7 8 10

当i是3的倍数的时候，执行continue语句。
continue语句的作用是**结束本轮的循环**，程序跳
转到循环头部，根据头部的要求继续，因此输出不
是3的倍数的所有数字。

猜数字游戏for循环解决方案： 利用for循环实现最多猜5次，若5次以内猜对则提前结束循环

```
import random
```

```
target = random.randint(0,100)
```

```
for i in range(5):
```

```
    guess = int(input("What's your guess: "))
```

```
    if guess == target:
```

```
        print("Good guess - you found it")
```

```
        break
```

```
    elif guess < target:
```

```
        print("too low")
```

```
    else:
```

```
        print("too high")
```

for循环控制循环5次

**5次以内猜对用break
语句结束循环**

continue?

不用和用break语句实现的方法

猜数字游戏while循环解决方案：利用while循环实现猜多次，直到猜对停止

```
import random
target = random.randint(0,100)
found = False
while not found:
    guess = int(input("What's your guess: "))
    if guess == target:
        print("Good guess - you found it")
        found = True
    elif guess < target:
        print("too low")
    else:
        print("too high")
```

```
import random
target = random.randint(0,100)
k=0
while 1:
    guess = int(input("What's your guess: "))
    k=k+1
    if guess == target:
        print("Good guess - you found it")
        break
    elif guess < target:
        print("too low")
    else:
        print("too high")
print("You haved guessed {} times!".format(k))
```

循环的扩展

循环与else

<i>for</i> <变量> <i>in</i> <遍历结构> :	<i>while</i> <条件> :
<语句块1>	<语句块1>
<i>else</i> :	<i>else</i> :
<语句块2>	<语句块2>

else 子句

```
for i in range(5):  
    print(i, end = " ")  
else:  
    print("for循环正常结束!")
```

输出: 0 1 2 3 4 for循环正常结束!

**遍历结束而正常退出循环
执行else中的语句**

```
for i in range(5):  
    print(i, end = " ")  
    if i >= 3: break  
else:  
    print("for循环正常结束!")
```

输出: 0 1 2 3

**由于break语句而提前退出循环
else中语句不被执行**

猜数字游戏for循环解决方案：利用for循环实现最多猜5次，若5次以内猜对则提前结束循环；若5次还未猜对则停止游戏并给出未提示信息。

```
import random
target = random.randint(0,100)
for i in range(5):
    guess = int(input("What's your guess: "))
    if guess == target:
        print("Good guess - you found it")
        break
    elif guess < target:
        print("too low")
    else:
        print("too high")
else:
    print("Too many times!")
```

循环5次后执行else语句块

循环的嵌套

在一个循环体内又包含一个完整的循环结构，称为循环的嵌套。这种语句结构称为多重循环结构。内层循环中还可以包含新的循环，形成多重循环结构。

猜数字游戏循环嵌套解决方案：实现多轮猜数字游戏。

```
import random
```

```
reply = 'y'  
while reply == 'y':
```

```
    target = random.randint(1, 100)
```

```
    for i in range(5):
```

```
        guess = int(input("What's your guess: "))
```

```
        if guess == target:
```

```
            print("Good guess - you found it")
```

```
            break
```

```
        elif guess < target:
```

```
            print("too low")
```

```
        else:
```

```
            print("too high")
```

```
    else:
```

```
        print("Too many times!")
```

```
    reply = input('Play again(y/n):')
```

设一个reply变量，初始值为'y'

while循环中嵌套了一个for循环

每一轮游戏结束时，重新输入reply

```
for i in range(3):
    for j in range(2):
        print(i, j)
```

```
0 0
0 1
1 0
1 1
2 0
2 1
```

```
for i in range(2):
    for j in range(3):
        print(i, j, end= "" )
```

```
for i in range(2):
    for j in range(3):
        print(i, j, end= "" )
    print()
```

```
for i in range(2):
    for j in range(3):
        print(i, j)
```

```
0 0
0 1
0 2
1 0
1 1
1 2
```


```
0 0 0 1 0 2 1 0 1 1 1 2
```

```
0 0 0 1 0 2
1 0 1 1 1 2
```

```
>>> s = "PYTHON"
>>> while s != "":
    for c in s:
        print(c, end="")
    s = s[:-1]
```

PYTHONPYTHOPYTHPYTPYP

```
>>> s = "PYTHON"
>>> while s != "":
    for c in s:
        if c == "T":
            break
        print(c, end="")
    s = s[:-1]
```



PYPYPYPYPYP

- **break**仅跳出当前最内层循环，即其所在的那层循环

复盘：

顺序结构
分支结构
循环结构

