

Python基础语法精讲

嵩天

分支与循环

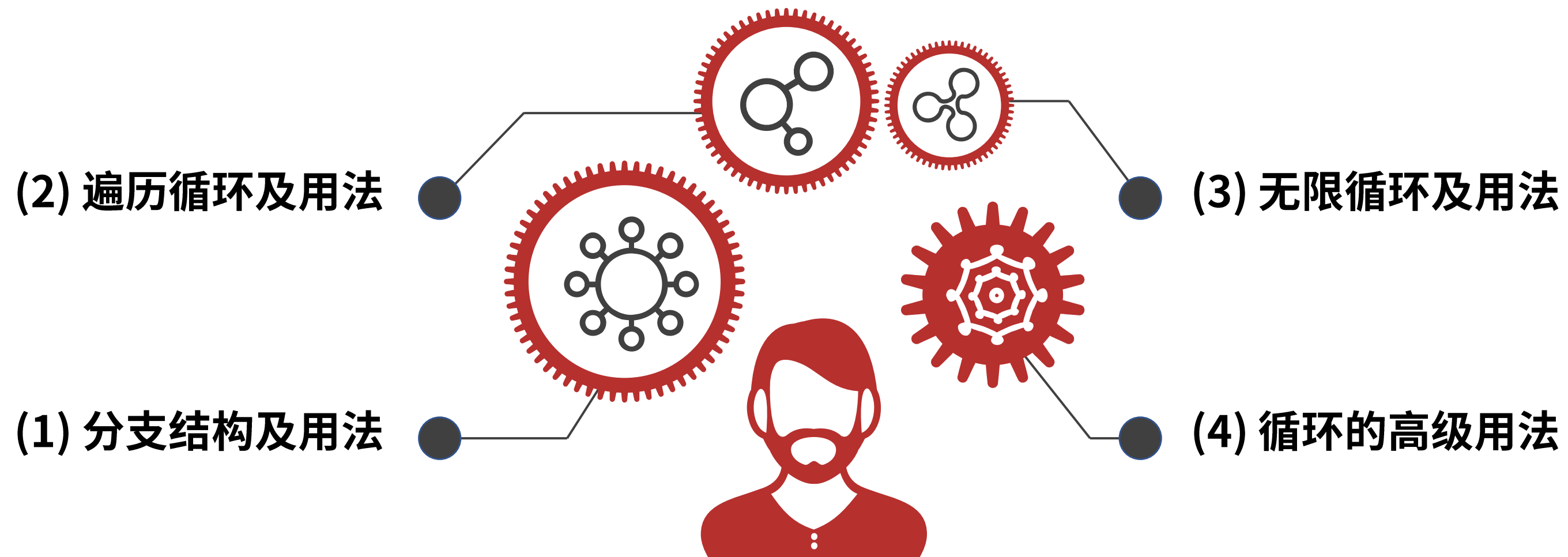
高天



分支与循环

单元开篇

单元开篇



分支与循环

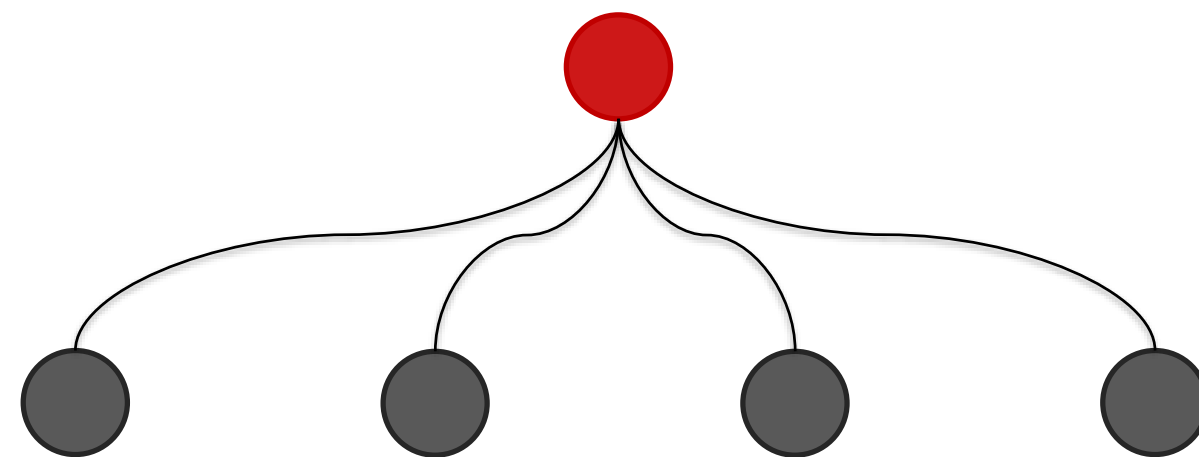


分支与循环

分支结构 及用法

分支结构及用法

程序的分支结构



单分支结构

二分支结构

多分支结构

条件判断及组合

单分支结构

最基本的分支结构

if <条件> :
 <语句块>

```
guess = eval(input())  
if guess == 99:  
    print("猜对了")
```

二分支结构

标准用途的分支结构

if <条件> :
 <语句块1>

else :
 <语句块2>

```
guess = eval(input())  
if guess == 99:  
    print("猜对了")  
else :  
    print("猜错了")
```


二分支结构

二分支的紧凑形式

<表达式1> *if* <条件> *else* <表达式2>

```
guess = eval(input())  
print("猜{}了".format("对" if guess==99 else "错"))
```

注意：紧凑形式的元素是表达式，不能用于语句

多分支结构

分支结构的常用方式

if <条件1> :

<语句块1>

elif <条件2> :

<语句块2>

.....

else:

<语句块N>

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

逻辑错误程序

注意：多条件之间的包含关系、变量取值范围的覆盖

条件判断及组合

True和False

- Python语言提供True和False表示真假
- 任何判断产生的结果是True或False
- False的等价值是：None, 0, 0.0, 0j, '', (), [], {}

条件判断及组合

比较判断

操作符	数学符号	描述
<	<	小于
<=	≤	小于等于
>=	≥	大于等于
>	>	大于
==	=	等于
!=	≠	不等于

条件的组合

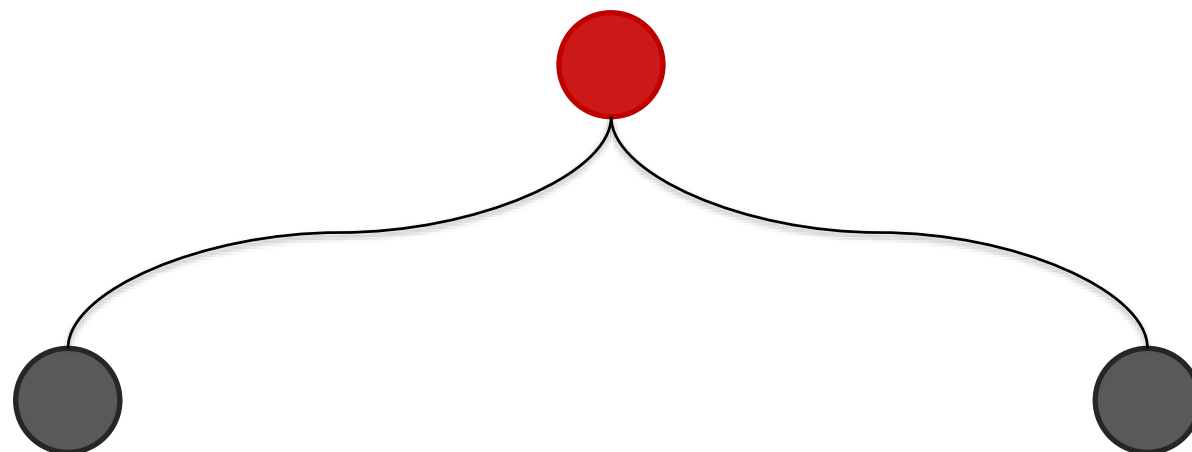
操作符及使用	描述
<code>x and y</code>	两个条件x和y的逻辑与
<code>x or y</code>	两个条件x和y的逻辑或
<code>not x</code>	条件x的逻辑非

```
guess = eval(input())
if guess > 99 or guess < 99:
    print("猜对了")
else :
    print("猜错了")
```

分支与循环

循环遍历 及用法

程序的循环结构



遍历循环（for循环）

无限循环（while循环）

遍历循环

遍历某个结构形成的循环运行方式

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

The diagram illustrates the execution of a for loop. A blue arrow points from the loop variable placeholder '<循环变量>' to the statement block '<语句块>'. Another blue arrow points from the end of the statement block back to the loop structure placeholder '<遍历结构>', indicating the iterative nature of the loop.

- 由保留字for和in组成，完整遍历所有元素后结束
- 每次循环，从遍历结构中逐一提取元素，放在循环变量中，并执行一次语句块

遍历循环

循环遍历可用于任何遍历结构

- 计数循环(N次)
- 计数循环(特定次)
- 字符串遍历循环
- 列表遍历循环
- 文件遍历循环
- 元组遍历循环
- 集合遍历循环
- 字典遍历循环
-

遍历循环的应用

计数循环(N次)

for i in range (N) :

<语句块>

```
for i in range(1, 6) :  
    print(i)
```

```
1  
2  
3  
4  
5
```

遍历由range()函数产生的数字序列产生循环

遍历循环的应用

计数循环(特定次)

```
for i in range(M, N, K) :
```

<语句块>

遍历由range()函数产生的数字序列产生循环

遍历循环的应用

字符串遍历循环

for c in s :

<语句块>

```
for c in "python123" :  
    print(ic, end=", ")
```

P, y, t, h, o, n, 1, 2, 3,

s是字符串，遍历字符串每个字符产生循环

遍历循环的应用

列表遍历循环

for item **in** ls :

<语句块>

```
for item in [123, "py", 456]:  
    print(item, end=", ")
```

123, py, 456

ls是一个列表，遍历其每个元素，产生循环

遍历循环的应用

文件遍历循环

优美胜于丑陋
明了胜于隐晦
简洁胜于复杂

for line **in** fi :

<语句块>

```
for line in fi:  
    print(line)
```

优美胜于丑陋
明了胜于隐晦
简洁胜于复杂

fi是一个文件标识符，遍历其每行，产生循环


分支与循环

无限循环 及用法

无限循环

无限遍历循环

while <条件> :
 <语句块>



```
a = 3  
while a > 0 :  
    a = a - 1  
    print(a)
```

2
1
0

反复执行语句块，直到条件不满足时结束

循环控制保留字


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

循环控制保留字


break 和 continue

```
for c in "PYTHON" :  
    if c == "T" :  
        continue  
    print(c, end=" ")
```



PYHON

```
for c in "PYTHON" :  
    if c == "T" :  
        break  
    print(c, end=" ")
```



PY



分支与循环

循环的 高级用法

循环的扩展

循环完成的奖励：else扩展

for <变量> *in* <遍历结构> :

<语句块1>

else :

<语句块2>

while <条件> :

<语句块1>

else :

<语句块2>

当循环没有被break语句退出时，执行else语句块

循环的扩展

循环完成的奖励：else扩展

```
for c in "PYTHON":  
    if c == "T":  
        continue  
    print(c, end="")  
else:  
    print("正常退出")
```

PYTHON正常退出

```
for c in "PYTHON":  
    if c == "T":  
        break  
    print(c, end="")  
else:  
    print("正常退出")
```

PY

基本数据类型

单元小结

单元小结



分支与循环

 Python ▶ 123

Thank you