

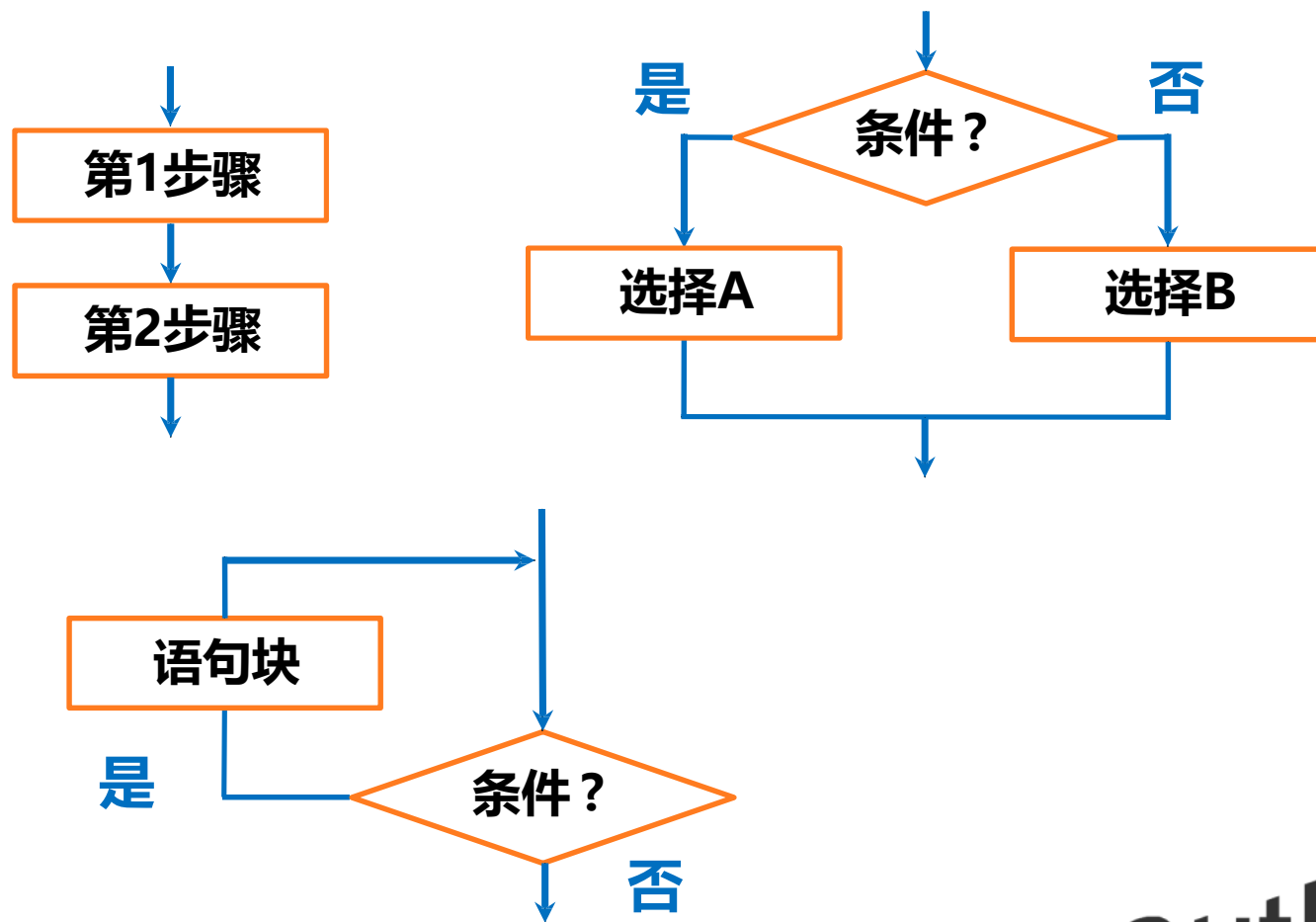
4.1 程序分支结构（二分支、多分支、条件判断与组合、程序异常处理）

"程序的控制结构"

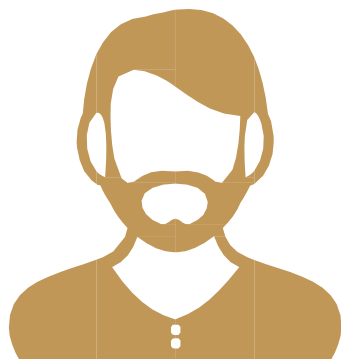
- 顺序结构

- 分支结构

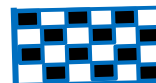
- 循环结构



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
- 程序的异常处理



- 在Python的语法规则里，只要一行语句以冒号 (:) 结尾，它接下来的内容就应该有缩进。

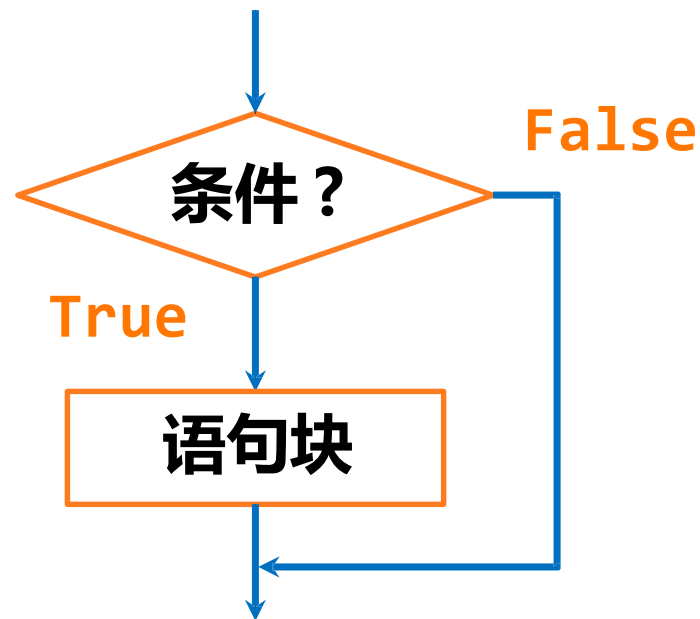


单分支结构

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>



单分支结构

单分支示例

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

```
if True:  
    print("条件正确")
```



PY01B16 阴阳



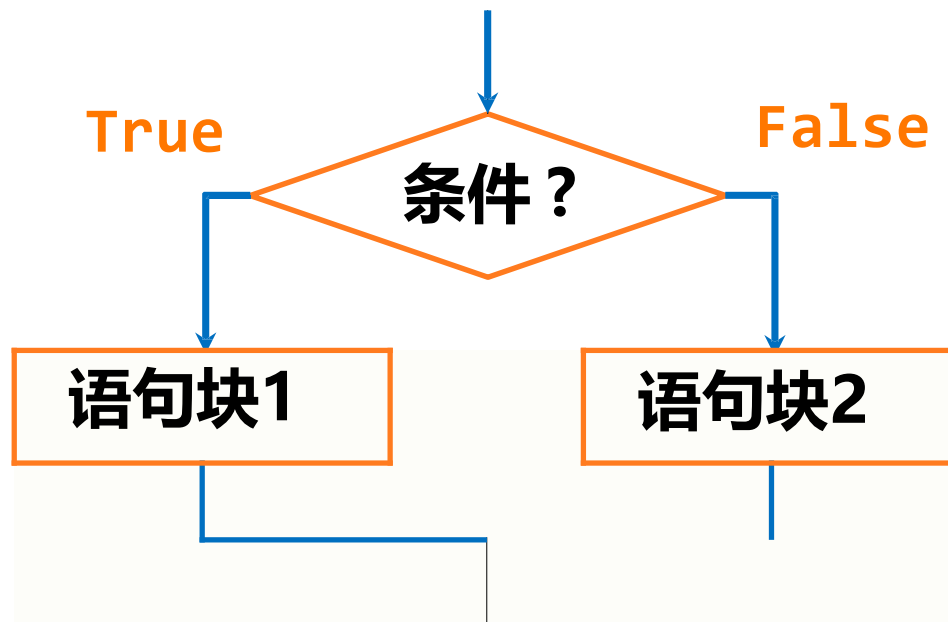
二分支结构

二分支结构

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>



二分支结构

二分支示例

```
guess = eval(input())
```

```
if guess == 99:
```

```
    print("猜对了")
```

```
else :
```

```
    print("猜错了")
```

```
if True:
```

```
    print("语句块1")
```

```
else :
```

```
    print("语句块2")
```

二分支结构

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〉

```
guess = eval(input())
```

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



多分支结构

多分支结构

if <条件1> :

<语句块1>

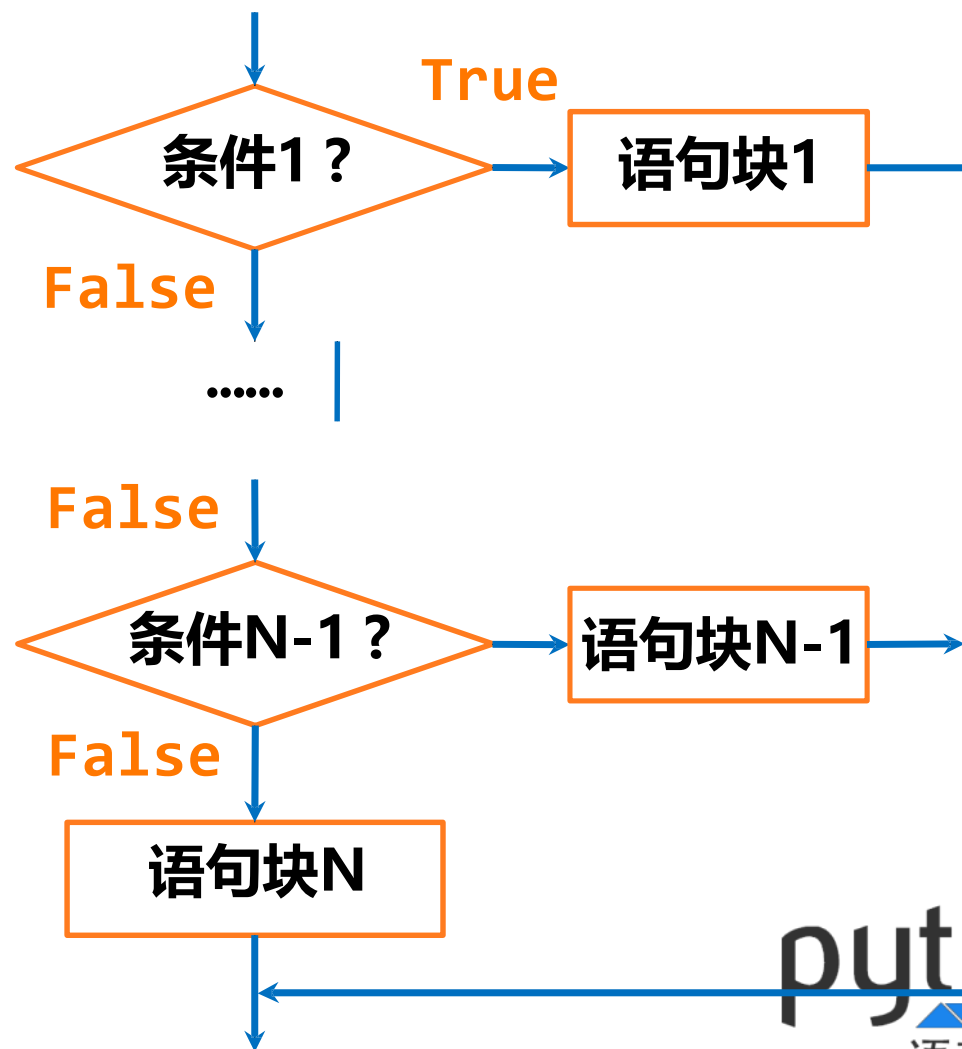
elif <条件2> :

<语句块2>

.....

else :

<语句块N>



如果多个elif 的条件都是True，那么Python只会运行它遇到的第一个是True的语句块

多分支结构

对不同分数分级的问题

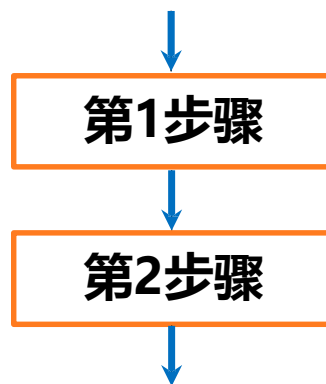
```
score = eval(input())
if score <= 60:
    grade = "D"
elif score >= 70:
    grade = "C"
elif score >= 80:
    grade = "B"
elif score >= 90:
    grade = "A"
print("输入成绩属于级别{}".format(grade))
```

- 注意多条件之间的包含关系

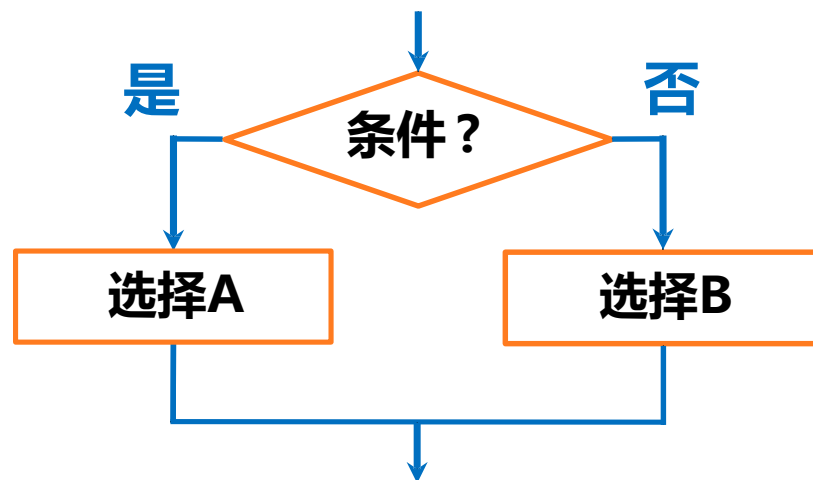
- 注意变量取值范围的覆盖

"程序的控制结构"

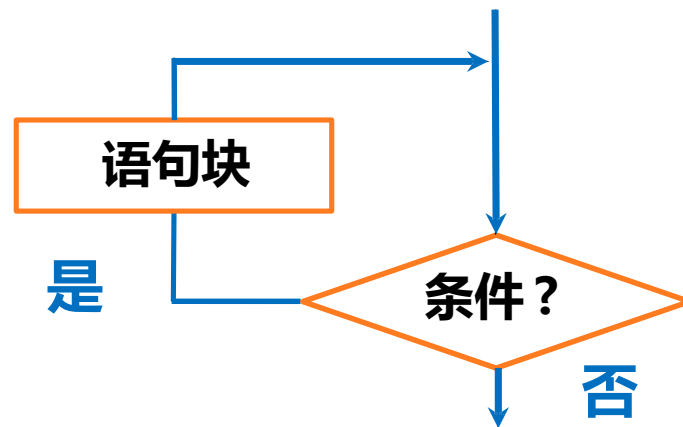
- 顺序结构



- 分支结构



- 循环结构





条件判断及组合

条件判断

操作符

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

条件组合

用于条件组合的三个保留字

操作符及使用	描述
x and y	两个条件x和y的逻辑 与
x or y	两个条件x和y的逻辑 或
not x	条件x的逻辑 非

条件判断及组合

示例

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

```
if not True:  
    print("语句块2")  
else :  
    print("语句块1")
```



程序的异常处理

异常处理

```
num = eval(input("请输入一个整数: "))  
print(num**2)
```

当用户没有输入整数时，会产生异常，怎么处理？

异常处理

异常发生的代码行数

Traceback (most recent call last):

File "t.py", line 1, in <module>

num = eval(input("请输入一个整数: "))

File "<string>", line 1, in <module>

NameError: name 'abc' is not defined

异常类型

异常内容提示

异常处理

异常处理的基本使用

try :

＜语句块1＞

except :

＜语句块2＞

try :

＜语句块1＞

except ＜异常类型＞ :

＜语句块2＞

异常处理

- 示例1

- *try* :

- `num = eval(input("请输入一个整数: "))`
`print(num**2)`

- *except* :

- `print("输入不是整数")`

异常处理

- 示例2

- *try* :

- `num = eval(input("请输入一个整数: "))`

- `print(num**2)`

标注异常类型后，仅响应此类异常

- except* `NameError:`

异常类型名字等同于变量名

- `print("输入不是整数")`

异常处理

try :

＜语句块1＞

except :

＜语句块2＞

else :

＜语句块3＞

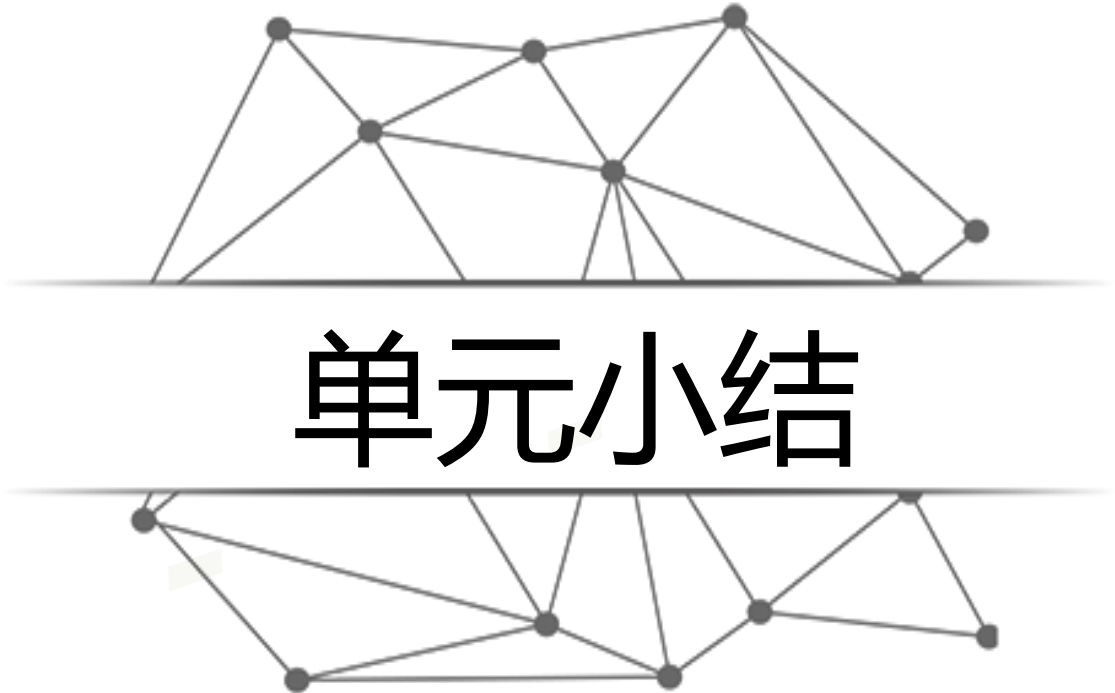
finally :

＜语句块4＞

异常处理的高级使用

- *finally* 对应语句块4一定执行

- *else* 对应语句块3在不发生异常时执行



单元小结

程序的分支结构

- 单分支 *if* 二分支 *if-else* 及紧凑形式
- 多分支 *if-elif-else* 及条件之间关系
- *not and or > >= == <= < !=*
- 异常处理 *try-except-else-finally*

