PYTHON程序设计

计算机学院 王纯

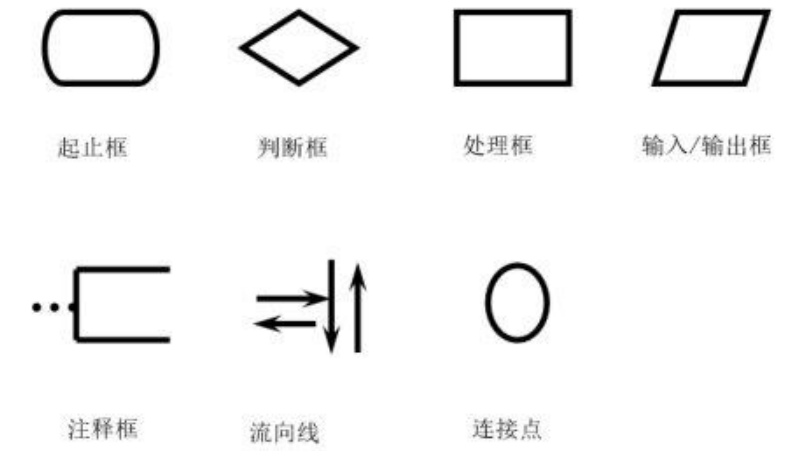
三 流程控制

三 流程控制

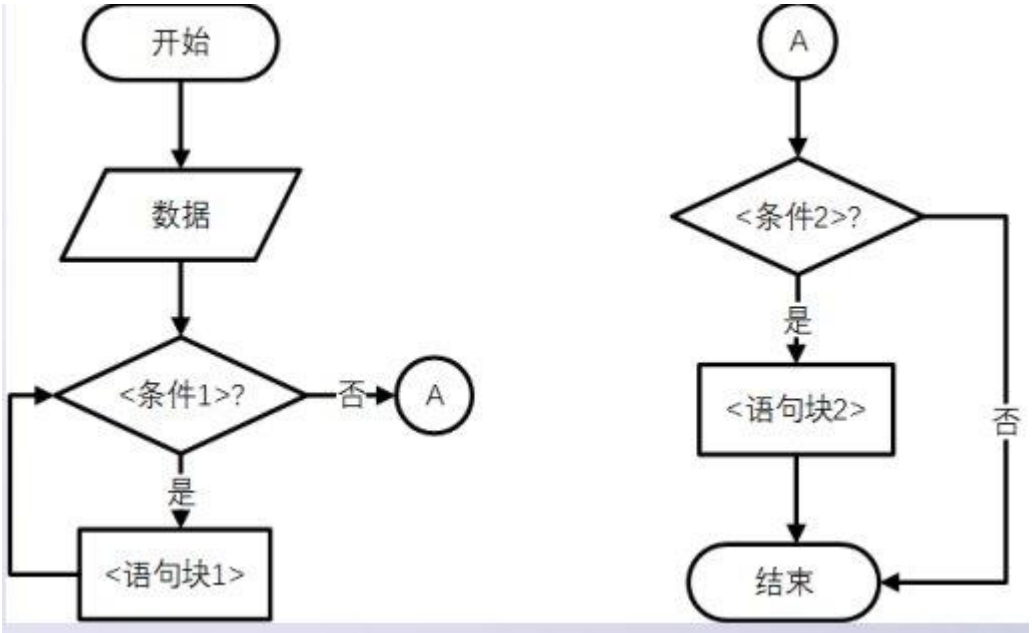
- 流程图与程序结构
- 选择控制结构
- 循环控制结构
- 综合实例

流程图与控制结构：流程图

- ✓ 程序流程图用一系列图形、 流程线和文字说明描述程序的基本操作和控制流程， 它是程序分析和过程描述的最基本方式。
- ✓ 流程图的基本元素包括7种

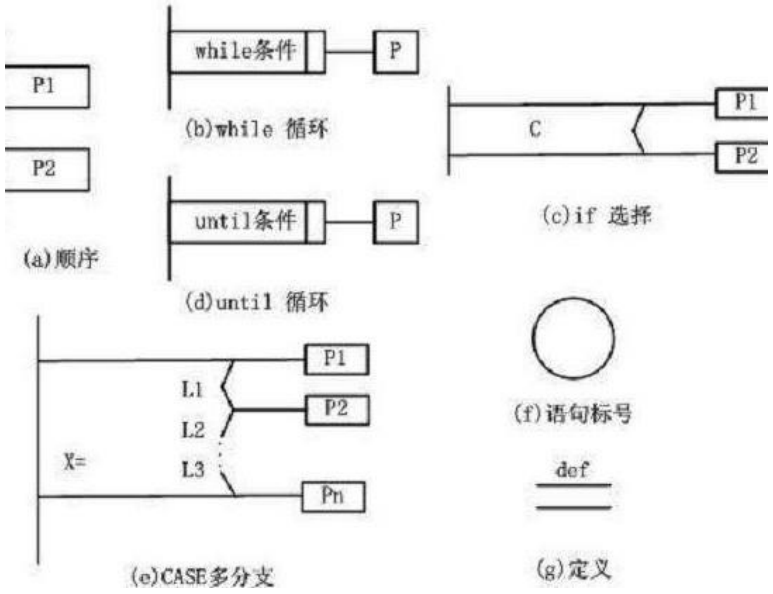


程序流程图示例



流程图与控制结构：其他程序流程表示方法

- ✓ N-S图（盒图）
- ✓ PAD图
- ✓ 伪代码。。。

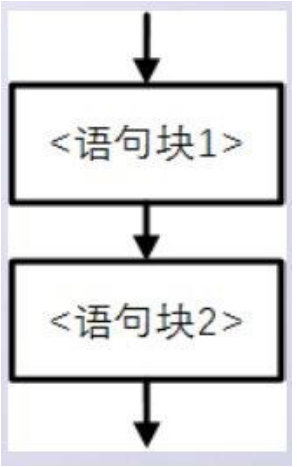


算法 2 基于多智能体深度强化学习的训练过程

1. **构建初始的近端策略优化网络，按照给定的数据设定初始参数**
2. **循环：从t=** 1**到**T
3. **循环：从智能体k=** 1**到M**
4. **以概率**ε**随机选取一个动作a，或者以** $1-\varepsilon$**的概率选取根据当前策略**π_θ**计算出最优动作a**
5. **执行动作a并计算奖励**$Q(s_t, a)$
6. **将状态转移到下一个状态**s_{t+1}
7. **选取下一个虚拟网络功能**f_{i+1}
8. **存储状态转移信息**$(s_t, a, U(s_{t+1}, a))$
9. **计算奖励目标函数：**
10. $J^{\theta^k}(\theta) = \sum_{(s_t,a_t)} min(\frac{\pi_\theta(s,a)}{\pi_{\theta^k}(s,a)} R(s_t, a_t),\ clip(\frac{\pi_\theta(s,a)}{\pi_{\theta^k}(s,a)}, 1-\varepsilon, 1+\varepsilon) R(s_t, a_t))$
11. **更新参数：**
12. $\theta_{k+1} = \text{argmax} \frac{1}{|D_k|} \sum_{\tau \in D_k} J^{\theta^k}(\theta)$
13. **结束循环**
14. **结束循环**

流程图与控制结构：程序的基本控制结构

程序由三种基本结构组成：**顺序结构、选择结构、循环结构**，这些基本结构都有一个入口和一个出口。 任何程序都由这三种基本结构组合而成。

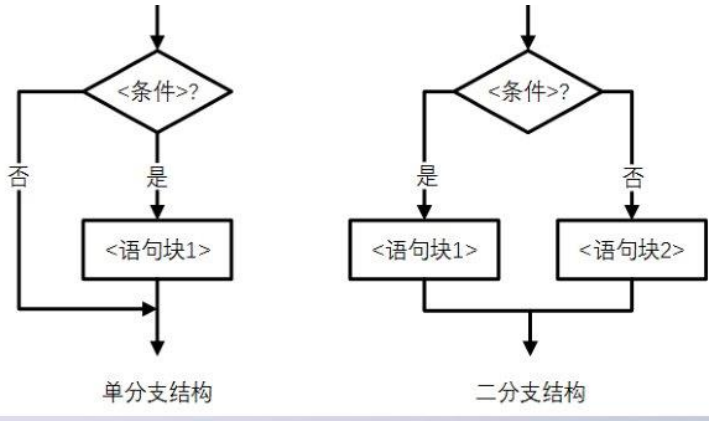


顺序结构

顺序结构是程序的基础， 但单一的顺序结构不可能解决所有问题。

选择结构

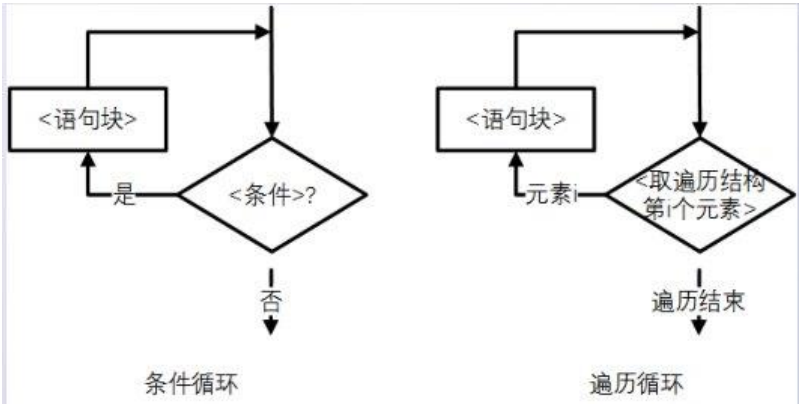
是程序根据条件判断结果而选择不同执行路径的一种运行方式，包括单分支结构和二分支结构。 由二分支结构可组合形成多分支结构。



流程图与控制结构：程序的基本控制结构

循环结构

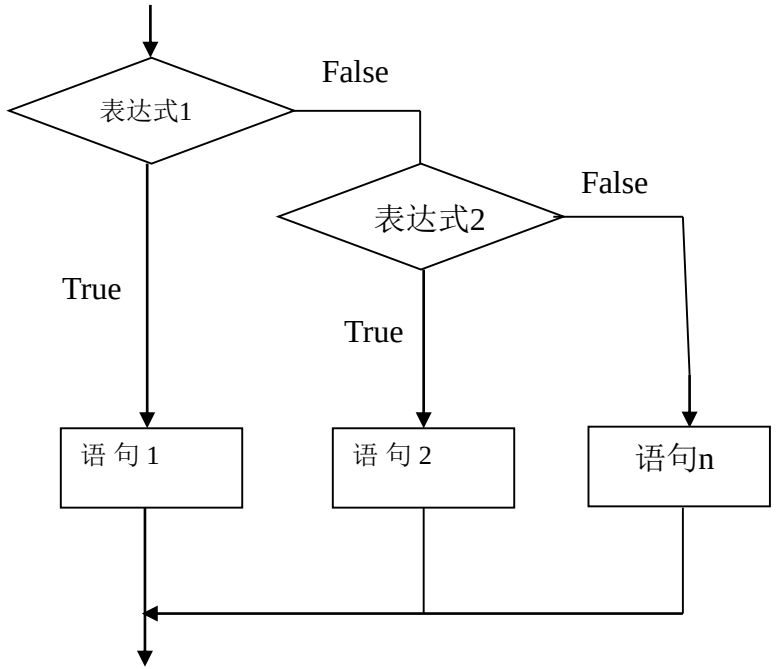
是程序根据条件判断结果向后反复执行的一种运行方式， 根据循环体触发条件不同， 包括条件循环和遍历循环结构。



- ✓ 条件语句用来判断给定的条件是否满足，由关键字和条件表达式构成。此时，程序不再是顺序依次执行，条件不满足将跳过对应代码块。
- ✓ 一般来说，条件表达式是由条件运算符和相应的操作数据构成，Python中所有合法的表达式都可以作为条件表达式。条件表达式的值只要不是**False、0、空值（None）、空列表、空集合、空元组、空字符串**等，其它均等同于True。

选择结构：IF语句

- ✓单分支：if
- ✓双分支：if-else
- ✓多分支：if-elif-else
- ✓选择结构嵌套
- ✓在Python中没有switch…case语句




```
if 表达式1:
    语句块1
elif 表达式2:
    语句块2
……
else:
    语句块n
```

选择结构：单分支和二分支

单分支：if

```
score = int(input("请输入百分制成绩：  "))
if score >= 60:
   print("及格")
if score < 60:
   print("不及格")
```

二分支：if-else

```
score = int(input("请输入百分制成绩：  "))
if score >= 60:
    print("及格")
else:
    print("不及格")
```

当代码块只有一句时

```
score = int(input("请输入百分制成绩：  "))
if score >= 60: print("及格")
if score < 60: print("不及格")
```

二分支结构的简洁表达方式（三元运算符）

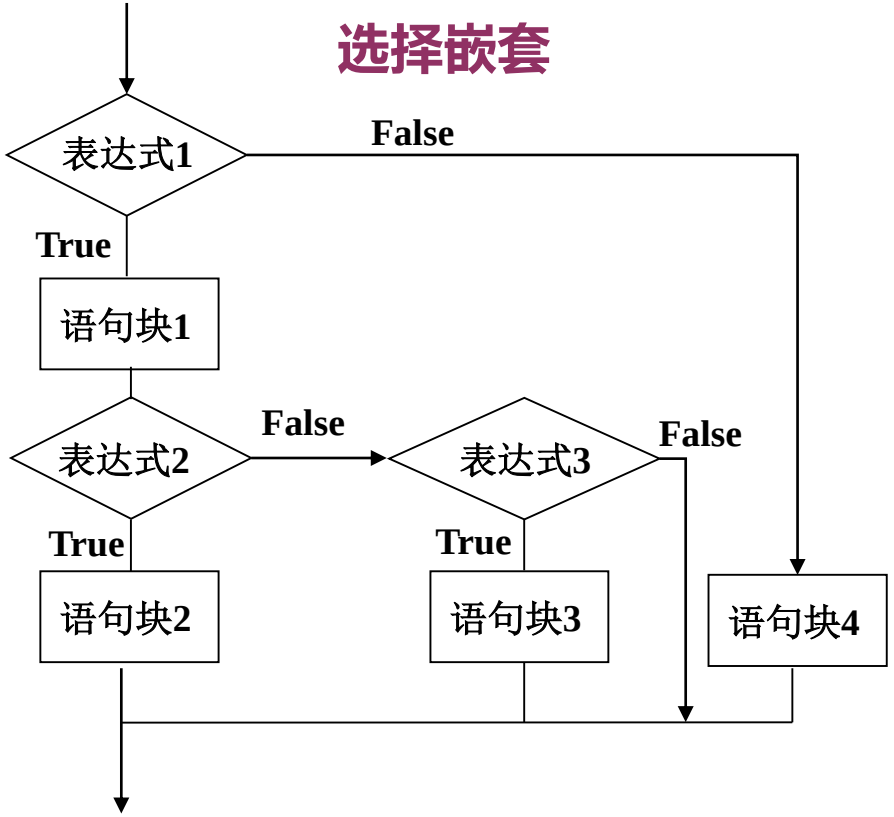
```
<表达式1> if <条件表达式> else <表达式2>

score = int(input("请输入百分制成绩：  "))
print("及格" if score >= 60 else "不及格")
```

选择结构：多分支与选择结构的嵌套

多分支：if-elif-else

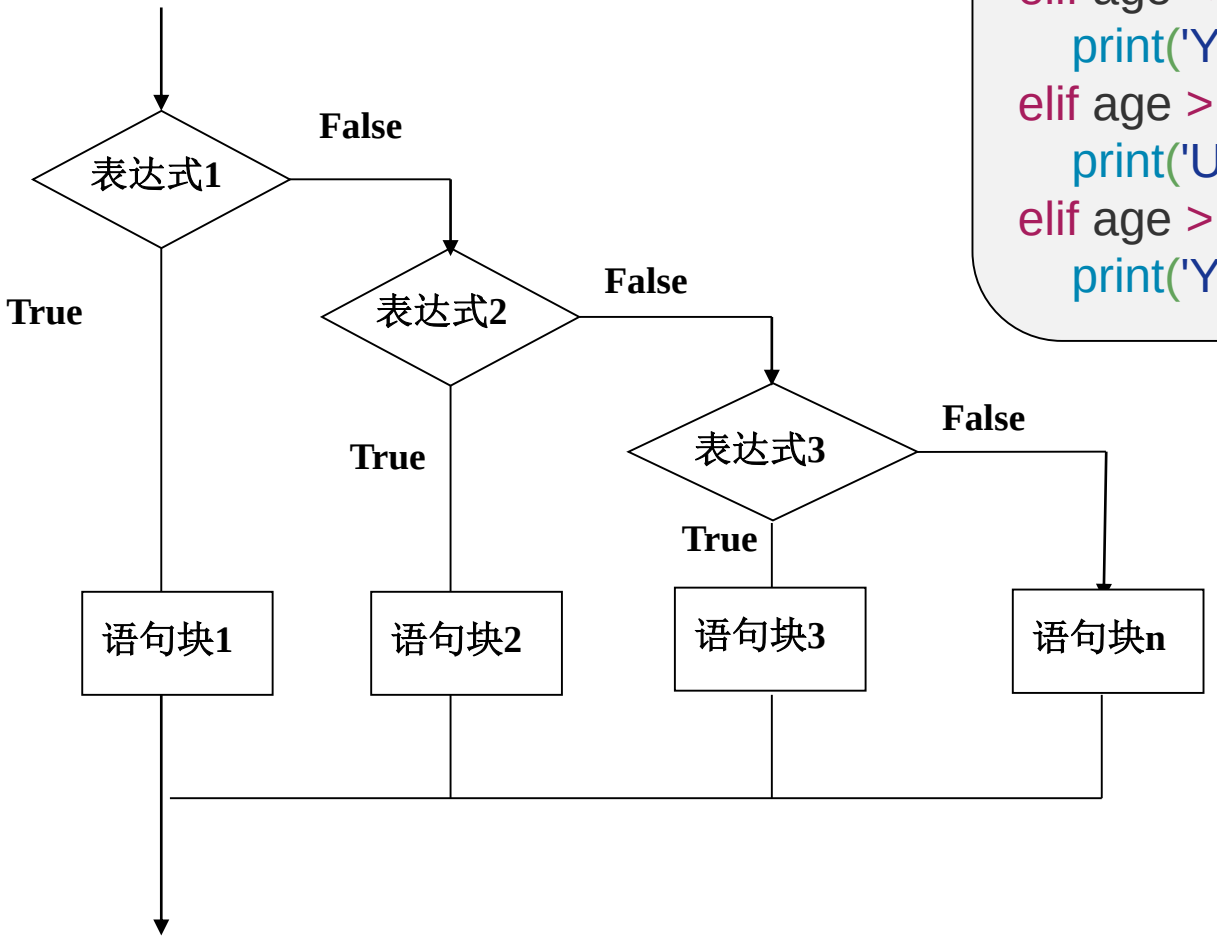
```
score = int(input("请输入百分制成绩：  "))
if score > 100:
    print("错误，成绩必须小于等于100")
elif score >= 90:
    print("成绩：A")
elif score >= 80:
    print("成绩：B")
elif score >= 70:
    print("成绩：C")
elif score >= 60:
    print("成绩：D")
elif score >= 0:
    print("成绩：E")
else:
    print("错误，成绩必须大于等于0")
```

选择嵌套




```
score = int(input("请输入百分制成绩：  "))
if 0 <= score <= 100:
    if score >= 60:
        print("及格")
else:
    print("成绩必须大于等于0且小于等于100")
```

选择结构：注意事项




```
if name == 'Alice':
    print('Hi, Alice')
elif age < 12:
    print('You are not Alice, kiddo.')
elif age > 2000:
    print('Unlike you, Alice is not an undead immortal vampire.')
elif age > 100:
    print('You are not Alice, grannie.')
```

- ✓ 只要为真，执行完对应代码块就结束。
- ✓ 如果有多个真值，则执行完第一个就结束。如输入age为3000的话，就不会走>100的分支。
- ✓ 要注意条件之间的逻辑，不要写重叠、冲突的分支，或者永远不会执行的“死代码”。

循环结构：WHILE与IF的区别

- ✓ **while**语句。只要条件为真，就执行代码块。当条件为假时，跳过代码块。
- ✓ 语法格式：

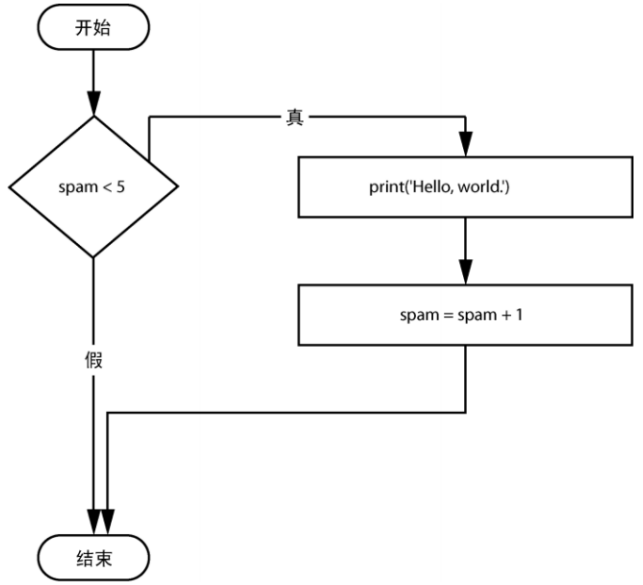
  while 条件：

  代码块1

- ✓ 条件检查，总是在每次迭代开始时候进行。

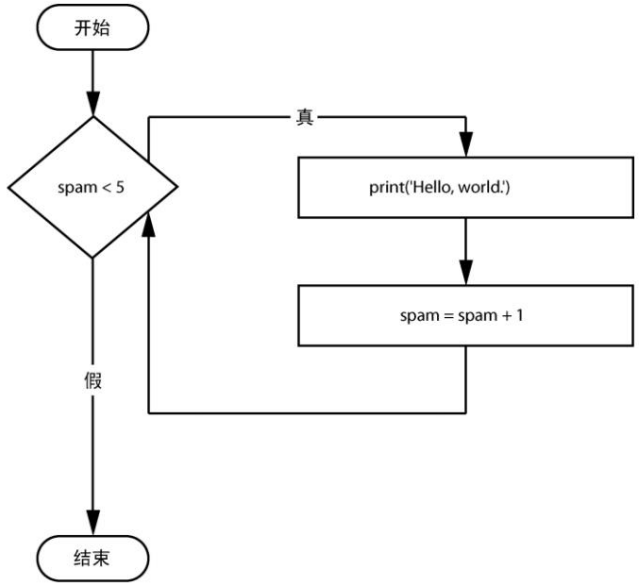
if语句

```
spam = 0
if spam < 5:
    print('Hello, world.')
    spam = spam + 1
```



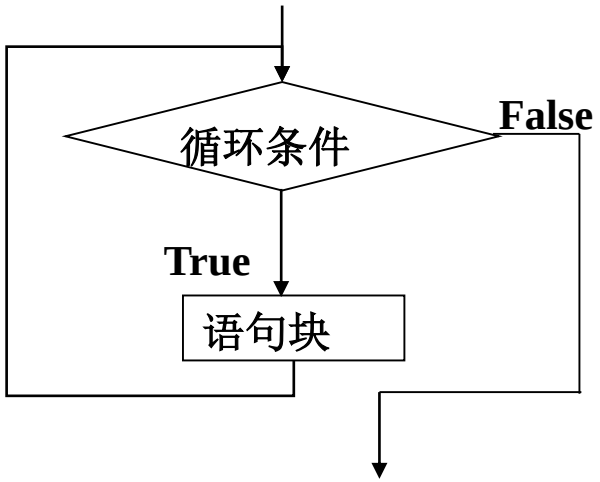
while语句

```
spam = 0
while spam < 5:
    print('Hello, world.')
    spam = spam + 1
```



循环结构：WHILE与FOR循环

```
while <条件表达式>:
    <语句块>

for <循环变量> in <可迭代对象>:
    <语句块>
```



else扩展

```
while <条件表达式>:
    <语句块1>
else:
    <语句块2>


for <循环变量> in <可迭代对象>:
    <语句块1>
else:
    <语句块2>
```

循环结构

range()是Python的内置函数，返回一个可迭代对象
可迭代对象可以是正序也可以逆序

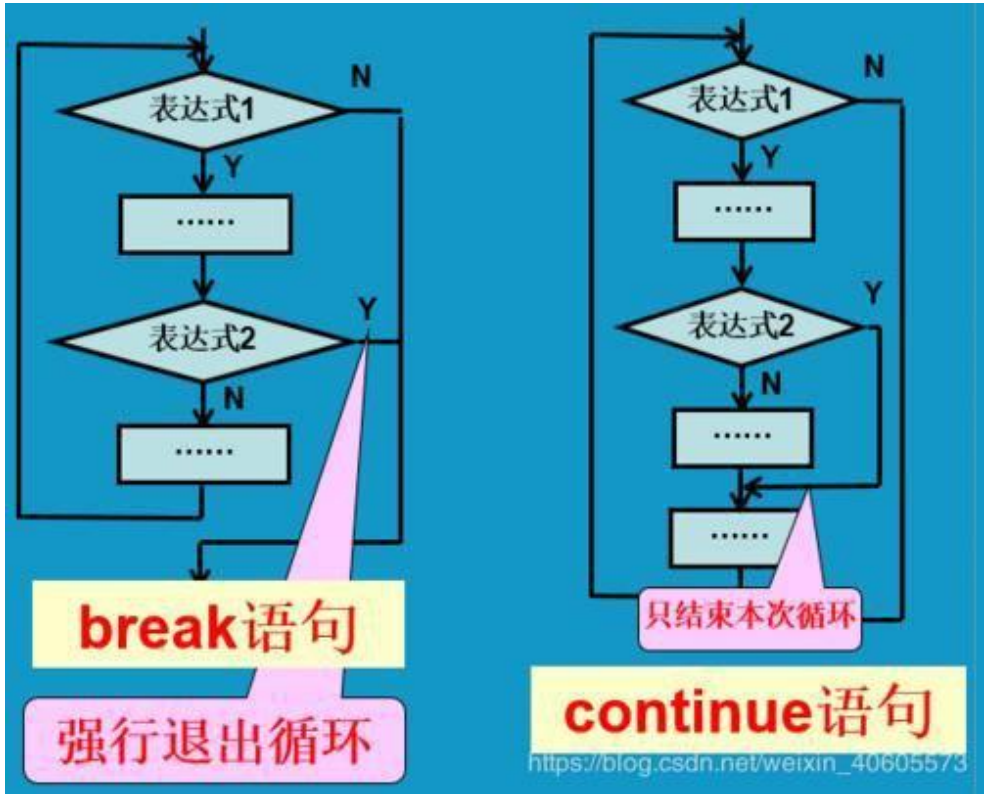
range(start, stop, step)

```
sum = 0
for i in range(1,11):
    sum = sum + i
print(sum)

sum = 0;
for(i=1;i<11;i++)
    sum = sum + i;
printf(sum);
```

```
sum = 0
for i in range(1,11):
    sum = sum + I
    i += 2   # 无实际效果
print(sum)
```

```
# 输出一个字符串中的所有的大写字符
s = input("请输入一个字符串： ")
for letter in s:
    if letter.isupper():
        print(letter)

# 输出一个列表对象中的所有元素内容
languages = ["C", "C++", "Java", "Python", 1]
for x in languages:
    print(x)

# 输出一个整数列表中所有的偶数
list = [8, 9, 1, 2, 4, 3, 6, 7, 0]
print(list)
print([x for x in list if x % 2 == 0])
```

循环结构：BREAK与CONTINUE



使用**break**语句

完全中止循环

```
for i in range(10):
    print("i = ", i, end=', ')
    for j in range(10):
        if j == 5:
            break
        print(j, end='  ')
    print()
```

使用**continue**语句

直接跳到下一次循环

```
for i in range(10):
    print("i = ", i, end=', ')
    for j in range(10):
        if j == 5:
            continue
        print(j, end=' ')
    print()
```

循环结构：ELSE扩展

只要没有**break**或者**return**， 都会执行**else**语句

```
# break、continue对else分支的影响
for j in range(10):
    if j == 5:
        break
    print(j, end=' ')
else:
    print("循环正常结束")

for j in range(10):
    if j == 5:
        continue
    print(j, end=' ')
else:
    print("循环正常结束")
```

```
import random

point = random.randint(1, 6)
count = 1
while count <= 3:
    guess = int(input("请输入您的猜测： "))
    if guess > point:
        print("您的猜测偏大")
    elif guess < point:
        print("您的猜测偏小")
    else:
        print("恭喜您猜对了 ")
        break
    count = count + 1
else:
    print("很遗憾， 三次全猜错了！ ")
```

综合实例

Collatz conjecture，称为奇偶归一猜想、3n + 1猜想：对于每一个正整数，如果它是奇数，则对它乘3再加1，如果它是偶数，则对它除以2，如此循环，最终都能够得到1。

比如：

10：5、16、8、4、2、1

```
n = int(input("请输入一个正整数： "))
m = 5
while n > 1:
    if n % 2 == 0:  #除以2的余数为0，代表是偶数
            n = n // 2  #整除，为了避免'x.0'的浮点数
       else:
        n = 3 * n + 1
    m = m - 1       #m变量为了控制一行最多输出4个值
       if m>0:
        print(str(n)+'\t',end='') #end用''，避免自动换行
            #'\t'，为了用tab符简单实现数字位数不同的整齐排列
       else:
        print('\n' + str(n)+'\t' ,end='')#每4个换行一次
            m=4
```

```
请输入一个正整数：  >? 11
34   17   52   26
13   40   20   10
5    16   8    4
2    1
```

综合实例

如果将Python的**合法标识符**改为只允许采用**大写字母、小写字母、数字和下划线**且标识符的首字符**不能是数字**。现在需要写一段程序来判断给定的标识符是否合法。

程序的输入要求为：第一行为一个正整数n，后边是n行，每行一个字符串。

输出为n行，与输入的n行字符串对应，如果输入的字符串是合法的标识符则输出yes，否则输出no。

```
# 检查合法标识符

n = int(input()) #读入n并转换为整数
for i in range(n): #n次循环
    identifier = input() #读入一个标识符
    for ch in identifier: #遍历读入的标识符
        if ch.isalpha() or ch.isdigit() or ch =='_':
            continue #如果字符为字母、数字和下划线则继续
        break #否则该标识符含有非法字符，循环可以提前结束
    else: #如果循环正常结束，说明没有非法字符
        if identifier[0].isdigit():
            print('no') #如果第一个字符为数字，则标识符非法
        else:
            print('yes') #否则，标识符合法
        continue #继续读入下一个标识符
    print('no') #循环没有正常结束，一定含有非法字符
```

多选题 1分

设置

下面哪些说法正确?

A for i in range(5)，代表了循环4遍

B for i in range(1,-5,-1)，代表循环6遍

C 执行完break语句，循环将跳出并执行else语句

D for x in 'abcd'，代表循环4遍

提交

多选题 1分

设置

下列哪些说法正确？

A if-elif语句，只要有一个条件为真，执行完即跳出整个语句

B if-elif语句，如果条件设置不当，有可能某些分支永远不会执行

C 遇到continue，while循环将继续执行完本次循环的剩余语句

D while循环语句的判断条件，可以永远等于True

提交

填空题 3分

设置

输出一个斐波拉契数列（<100）
1，1，2，3，5，8，11……

```
a,b =0, [填空1]
while b<[填空2]:
    print(b，  end=‘, ’)
    a, b = [填空3]
```

正常使用填空题需3.0以上版本雨课堂

作答

三 流程控制

- 流程图与程序结构
- 选择控制结构
- 循环控制结构
- 综合实例

谢谢