Iteration 迭代迴圈-while

2020年8月20日　　下午 03:47

conditional的無限執行版 或稱為 **loop**

while

[
while conditions :
　　statements

while is nothing but an **if** that repeats
The statements in a while block are repeated if the condition is satisfied

while 重覆執行statement 直到condition 不符合condition

要讓此迴圈結束，其 condition 和 statement 要有一定關係，即會有一個loop counter

EX:

[

```
sum = 0
i = 1

while i <= 10 :
        sum += i
        i += 1

print(sum)
```

]

[

```
exit = input("Press y or Y to exit: ")

while not (exit =="y" or exit == "Y"):
        exit = input("Press y or Y to exit: ")
```

]

重要練習題目 EX:

Given an integer n , is n =2^K for some integer k ≥ 0

[

```
n = int(input())
k = 0
m = 1

while n > m :
        m *= 2
        k += 1
        #print
if m == n :
        print(n, "is 2 to the power of", k)
```

]

輸入的n若大於1，為不論何數，m從1開始乘上2，直到m>n 或 m=n

m>n時不會有作為，代表m不為2的平方值

m=n時將會顯示n, "is 2 to the power of", k，代表n為2的K次方

而迴圈也會有無限迴圈(infinte)之問題

此時大部分是因為程式撰寫時有logical error

[

```
n = int(input())
k = 0
m = 1

while n != m :
        m *= 2
        k += 1
        #print
if m == n :
        print(n, "is 2 to the power of", k)
```

]

此時輸入了一個非2平方數，此時會不停做statement 視窗無法關閉

break

當在迴圈執行中之想要特定條件進行跳出整個迴圈block，不執行下方同等statement
可使用break (須配合if else …)

***要注意在break中會跳出迴圈執行下一步，而如不符合while的conditions 也會跳出迴圈**
故有時在程式停止之時無法知道之道是因為何者而跳出迴圈

continue

當想要在迴圈中之特定條件進行返回前面上方之while迴圈條件進行檢查 不執行下方同等statement
可使用continue(須配合if else…)

EX:
和上方同樣題目

[

```
n = int(input())
m = n
k = 0

while m > 1 :
        if m %2 != 0 :
                break
        m //= 2
        k += 1

if m == 1 :
        print(n, "is 2 to the power of", k)
```

]

此寫法配合break可更快速將非2倍數進行淘汰

[

```
n = int(input())
m = n
k = 0

while m > 1 :
        if m %2 != 0 :
                continue
        m //= 2
        k += 1

if m == 1 :
        print(n, "is 2 to the power of", k)
```

]

上面式子有logical error之問題(不符合要求)

EX:

[

```
exit = input("Press Y or y to exit: ")

while not ( exit == "y" or exit =="Y"):
        exit = input("Press Y or y to exit: ")
```

]

可改寫成以下以減少potential inconsistency

[

```
while True:
        exit = input("Press Y or y to exit: ")
        if exit == "y" or exit == "Y" :
                break
```

]

while True等於無論如何

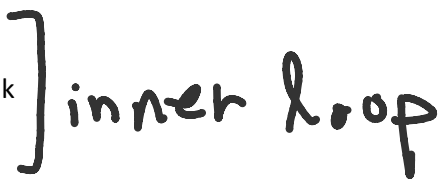
而break及continue皆會在同一個迴圈中進行跳出或返回 不會一次跳出兩個迴圈

ex:

nested-while例子

[

```
a = 1
b = 1
while a <= 10 :
        while b <= 10 :
                if b == 5 :
                        break
                print(a*b)
                b += 1
        a += 1
print(a, b)
```



]

此nested-while分為內層及外層迴圈

一開始a=1 b=1故在內層迴圈中執行b+=1直到b=5

在b為5的情況之下，開始跳出內層迴圈，此時在外層迴圈之中繼續執行 a +=1，

再重複一次外層迴圈判斷 a<=10 再進到 內層迴圈判斷b<=10 而 if b==5 又跳出執行a+=1

如此重覆

故a會在外層迴圈中加到11為止，b就停在5