04 流程控制應用

本節課針對流程控制部分，運用前兩次課程內容進行進一步練習，同時補充幾個常用指令。

#亂數產生器

import random                 #載入random模組

random.seed()       #初始化隨機數，若()內設定數字，無論隨機幾次都會產生固定結果

r = random.random()  #隨機產生0-1(不含1)的浮點數

print(r)

r = random.uniform(1, 50)     #隨機產生1-50的浮點數

print(r)

r = random.randint(1,50)      #隨機產生1-50的整數

print(r)

r = random.randrange(1,50,2)  #隨機產生1-49的整數(奇數)

print(r)

r = random.choice([5,8,7,4,66,33,99,45])

print(r)        #從串列中隨機產生1個數字

r = random.choices([5,8,7,4,66,33,99,45],k=4)

print(r)         #從串列中隨機產生指定個數數字(會重複)

r = random.sample([5,8,7,4,66,33,99,45],k=4)

print(r)         #從串列中隨機產生指定個數數字(不重複)

1. 亂數產生器

使用亂數產生器產生10個亂數。

import random

for i in range(1,11):

    randNum = random.randint(1,100)

    print('%4d'%(randNum),end = '')

輸出結果

89 21 1 43 44 48 92 52 29 8

若要檢查產生的亂數中有多少個是偶數或是奇數，此時就必須藉助選擇敘述來判斷。

import random

even = 0

odd = 0

for i in range(1,11):

    randNum = random.randint(1,100)

    print(randNum,end = ' ')

    if randNum %2 ==0:

        even +=1

    else:

        odd +=1

print('\n even = %d, odd = %d'%(even, odd))

輸出結果

47 53 82 21 53 1 84 35 65 84

even = 3, odd = 7

這支程式和上面相比多了 if – else 條件判斷以及列印奇偶數的部分

如果程式在使用時並不需要載入random所有功能，可指定載入使用到的模組：

from random import randint

用上述指令替代import random，在執行randNum = random.randint(1,100) 這行指令時，改為

randNum =randint(1,100)

如果需要同時載入多個模組，使用逗號分隔：

from random import randint, choice

以下利用亂數產生器產生100個亂數，然後判斷這些亂數有多少個屬於3、5、7的倍數，和3、5、7倍數以外的數字。

import random

times3 = 0

times5 = 0

times7 = 0

others = 0

for i in range(1,101):

    flag = False

    randNum = random.randint(1,100)

    print(randNum,end =' ')

    if randNum % 3 == 0:

        times3 += 1

        flag = True

    if randNum %5 == 0:

        times5 += 1

        flag = True

    if randNum %7 == 0:

        times7 += 1

        flag = True

    if flag == False:

        others += 1

print('\n times3 = %d, times5 = %d, times7 = %d'%(times3,times5,times7))

print('others = %d'%(others))

輸出結果

75 55 46 9 55 37 54 86 53 81 19 55 70 39 90 25 25 89 35 65 73 76 89 33 5 60 51 11 51 84 25 47 29 99 20 86 99 25 26 18 79 19 58 75 47 77 9 81 4 62 49 17 62 48 87 81 46 9 45 89 91

9 48 81 53 98 22 19 14 49 62 65 20 47 94 70 69 10 40 91 30 29 86 53 54 84 2 55 92 38 99 66 43 72 33 16 21 9 11 25

times3 = 35, times5 = 25, times7 = 13

others = 39

這個程式也是使用選擇敘述來判斷所產生亂數是否為3、5、7的倍數，注意這個程式題目僅要求是否為3、5、7倍數，因此如15、35這些數字會被重複計算，因此最後面個數加總和可能超過100。

import random

times3 = 0

times5 = 0

times7 = 0

others = 0

count = 1     #新增1個用於迴圈記數的變數

for i in range(1,101):

    flag = False

    randNum = random.randint(1,100)

    #進行排版 每列10個數字 每個數字佔4格

    if count % 10 !=0:

        print('%4d'%(randNum),end =' ')

    else:

        print('%4d'%(randNum))

    count +=1

    if randNum % 3 == 0:

        times3 += 1

        flag = True

    if randNum %5 == 0:

        times5 += 1

        flag = True

    if randNum %7 == 0:

        times7 += 1

        flag = True

    if flag == False:

        others += 1

print('\n times3 = %d, times5 = %d, times7 = %d'%(times3,times5,times7))

print('others = %d'%(others))

這支程式將前面程式排版做一修正，以count這個變數配合if條件式，設定每列印出10個數字。列印時以%4d來控制每個數字佔用4個字元空間，這樣就能印出10 \* 10整齊的數字排版。

輸出結果

26 17 63 34 99 30 100 79 23 81

54 63 97 97 37 29 45 68 50 92

11 77 27 44 88 83 91 53 72 89

26 25 45 13 87 99 88 54 10 17

49 2 17 9 78 97 12 1 48 27

98 51 41 8 66 51 76 81 49 43

35 14 1 3 21 25 20 25 83 100

1 48 74 2 49 10 16 28 47 70

9 10 51 28 23 37 86 4 25 89

16 83 90 75 76 94 67 37 5 81

如果我們要在計算時，詳列同為35、37、57倍數的個數，以及僅為3、5、7倍數的個數共有多少個，讓總個數加總等於100。可參考方式如下：

import random

times35 = 0; times37 = 0;times57 = 0

times3 = 0; times5 = 0; times7 = 0

others = 0

count = 0

for i in range(1,101):

    randNum = random.randint(1,100)

    print(f"{randNum:4}",end ='')

    count += 1

    if count %10 == 0:

        print()

    if randNum %3==0 and randNum %5==0:

        times35 += 1

    elif randNum %3==0 and randNum %7==0:

        times37 += 1

    elif randNum %5==0 and randNum %7==0:

        times57 += 1

    elif randNum %3 == 0:

        times3 += 1

    elif randNum %5 == 0:

        times5 += 1

    elif randNum %7 == 0:

        times7 += 1

    else :

        others += 1

print(f'3、5倍數有{times35}個，3、7倍數有{times37}個，5、7倍數有{times57}個')

print('3的倍數有 %d個， 5的倍數有 %d個， 7的倍數有 %d個'%(times3,times5,times7))

print('others = %d'%(others))

print(times35+times37+times57+times3+times5+times7+others)  #驗證用(可省略)

輸出結果：

60 23 5 54 42 38 45 63 95 58

64 65 23 22 51 94 21 82 72 17

63 16 24 56 15 20 48 77 4 4

46 23 73 100 11 20 53 50 54 11

64 85 7 81 91 16 25 1 8 88

40 86 48 88 95 82 82 5 47 7

68 51 34 7 85 45 11 92 1 95

9 32 87 43 6 50 37 27 69 24

82 13 54 39 97 67 3 41 62 56

10 58 5 10 59 54 40 93 22 9

3、5倍數有4個，3、7倍數有4個，5、7倍數有0個

3的倍數有 21個， 5的倍數有 19個， 7的倍數有 7個

others = 45

100

1. 定數迴圈與不定數迴圈

當我們對迴圈設定固定執行次數時，稱之為定數迴圈。不定數迴圈，表示事先沒有設定迴圈次數，在執行中讓使用者以其他方式中止迴圈執行。

定數迴圈例：產生60個亂數(範圍1 - 49)，每6個一列，排成6 \* 10的矩形格式

import random

count = 1

while count <= 10:

    for i in range(1,7):

        randNum = random.randint(1,49)

        print('%3d'%(randNum),end = ' ')

    print()

    count += 1

print('Over')

輸出結果

39 36 29 5 5 8

39 44 13 19 15 49

11 14 31 9 12 36

33 4 14 34 21 24

5 48 41 16 2 38

10 49 34 32 24 22

36 31 20 49 18 23

34 22 8 21 34 44

1 13 3 22 33 22

49 34 41 44 21 10

Over

這程式屬於多重迴圈，外迴圈用while來控制亂數產生的次數，以count便數來控制。在內迴圈控制每次產生幾個亂數，兩個迴圈配合就可以產生10 \* 6 共計60個亂數。

不定數迴圈例：與上例相同每次產生6個亂數，但一共要產生幾次，則讓使用者決定。

import random

again = 1

while again == 1:

    for i in range(1,7):

        randNum = random.randint(1,49)

        print('%3d'%(randNum),end = ' ')

    print()

    again = int(input("繼續請按'1'，停止請按任意數："))

print('Over')

輸出結果

23 35 39 22 2 47

繼續請按'1'，停止請按：'0'： 1

27 43 18 7 4 6

繼續請按'1'，停止請按：'0'： 1

41 30 18 4 44 48

繼續請按'1'，停止請按：'0'： 0

Over

程式使用again控制程式是否繼續產生6個亂數。以交談方式引導使用者輸入一個數值，並指定給again變數，若輸入1，則表示繼續產生6個亂數，若為0則結束迴圈執行。

1. break 與 continue敘述

迴圈內使用break敘述時，迴圈會立即中止執行，所以可以將迴圈設為無窮迴圈，然後在迴圈內部使用條件式運用break終止這個迴圈。

import random

while True:

    for i in range(1,7):

        randNum = random.randint(1,49)

        print('%3d'%(randNum),end = ' ')

    print()

    again = input("繼續請按'1'，停止請按：'0'： ")

    if again == '0':

        break

print('Over')

輸出結果

24 49 27 13 29 21

繼續請按'1'，停止請按：'0'： 1

12 30 43 49 23 37

繼續請按'1'，停止請按：'0'： 1

10 46 21 42 9 45

繼續請按'1'，停止請按：'0'： 0

Over

程式中while True: 表示此迴圈是無窮迴圈，當程式提示使用者輸入時，變數again若為0，就以break結束對應的while迴圈。

加法練習：以亂數產生器產生兩個數，分別指定給n1與n2，然後由使用者輸入數字和，數答錯，將繼續作答，若答對，則以break敘述結束迴圈的執行。

import random

n1 = random.randint(1,100)

n2 = random.randint(1,100)

while True:

    solution = n1 + n2

    answer = eval(input('%d+%d = '%(n1,n2)))

    if answer == solution:

        print('恭喜你答對了！')

        break

    else:

        print('答錯！再試一次')

print('結束')

輸出結果

2+74 = 76

恭喜你答對了！

結束

36+88 = 102

答錯！再試一次

36+88 = 99

答錯！再試一次

36+88 = 124

恭喜你答對了！

結束

當迴圈執行到continue 這行敘述時，就不會再執行continue以下的敘述，然後從迴圈起始點繼續執行下一次迴圈。

範例：列出從1 到15的數，但排除5的倍數，再將顯示出來的數相加

total = 0

n = 1

while n <= 15:

    if n % 5 ==0:

        n += 1

        continue

    print('%3d'% n, end =' ')

    total += n

    n += 1

print('\n total = %d' % total)

輸出結果

1 2 3 4 6 7 8 9 11 12 13 14

total = 90

程式透過條件判斷式，判斷是否能被5整除，若能被5整除就跳過這一次計算，這樣進行加總時，就不會把5的倍數加進去。

把上題，continue的敘述改為break看看結果：

total = 0

n = 1

while n <= 15:

    if n % 5 ==0:

        n += 1

        break

    print('%3d'% n, end =' ')

    total += n

    n += 1

print('\n total = %d' % total)

輸出結果

1 2 3 4

total = 10

程式進行到5的時候，執行break敘述，整個迴圈直接中斷不再執行。

練習題

1. 撰寫一個程式，由使用者輸入10個數字，然後找出最小值並且輸出
2. 撰寫一個程式，由使用者輸入數字，輸入的動作直到輸入值為9999結束，然後找出其最小值並且輸出。
3. 撰寫一個程式：
   1. 由使用者輸入兩個正整數a、b (a<b)
   2. 輸出從a到b(包含a 、b)之間4或9的倍數(一列輸出10個數字，欄寬為4、靠左對齊)。
   3. 輸出4或9的倍數共有幾個數。
   4. 輸出4或9的倍數加總和。
4. 撰寫一個程式：
   1. 輸入一個正整數。
   2. 將這個正整數以反轉的順序輸出。
   3. 若輸入值為0，就輸出0
5. 撰寫一個程式：
   1. 輸入代表成績的正整數。
   2. 根據分數與等級對照表，印出對應的等級。
   3. 輸入9999結束迴圈

|  |  |
| --- | --- |
| 分數 | 等級 |
| 90 ~ 100 | A |
| 80 ~ 89 | B |
| 70 ~ 79 | C |
| 60 ~ 69 | D |
| 0 ~ 59 | E |

1. 撰寫一個程式：
   1. 以迴圈方式，提供使用者反覆輸入身高與體重，直到輸入9999結束。
   2. 計算出BMI值，並列印出相對應的意義。
   3. 輸出到小數點後第二位。

BMI公式： BMI = 體重(公斤) / ()

|  |  |
| --- | --- |
| BMI | 說明 |
| < 18.5 | 過輕 |
| 18.5 - 24.9 | 正常 |
| 25.0 - 29.9 | 過重 |
| > 30 | 肥胖 |

1. 撰寫一個程式：
   1. 以迴圈方式，提供使用者反覆輸入西元年分，直到輸入9999結束。
   2. 判斷輸入年份是否為閏年。判斷規則為，每4年一閏，100年不閏，但400年也一閏
2. 撰寫一個程式：
   1. 輸入兩個西元年分，顯示兩個西元年分之間所有閏年。
   2. 每一列輸出10筆資料，並且對齊
3. 輸入3個正整數 a、b、c然後求出最大公因數
4. 完美數：一個數如果恰好等於它的因數(不含數字本身)之和，這個數就稱為[完美數]。請使用 for 迴圈設計一程式，輸入一個值，找出這個值以內的所有完美數，執行結果如範例。

執行結果

請輸入一個數： 1000

1 ~ 1000 完美數有: 6  28  496