

1. Flowchart

-*- coding: utf-8 -*-

"""

Created on Sun Sep 26 10:04:54 2021

@author: ZHY

"""

def Print_values(a,b,c):

if a > b: #python if 语句参考 <https://www.runoob.com/python3/python3-if-example.html>

if b > c:

print(a,b,c)

elif a > c:

print(a,c,b)

else: print(c,a,b)

elif a < b:

if b < c:

print(c,b,a)

elif b > c:

if a > c:

print(a,c,b)

else: print (c,a,b)

elif b < c:

print(c,b,a)

import random

a = round(random.uniform(1,100)) # 随机数生成参考

https://blog.csdn.net/qq_32618817/article/details/80583746

b = round(random.uniform(1,100))

c = round(random.uniform(1,100))

```
print("a="+str(a),"b="+str(b),"c="+str(c))
```

```
print("The result is:")
```

```
Print_values(a,b,c)
```

2. Matrix multiplication

"""

Created on Sun Sep 26 14:33:03 2021

@author: ZHY

"""

```
import numpy as np
```

```
M1 = np.random.randint(0,50,(5,10))#生成随机矩阵参考
```

```
https://blog.csdn.net/furide/article/details/103363451
```

```
M2 = np.random.randint(0,50,(10,5))
```

```
print(M1)
```

```
print(M2)
```

```
M = []
```

```
for i in range(M1.shape[0]):#矩阵中行数与列数的表达参考 https://www.cnblogs.com/Yanjy-OnlyOne/p/11298253.html
```

```
    x = []
```

```
    for k in range(M2.shape[1]):
```

```
        y = 0
```

```
        for j in range(M1.shape[1]):
```

```
            y += M1[i][j]*M2[j][k]
```

```
        x.append(y)
```

```
    M.append(x)
```

```
print(M)
```

3. Pascal triangle

-*- coding: utf-8 -*-

"""

Created on Sun Oct 10 14:31:33 2021

@author: ZHY

"""

#import numpy as np

#def Pascal_triangle(k):

j = int(k)

m = []

if j == 1:

m.append(1)

print(m)

elif j == 2:

m.append(1)

m.append(1)

print(m)

elif j > 2:

m = [1,1]

y = [1,1]

for n in range(j-2):

for i in range(n+1):

x = int(m[i]+m[i+1])

m.insert(i+1,x)

y.insert(n+1,x)

```
#     #     print(y)
```

```
#     else:
```

```
# #         print("invalid k value")
```

```
# #
```

```
# #
```

```
# #Pascal_triangle(4)
```

以下代码受题目材料中 “a formula for any entry in the triangle” 启发:

```
def Pascal_triangle(k):
```

```
    a = 1
```

```
    n=[]
```

```
    for b in range(1,k+1): #阶乘写法参考 https://www.php.cn/python-tutorials-460228.html
```

```
        a *= b
```

```
    for i in range(k+1):
```

```
        c = 1
```

```
        e = 1
```

```
        for d in range(1,i+1):
```

```
            c *= d
```

```
        for f in range(1,k-i+1):
```

```
            e *= f
```

```
        m = int(a/(c*e)) # 此处不加 int 会导致输出结果带有一位小数
```

```
        n.append(m)
```

```
    print(n)
```

```
print("Pascal_triangle(100)" + Pascal_triangle(100))
```

```
print("Pascal_triangle(200)" + Pascal_triangle(200))
```

```
j = int(input("If you want to know more, please enter the row number:"))
```

```
k = j-1
```

```
Pascal_triangle(k)
```

4. Add or double

"""

Created on Sun Oct 10 20:14:16 2021

@author: ZHY

"""

```
import random
```

```
x = int(random.randint(1,100))
```

```
print("The random value is:"+str(x))
```

```
i = 0
```

```
if x != 1 or 2 or 3:
```

```
    while x >= 2:
```

```
        if x % 2 == 0:
```

```
            i += 1
```

```
            x = x/2
```

```
        else:
```

```
            i += 1
```

```
            x = x-1
```

```
#     if x >= 2:
```

```
#         i += 1
```

```
#         x = x/2
```

```
#         continue
```

```
# elif x % 2 != 0:
```

```
#     if x >= 1:
```

```
#         i += 1
```

```
#         x = x-1
```

```

#         continue

# while x % 2 != 0:
#     if x >= 1:
#         i += 1
#         x = x-1
# while x % 2 == 0:
#     if x >= 2:
#         i += 1
#         x = x/2
# else:
#     x = x-1
#     i += 1
#     while x % 2 == 0:
#         if x >= 2:
#             i += 1
#             x = x/2
#     else:
#         x = x-1
#         i += 1
#         while x % 2 == 0:
#             if x >= 2:
#                 i += 1
#                 x = x/2
#             else:
#                 x = x-1
#                 i += 1
print(i)
elif x == 1:

```

```
    print("0")  
elif x == 2:  
    print("1")  
else:  
    print("2")
```


5. Dynamic programming

FINAL-5.1

import random

i = random.randint(1, 101)

def Find_expression(i):

num = "1 2 3 4 5 6 7 8 9"

n = ["+", "-", " "]

k = 0

j = 0

for a in range(3):

num = num[:15] + n[a] + num[16:]

for b in range(3):

num = num[:13] + n[b] + num[14:]

for c in range(3):

num = num[:11] + n[c] + num[12:]

for d in range(3):

num = num[:9] + n[d] + num[10:]

for e in range(3):

num = num[:7] + n[e] + num[8:]

for f in range(3):

num = num[:5] + n[f] + num[6:]

for g in range(3):

num = num[:3] + n[g] + num[4:]

for h in range(3):

num = num[:1] + n[h] + num[2:]

print(num)

j += 1 #测试总个数是否为 3^8

print(j)

m = num.replace(" ", "")

print(m)

`x = eval(m)` #计算字符串形式的数学运算参考

<https://blog.csdn.net/llb19900510/article/details/109527054>

```
# print(x)
```

```
if x == i:
```

```
    print(str(m) + "=" + str(x))
```

```
    k += 1
```

```
#print(k)
```

```
i = random.randint(1, 101)
```

```
Find_expression(i)
```

#5.2

```
import matplotlib.pyplot as plt
```

```
Y = []
```

```
X = []
```

```
def Total_solutions(i):
```

```
    num = "1 2 3 4 5 6 7 8 9"
```

```
    n = ["+", "-", " "]
```

```
    k = 0
```

```
    # y = []
```

```
    # j = 0
```

```
    for a in range(3):
```

```
        num = num[:15] + n[a] + num[16:]
```

```
        for b in range(3):
```

```
            num = num[:13] + n[b] + num[14:]
```

```
            for c in range(3):
```

```
                num = num[:11] + n[c] + num[12:]
```

```
                for d in range(3):
```

```
                    num = num[:9] + n[d] + num[10:]
```

```
                    for e in range(3):
```

```
                        num = num[:7] + n[e] + num[8:]
```

```

for f in range(3):

    num = num[:5] + n[f] + num[6:]

    for g in range(3):

        num = num[:3] + n[g] + num[4:]

        for h in range(3):

            num = num[:1] + n[h] + num[2:]

        # print(num)

#
    j += 1    #测试总个数是否为 3^8

# print(j)

    m = num.replace(" ", "")

    # print(m)

    x = eval(m)#计算字符串形式的数学运算参考
https://blog.csdn.net/llb19900510/article/details/109527054

    # print(x)

    if x == i:

        # print(str(m) + "=" +str(x))

        k += 1


Y.append(k)

X.append(i)


for i in range(1,101):

    Total_solutions(i)

print(X)

print(Y)


plt.plot(X,Y,marker='o',linestyle='dashed')

plt.show()

print("based on the plot,1 and 47 yield the maximum Total_solutions(26) and 90 yields the
minimum of Total_solutions(6)")

```