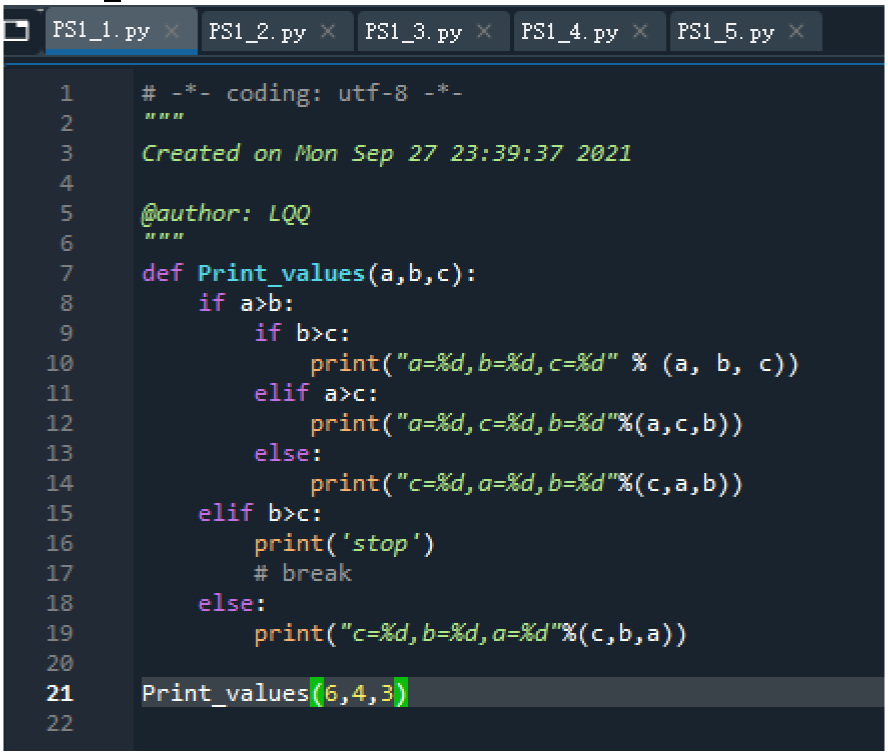
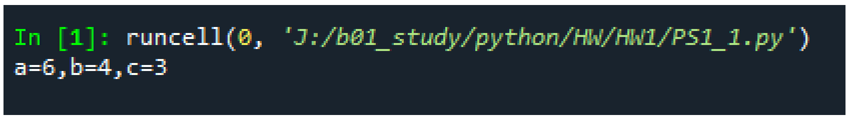
1. PS1_1

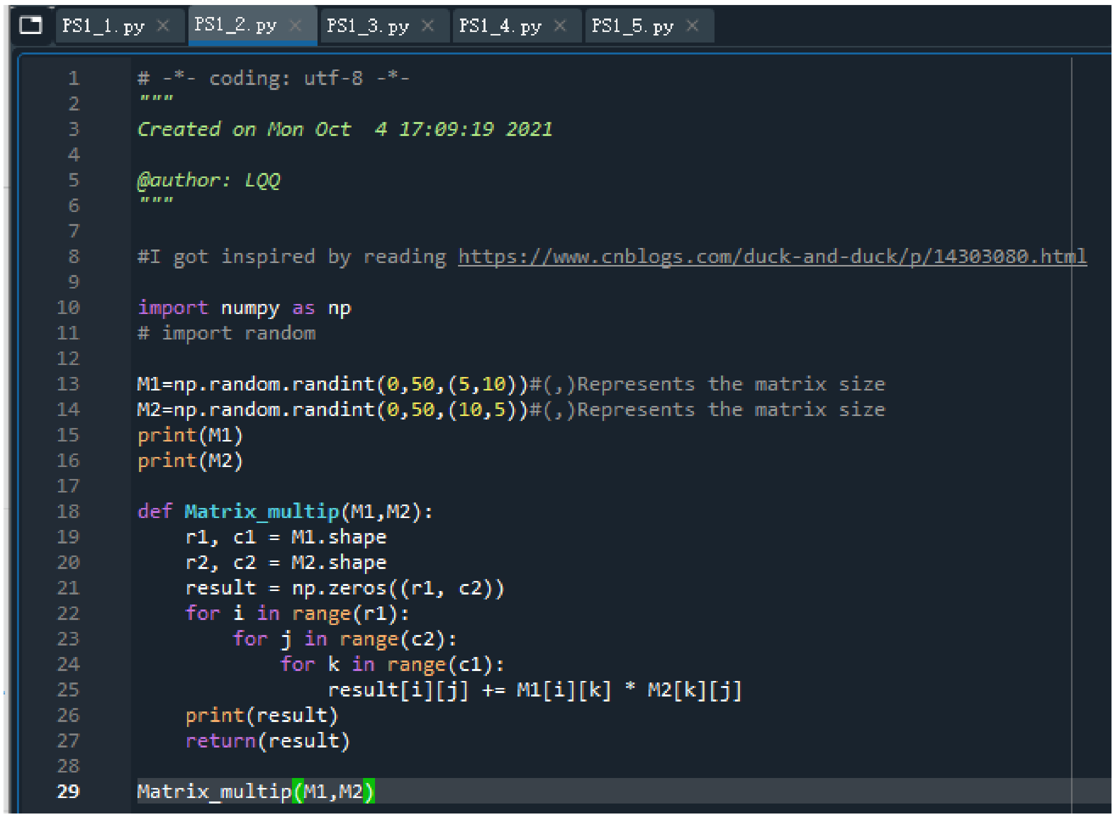


```python
# -*- coding: utf-8 -*-
"""
Created on Mon Sep 27 23:39:37 2021

@author: LQQ
"""
def Print_values(a,b,c):
    if a>b:
        if b>c:
            print("a=%d,b=%d,c=%d" % (a, b, c))
        elif a>c:
            print("a=%d,c=%d,b=%d"%(a,c,b))
        else:
            print("c=%d,a=%d,b=%d"%(c,a,b))
    elif b>c:
        print('stop')
        # break
    else:
        print("c=%d,b=%d,a=%d"%(c,b,a))

Print_values(6,4,3)

```

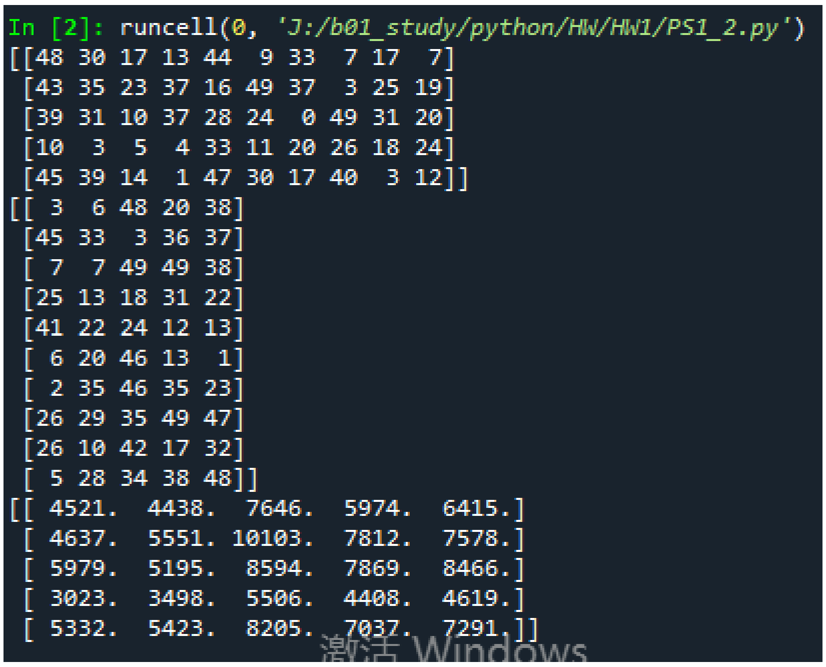


```
In [1]: runcell(0, 'J:/b01_study/python/HW/HW1/PS1_1.py')
a=6,b=4,c=3
```

2. PS1_2

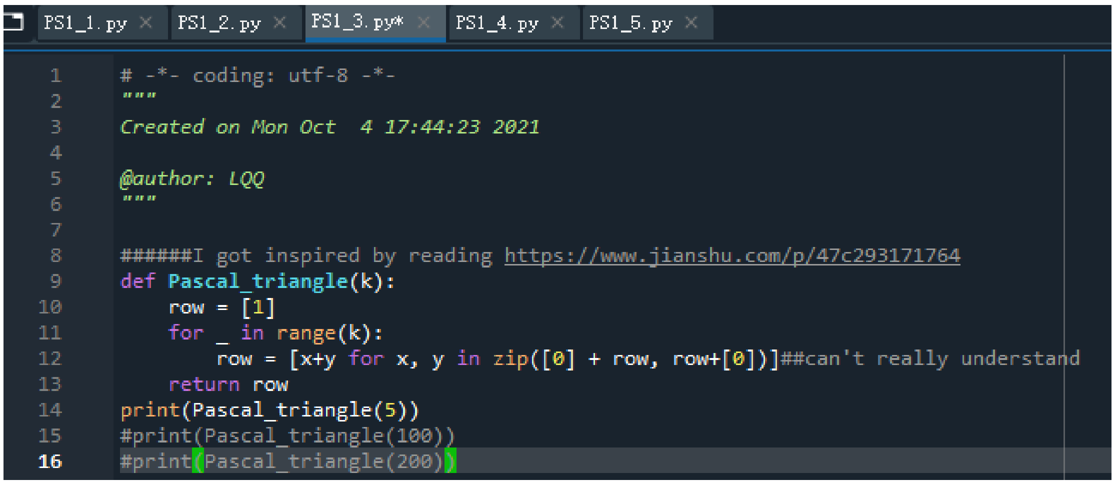


```python
# -*- coding: utf-8 -*-
"""
Created on Mon Oct  4 17:09:19 2021

@author: LQQ
"""

#I got inspired by reading https://www.cnblogs.com/duck-and-duck/p/14303080.html

import numpy as np
# import random

M1=np.random.randint(0,50,(5,10))#(,)Represents the matrix size
M2=np.random.randint(0,50,(10,5))#(,)Represents the matrix size
print(M1)
print(M2)

def Matrix_multip(M1,M2):
    r1, c1 = M1.shape
    r2, c2 = M2.shape
    result = np.zeros((r1, c2))
    for i in range(r1):
        for j in range(c2):
            for k in range(c1):
                result[i][j] += M1[i][k] * M2[k][j]
    print(result)
    return(result)

Matrix_multip(M1,M2)
```

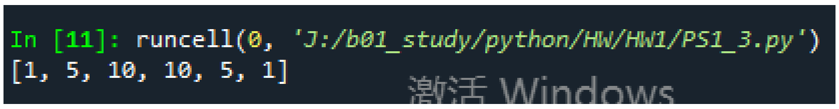


```
In [2]: runcell(0, 'J:/b01_study/python/HW/HW1/PS1_2.py')
[[48 30 17 13 44  9 33  7 17  7]
 [43 35 23 37 16 49 37  3 25 19]
 [39 31 10 37 28 24  0 49 31 20]
 [10  3  5  4 33 11 20 26 18 24]
 [45 39 14  1 47 30 17 40  3 12]]
[[ 3  6 48 20 38]
 [45 33  3 36 37]
 [ 7  7 49 49 38]
 [25 13 18 31 22]
 [41 22 24 12 13]
 [ 6 20 46 13  1]
 [ 2 35 46 35 23]
 [26 29 35 49 47]
 [26 10 42 17 32]
 [ 5 28 34 38 48]]
[[ 4521.  4438.  7646.  5974.  6415.]
 [ 4637.  5551. 10103.  7812.  7578.]
 [ 5979.  5195.  8594.  7869.  8466.]
 [ 3023.  3498.  5506.  4408.  4619.]
 [ 5332.  5423.  8205.  7037.  7291.]]
```

3. PS1_3

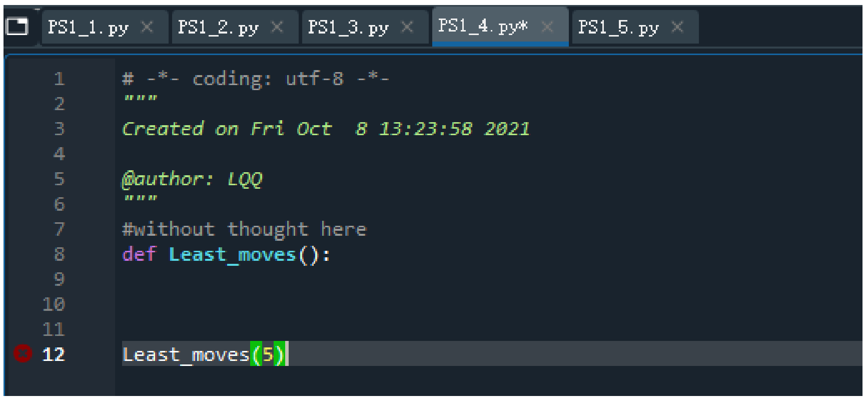


```python
# -*- coding: utf-8 -*-
"""
Created on Mon Oct  4 17:44:23 2021

@author: LQQ
"""

######I got inspired by reading https://www.jianshu.com/p/47c293171764
def Pascal_triangle(k):
    row = [1]
    for _ in range(k):
        row = [x+y for x, y in zip([0] + row, row+[0])]##can't really understand
    return row
print(Pascal_triangle(5))
#print(Pascal_triangle(100))
#print(Pascal_triangle(200))
```



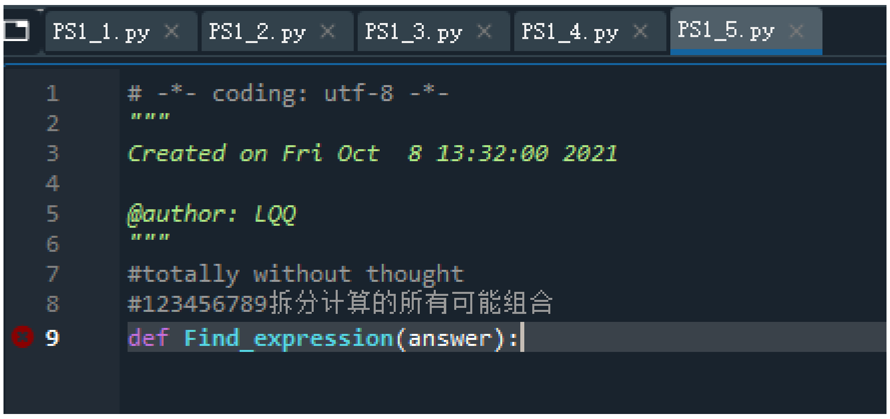
```
In [11]: runcell(0, 'J:/b01_study/python/HW/HW1/PS1_3.py')
[1, 5, 10, 10, 5, 1]
```

4. PS1_4



```python
# -*- coding: utf-8 -*-
"""
Created on Fri Oct  8 13:23:58 2021

@author: LQQ
"""
#without thought here
def Least_moves():



Least_moves(5)
```

5. PS1_5



```python
# -*- coding: utf-8 -*-
"""
Created on Fri Oct  8 13:32:00 2021

@author: LQQ
"""
#totally without thought
#123456789拆分计算的所有可能组合
def Find_expression(answer):
```