

第一题:

本题目的是对随机取得的 a, b, c 的值进行排序, 用 if 条件语句把每一种大于或小于的可能性都列出来, 就可以最后得到由大到小的排序。题目没有规定 a, b, c 的取值范围, 我就用了 random.random 取了小数, 把这一条语句改一下也可以得到其他范围的 a, b, c 的排序。

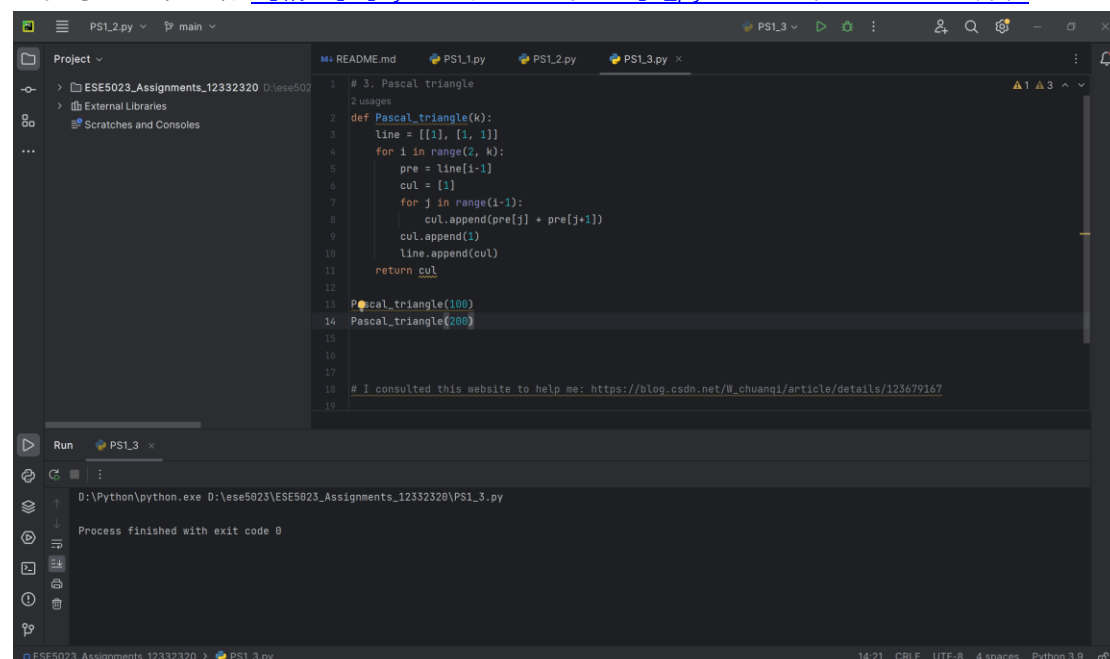
第二题:

我先引入 numpy 创建了两个随机数列, 这样比较简便。对于两个不同型的数列, 能够相乘的条件是 M1 的列数与 M2 的行数相等, 拿 M1 第一行与 M2 第一列相乘举例, 结果 = $M1(1,1) \cdot M2(1,1) + M1(1,2) \cdot M2(2,1) + M1(1,3) \cdot M2(3,1) + M1(1,4) \cdot M2(4,1) + \dots$ 。M1 的每一行需要分别乘以 M2 的每一列, 得到的新数列的行列数等于 M2 的列数。由此可见, $sum += M1[i, k] \cdot M2[k, j]$, 用 for 循环 3 次可以得到数列乘积的结果。

第三题:

我先给出了杨辉三角的前两行, 再用定义法计算第 k 行, 即第 k 行的第一个数字为 1, 之后第二个数字等一上一行的第一和第二个数字的和, 以此类推, 用 for 循环 2~k, 每一行要算 i-1 次, 第 k 行的第 j 个数字等于上一行的第 j 个和第 j+1 个数字的和, 最后一个数字也是 1。最后输出第 k 行。但是我使用 Pascal_triangle(100)和 Pascal_triangle(200)运行完没有结果(如下图), 所以选择了 print(Pascal_triangle(100)), print(Pascal_triangle(200))。

我参考了这个网站: [【精选】Python 实现杨辉三角】_python 杨辉三角-CSDN 博客](#)



```
1 # 3. Pascal triangle
2 usage:
3 def Pascal_triangle(k):
4     line = [[1], [1, 1]]
5     for i in range(2, k):
6         pre = line[i-1]
7         cul = [1]
8         for j in range(1-1):
9             cul.append(pre[j] + pre[j+1])
10        cul.append(1)
11        line.append(cul)
12    return cul
13
14 Pascal_triangle(100)
15 Pascal_triangle(200)
16
17
18 # I consulted this website to help me: https://blog.csdn.net/W_chuanqi/article/details/123679167
19
```

Run PS1_3

D:\Python\python.exe D:\ese5023\ESE5023_Assignments_12332320\PS1_3.py

Process finished with exit code 0

第四题:

本题使用倒算的方法, 即最终目标数字可被 2 整除就除以 2, 步数加一, 不可整除就先减一, 步数加一, 再除以 2, 步数加一。最后目标数字变为 1 (初始数字) 即可。

第五题:

这道题太复杂, 所以我寻求了 ChatGPT 的帮助。以下是我的理解:

第一小题中, 函数 Find_expression(num)下再定义了一个函数 backtrack(expr, pos, val, prev), expr 是当前的表达式, pos 用来决定下一步应该在哪里插入运算符, val 是当前表达式的计

算结果，prev 是最近添加到表达式中的数值。用 for 循环把每一种可能的解法算出来。

第二小题用了与第一题类似的解法，也是把每一个目标数字 num 的可能解法计算出来，但是这里不需要写明每一种具体解法，所以少定义了一个 expr，然后统计 1~100 每一个数字解法的个数，寻找最大值、最小值及其位置。我尝试把这两个题的解法合并起来，但是一直报错。