PS1_1 :random a,b,c value output

Just define a function named Print_values,and use if_else commands to finish the work

```
In [57]: runfile('C:/Users/WIN11/Documents/ESE5023_Assignments_12332302/
PS1_1.py', wdir='C:/Users/WIN11/Documents/ESE5023_Assignments_12332302')
a=15
b=235
c=4
4,235,15
```

PS1_2:

Just make two matrices and use function randint from random package to establish the matrix as asked. And for multiplication, I construct a 5 plus 5 matrix and use double for loop to calculate each element one after one.
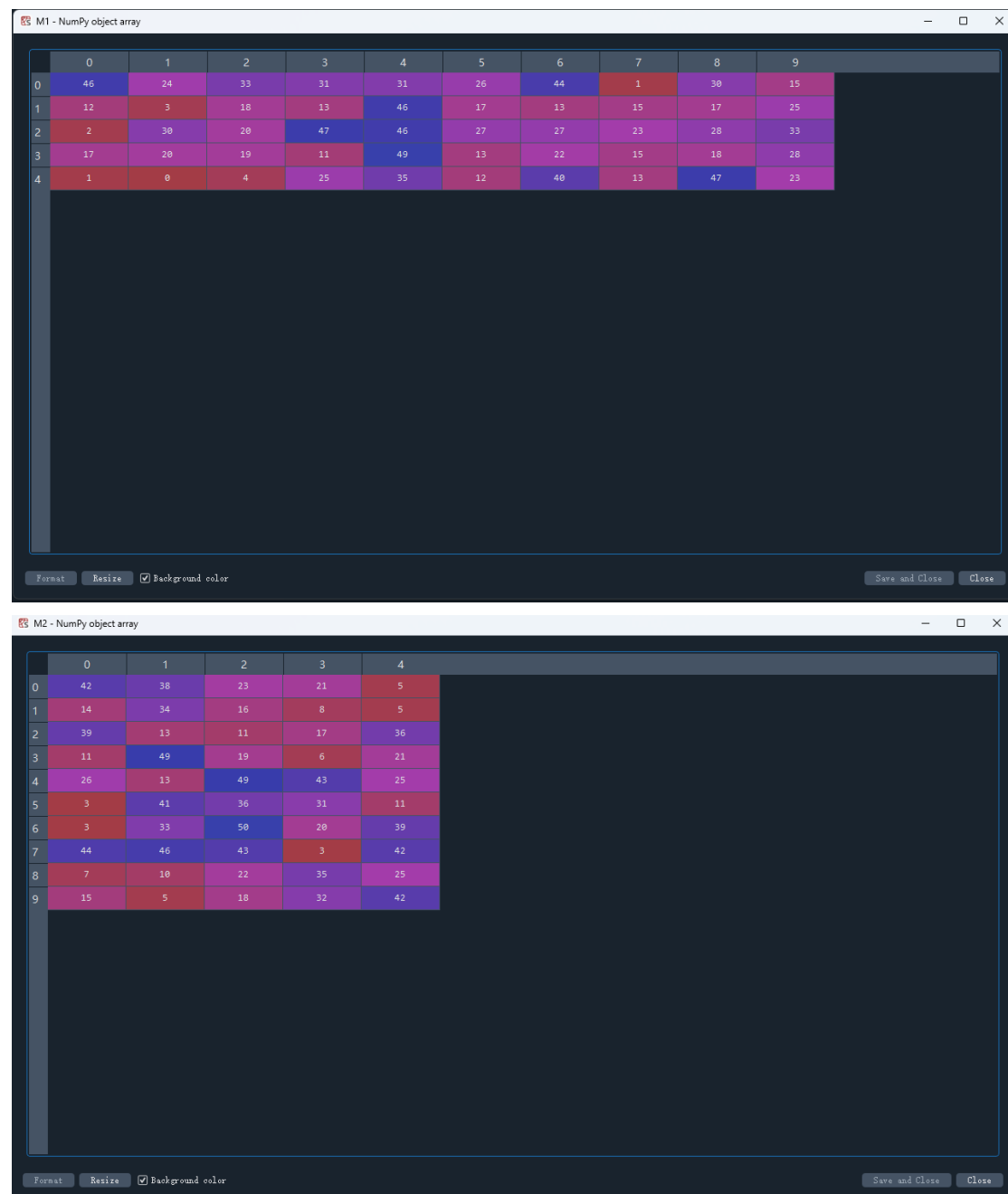
```
In [60]: runfile('C:/Users/WIN11/Documents/ESE5023_Assignments_12332302/
PS1_2.py', wdir='C:/Users/WIN11/Documents/ESE5023_Assignments_12332302')
[[4702. 4915. 3913. 3238. 2964.]
 [2587. 2027. 3023. 2638. 2146.]
 [2997. 4257. 3893. 2882. 3017.]
 [3130. 2749. 3530. 3013. 2325.]
 [1383. 1770. 2257. 1744. 1549.]]
```

PS1_3 : 100th and 200th line of Pascal Triangle

First initialize the triangle by type in the value of the first twp lines by hand. And other values can be calculated according to results of previous line. So for line x, first construct an array of n, and the xth element goes to the sum of (x-1)th and xth element of the previous line.

## 100th

[1, 99, 4851, 156849, 3764376, 71523144, 1120529256, 14887031544, 171200862756, 1731030945644, 15579278510796, 126050526132804, 924370524973896, 6186171974825304, 38000770702498296, 215337700647490344, 1130522928399324306, 5519611944537877494, 25144898858450330806, 107196674080761936594, 428786696323047746376, 1613054714739084379224, 5719012170438571889976, 19146258135816088501224, 60629817430084280253876, 181889452290252840761628, 517685364210719623706172, 1399667836569723427057428, 3599145865465003098147672, 8811701946483283447189128, 20560637875127661376774632, 45764000431735762419272568, 97248500917438495140954207, 197443926105102399225573693, 383273503615787010261407757, 711793649572175876199757263, 1265410932572757113244012912, 2154618614921181030658724688, 3515430371713505892127392912, 5498493658321124600506947888, 8247740487481686900760421832, 11868699725888281149874753368, 16390109145274293016493707032, 21726423750712434928840495368, 27651812046361280818524266832, 337966659167774898778196326128, 39674339023040098565708730672, 44739148260023940935799206928, 48467410615025936013782474172, 50445672272782096667406248628, 50445672272782096667406248628, 48467410615025936013782474172, 44739148260023940935799206928, 39674339023040098565708730672, 337966659167774898778196326128, 27651812046361280818524266832, 21726423750712434928840495368, 16390109145274293016493707032, 11868699725888281149874753368, 8247740487481686900760421832, 5498493658321124600506947888, 3515430371713505892127392912, 2154618614921181030658724688, 1265410932572757113244012912, 711793649572175876199757263, 383273503615787010261407757, 197443926105102399225573693, 97248500917438495140954207, 45764000431735762419272568, 20560637875127661376774632, 8811701946483283447189128, 3599145865465003098147672, 1399667836569723427057428, 517685364210719623706172, 181889452290252840761628, 60629817430084280253876, 19146258135816088501224, 5719012170438571889976, 1613054714739084379224, 428786696323047746376, 107196674080761936594, 25144898858450330806, 5519611944537877494, 1130522928399324306, 215337700647490344, 38000770702498296, 6186171974825304, 924370524973896, 126050526132804, 15579278510796, 1731030945644, 171200862756, 14887031544, 1120529256, 71523144, 3764376, 156849, 4851, 99, 1]

## 200th

[1, 199, 19701, 1293699, 63391251, 2472258789, 79936367511, 2203959847089, 52895036330136, 1122550215450664, 21328454093562616, 366461620334848584, 5741232051912627816, 82585414900589338584, 1097206226536401212616, 13532210127282281622264, 155620416463746238656036, 1675208012521503627885564, 16938214348828536681954036, 161358778796735007338614764, 1452229009170615066047532876, 12378523459120956991548018324, 100153507987433197477070330076, 770746561468507650149628192324, 5652141450769056101097273410376, 39564990155383392707680913872632, 264781087962950397351403038993768, 1696560304355200694140471323923032, 10421727583896232835434323846955768, 61452255753319166029629978545842632, 348229449268808607501236545093108248,

189841215891705337637770890712049352,
50437359403955349931489584373269471102,
1160906953003644566910503963011639339062,
23298321822592662584573267221641999107962,
410031599039717830992469605718533482276562,
6360490170469769280761235835048470603119352,
87363410392399278393445856104215039745835352,
1066892557216269975532081212424201849017322632,
11627253669347711916506428088408438467412653032,
113464594480811515266860347570217040690499665132,
994483798684759751456599516938961121346144123804,
7850504181489707239727786317316414425256426544804,
55957970882874445207083244558110603832551700321044,
360992022688017097651709953615480436754356081770344,
2112151454780677477844107741463807511600151218353544,
11230182325145350742854190341225599501568017133650264,
54356842559958525638607609470356165943841508430310264,
239901833990586185270393199489360386232915888168388569,
966877088507514019423099864608634283908418579587747869,
1876879054161644861233076207769701845233989007435039981,
3563350088335876475674391061127984662690616811217249819,
6617650164052342026252440542094828659282574077974892521,
12023617903700734104036124365214547845738761352940297679,
21375320717690193962730887760381418392424464627449418096,
37187201796529515524203051309156714189560369968302412304,
63318749004901607514183573850726297133575765081163566896,
105531248341502679190305956417877161889292941801939278160,
172182563083504371310499192050220632556214799782111453840,
275044873497026463262225982106196594862524939911684530160,
430198391879964468179379100217384417605487726528532213840,
658911460980705071515251533244348285193215378606992378160,
988367191471057607272877299866522427789823067910488567240,
1452045626975998213153980230668100850703567223226520240760,
2089529072965460843319142283156535370524645516350358395240,
2945480741409143598413730688304995642787753318228818460760,
4067568642898341159714199521944993982897373629935035017240,
5503181105097755686672152294396168329802329028735635611560,
72949144881528389334956437390832929002296110572975144880440,
9475003875416905741206985546165656298384603387887257143560,
12059095841439698216081617967847198925216767948220145455440,
15039995937076477550393928027315045850551249912948720736560,
18382217256426805894925912033385056039562638782492880900240,
22018260230225514753262905622406275915520083816392571627760,
25847522878960386884265150078476932596480098393156497128240,

9966663834314530225982971762382590098,
246252990031076120253743264881256829498,
5288576119238825249258962498164134766838,
99324424612105561544759718155421154091838,
1640126396158871323969878422874133929106248,
23927558260338655865720839569944246554591848,
309743000482142896122217126187671504553416248,
3571770735028382091998706667681023581492775768,
36819636619601087735603688946626721813473401268,
340393783442434545800581042710651122071498995396,
2830453888564316215684167855903197037677487121596,
21225437231435134388893644487559194557174782880396,
143891925127391430532499771720855838426561515111256,
883808055546524618388669196782727965846871786403256,
4928353394488247448302918063415550860400352842824936,
24996212272097716169578681727244076309941715555544136,
115508290439911866982041170124506852630663205414409311,
487073420526341648882313465629913511442586803250970731,

297385478284813053399609791225487289013265648179327440 07760,

335349581895640251705942956062783538674533603266050092 00240,

370649537884655015043410635648339700640273982557213259 58160,

401536999375042932963694855285701342360296814436981031 21340,

426374339542571361806810000973476683124851256567103569 22660,

443777373809615090860149184686679813048314573161679225 11340,

452742573280516405827020885387420819372522948377066684 20660,

452742573280516405827020885387420819372522948377066684 20660,

443777373809615090860149184686679813048314573161679225 11340,

426374339542571361806810000973476683124851256567103569 22660,

401536999375042932963694855285701342360296814436981031 21340,

370649537884655015043410635648339700640273982557213259 58160,

335349581895640251705942956062783538674533603266050092 00240,

297385478284813053399609791225487289013265648179327440 07760,

258475228789603868842651500784769325964800983931564971 28240,

220182602302255147532629056224062759155200838163925716 27760,

183822172564268058949259120333850560395626387824928809 00240,

150399959370764775503939280273150458505512499129487207 36560,

120590958414396982160816179678471989252167679482201454 55440,

947500387541690574120698554616565629838460338788725714 3560,

729491448815283893349564373908329290229611057297514488 0440,

550318110509775568667215229439616832980232902873563561 1560,

406756864289834115971419952194499398289737362993503501 7240,

294548074140914359841373068830499564278775331822881846 0760,

208952907296546084331914228315653537052464551635035839 5240,

145204562697599821315398023066810085070356722322652024 0760,

988367191471057607272877299866522427789823067910488567 240,

658911460980705071515251533244348285193215378606992378 160,

430198391879964468179379100217384417605487726528532213 840,

275044873497026463262225982106196594862524939911684530 160,

172182563083504371310499192050220632556214799782111453 840,

105531248341502679190305956417877161889292941801939278 160,

63318749004901607514183573850726297133575765081163566896,

37187201796529515524203051309156714189560369968302412304,

21375320717690193962730887760381418392424464627449418096,

12023617903700734104036124365214547845738761352940297679,

6617650164052342026252440542094828659282574077974892521,

3563350088335876475674391061127984662690616811217249819,

1876879054161644861233076207769701845233989007435039981,

966877088507514019423099864608634283908418579587747869,      4870734205263416488823134656299135144258680325097073 1,

239901833990586185270393199489360386232915888168388569,      11550829043991186698204117012450685263066320541440931 1,

54356842559958525638607609470356165943841508430310264,      2499621227209771616957868172724407630994171555554413 6,

11230182325145350742854190341225599501568017133650264,       492835339448824744830291806341555086040035284282493 6,

21121514547806774778441077414638075116001512183535 44,       8838080555465246183886691967827279658468717864032 56,

360992022688017097651709953615480436754356081770344,
1438919251273914305324997717208558384265615111256,
5595797088287444520708324455811060383255170321044,
21225437231435134388893644487559194557174782880396,
785050418148970723972778631731641442525642654804,
28304538885643162156841678559031970376774871215 96,
99448379868475975145659951693896112134614412380 4,
34039378344243454580058104271065112207149899539 6,
1134645944808115152668603475702170406904996651 32,
36819636619601087735603688946626721813473401268,
1162725366934771191650642808840843846741265303 2,
35717707350283820919987066667681023581492775768,
1066892557216269975532081212424201849017322632,
30974300048214289612221712618767150455341 6248,
8736341039239927839344585610421503974583535 2,
23927558260338655865720839569944246554591848,
636049017046976928076123583504847060311935 2,
16401263961588713239698784228741339291 06248,
410031599039717830992469605718533482276562,
9932442461210556154475971815542115409 1838,
232983218225926625845732672216419991 07962,
52885761192388252492589624981641347 66838,
11609069530036445669105039630116393 39062,
24625299003107612025374326488125 6829498,
50437359403955349931489584373269471 102,
9966663834314530225982971762382590098,
189841215891705337637770890712049 3352,
348229449268808607501236545093108248,
614522557533191660296299785458426 32, 10421727583896232835434323846955768, 16965603043552006941404713239230 32,
264781087962950397351403038993768, 395649901553833927076809138726 32, 56521414507690561010972734103 76,
770746561468507650149628192324, 100153507987433197477070330076, 1237852345912095699154801832 4,
1452229009170615066047532876, 16135877879673500733861476 4, 16938214348828536681954036, 16752080125215036278855 64,
15562041646374623865603 6, 1353221012728228162226 4, 1097206226536401212616, 8258541490058933858 4,
5741232051912627816, 36646162033484858 4, 21328454093562616, 1122550215450664, 52895036330136, 220395984708 9,
79936367511, 2472258789, 63391251, 1293699, 19701, 199, 1]

PS1_4:

We can see for a number x, the fastest way to decline always lies on being divided by two. So we come to the approach that for any number, we divide it by 2 if even, otherwise we minus 1. By continuing this loop, we finally get to 1, when the steps we take should be the least move.



```
In [63]: runfile('C:/Users/WIN11/Documents/ESE5023_Assignments_12332302/
PS1_4.py', wdir='C:/Users/WIN11/Documents/ESE5023_Assignments_12332302')
input the number:2
the least steps to 2 is 1

In [64]: runfile('C:/Users/WIN11/Documents/ESE5023_Assignments_12332302/
PS1_4.py', wdir='C:/Users/WIN11/Documents/ESE5023_Assignments_12332302')
input the number:5
the least steps to 5 is 3
```

PS1_5:

First there exists three types of possibilities between numbers:'+' '-' ' '

So naturally it reminds me of ternary number. With 0 stands for'+' 1 stands for '-'and 2 stands for ' '

Basically, there are nines places to insert,so the overall possibilities goes to 2*3**8(we note that for the first place,' ' is the same as'+'),and we can go through all situations by using a for loop.

For i as a decimal, we convert it to ternary integer using 'ten2three' function, and calculate the result. If the result is equal to the number we input, then output the result the specific i goes to.

As for the list Total_solutions, we merely found a list of 100, and run the loop to find whether the result falls on 1-100,and take down the overall count.

```
In [67]: Find_expression(50)
1+2+3+4-56+7+89=50
1+2+3-4+56-7+8-9=50
1+2+34-5-6+7+8+9=50
1+2+34-56+78-9=50
1+2-3+4+56+7-8-9=50
1+2-34+5-6-7+89=50
1-2+3-45+6+78+9=50
1-2+34+5+6+7+8-9=50
1-2+34-5-67+89=50
1-2-3+4+56-7-8+9=50
1-2-3-4-5-6+78-9=50
1-2-34-5-6+7+89=50
1-23+4+5-6+78-9=50
1-23-4-5-6+78+9=50
12+3+4-56+78+9=50
12-3+45+6+7-8-9=50
12-3-4-5+67-8-9=50
-1+2+3-4+56-7-8+9=50
-1+2-3+4+56-7+8-9=50
-1+2-34-5+6-7+89=50
-1+23-4+56-7-8-9=50
-1-2+3+4+56+7-8-9=50
-1-2+34+5+6+7-8+9=50
-1-23+4-5+6+78-9=50
-12+3+4+5+67-8-9=50
-12+3+45+6+7-8+9=50
-12+3-4-5+67-8+9=50
-12-3+4-5+67+8-9=50
Out[67]: 28
```
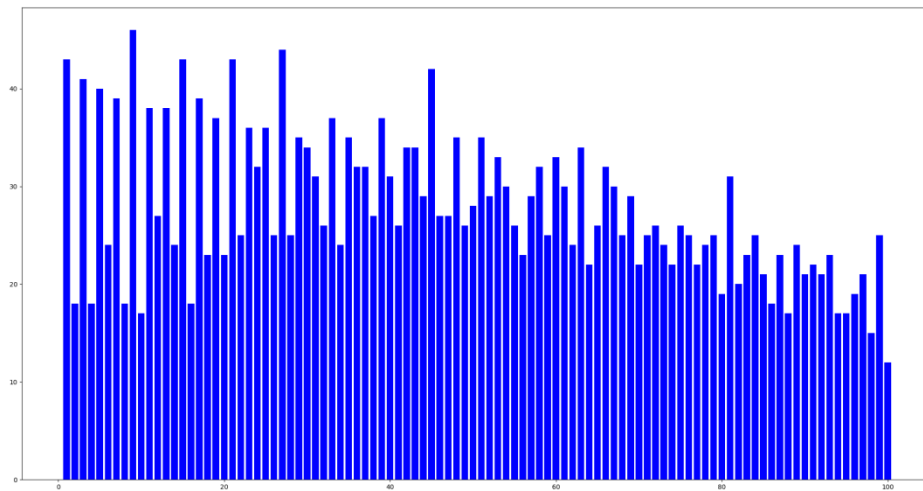
The max number falls on Find_expression(9), with a value of 46

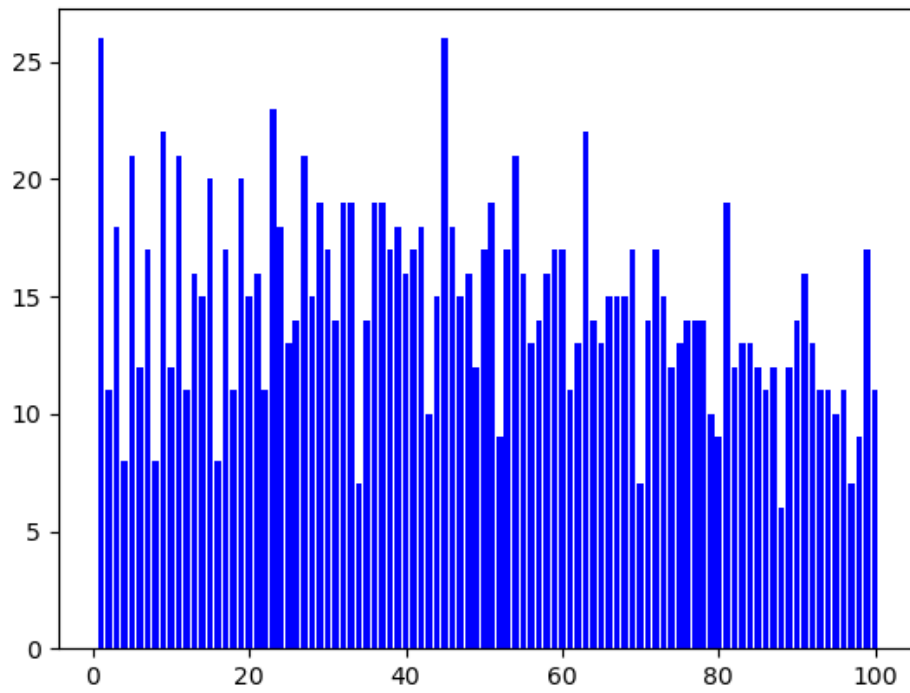The min number falls on Find_expression(100), with a value of 12

PS1_5M:

Same as the previous one but I found that 'between' maybe means that there is supposed to be no other choice in front of 1. I wonder if I get it wrong, so I add a modified version named PS1_5M in which there's nothing added before 1. And the result seems to be as below.(Time limited, I modify it on Thonny so the UI is different)

```
>>> Find_expression(50)
    12+3+4-56+7+89=50
    12+3-4+56-7+8-9=50
    12+34-5-6+7+8+9=50
    12+34-56+78-9=50
    12-3+4+56+7-8-9=50
    12-34+5-6-7+89=50
    1-2+3-45+6+78+9=50
    1-2+34+5+6+7+8-9=50
    1-2+34-5-67+89=50
    1-2-3+4+56-7-8+9=50
    1-2-3-4-5-6+78-9=50
    1-2-34-5-6+7+89=50
    1-23+4+5-6+78-9=50
    1-23-4-5-6+78+9=50
    12+3+4-56+78+9=50
    12-3+45+6+7-8-9=50
    12-3-4-5+67-8-9=50
17
```

>>> Total_solutions
[26, 11, 18, 8, 21, 12, 17, 8, 22, 12, 21, 11, 16, 15, 20, 8, 17, 11, 20, 15, 16, 11, 23, 18, 13, 14, 21, 15, 19, 17, 14, 19, 19, 17, 18, 16, 17, 18, 10, 15, 26, 18, 15, 16, 12, 17, 19, 9, 17, 21, 16, 13, 14, 16, 17, 17, 11, 13, 22, 14, 13, 15, 15, 15, 17, 7, 14, 17, 15, 12, 13, 14, 14, 14, 10, 9, 19, 12, 13, 13, 12, 11, 12, 6, 12, 14, 16, 13, 11, 11, 10, 11, 7, 9, 17, 11]

The max number falls on Find_expression(1&45), with a value of 26

The min number falls on Find_expression(88), with a value of 6