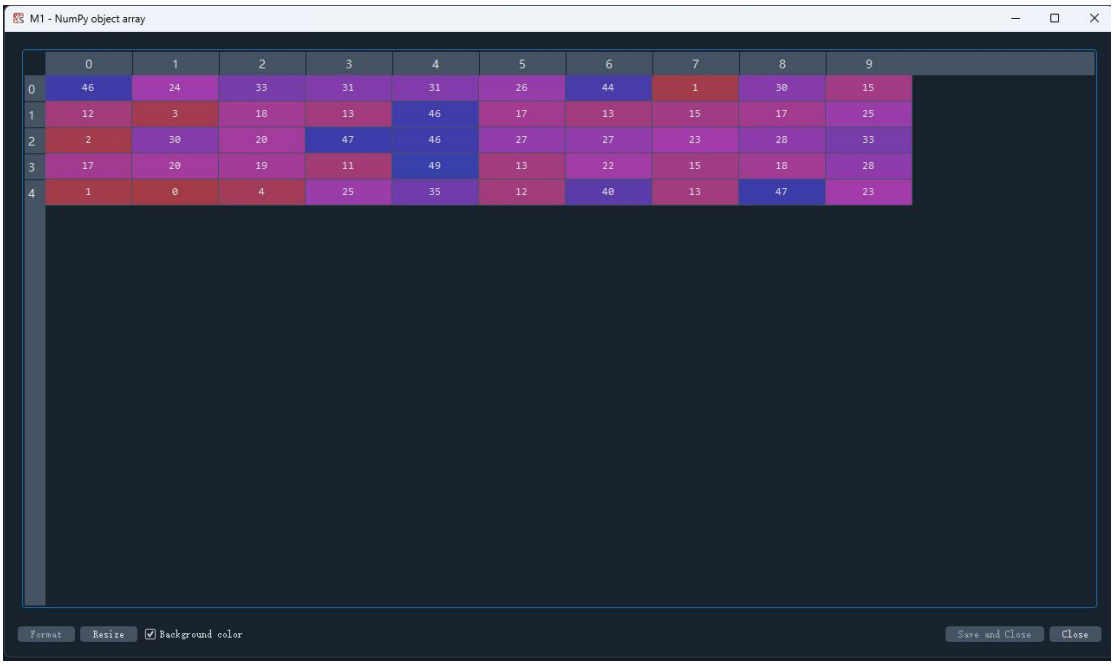
PS1_1 :random a,b,c value output

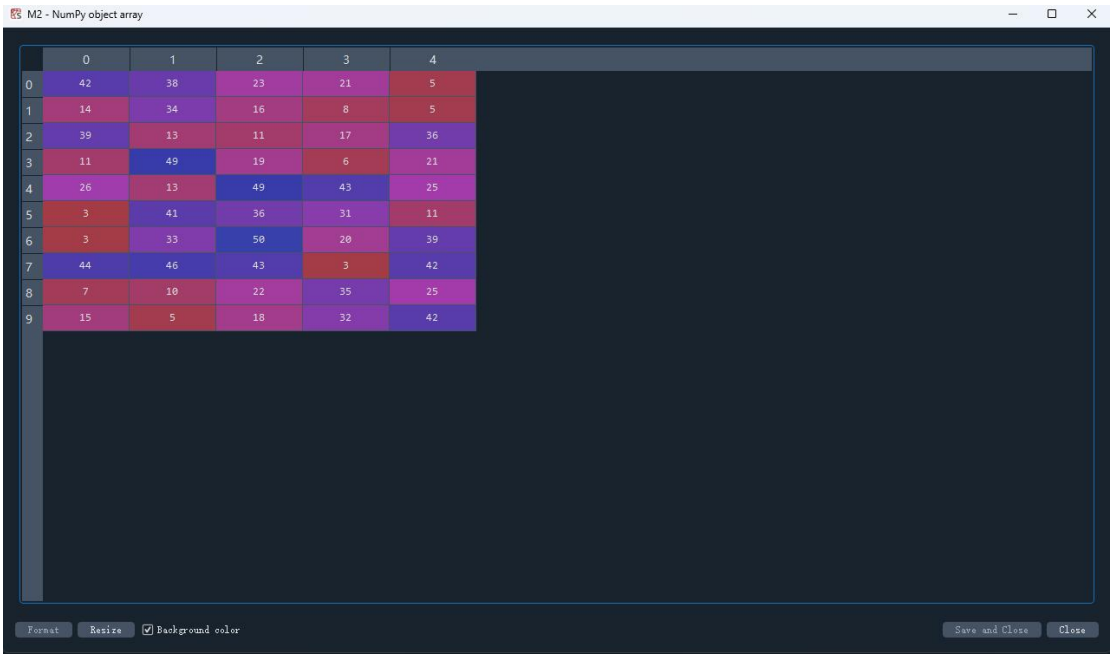
Just define a function named Print_values,and use if_else commands to finish the work

```
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, 457640004317357624192725 68, 9724850091743849514095 4207, 19744392610510239922557 3693, 38327350361578701026140 7757, 71179364957217587619975 7263, 12654109325727571132440 12912, 21546186149211810306587 24688, 35154303717135058921273 92912, 54984936583211246005069 47888, 82477404874816869007604 21832, 11868699725888281149874 753368, 16390109145274293016493 707032, 21726423750712434928840 495368, 27651812046361280818524 266832, 33796659167774898778196 326128, 39674339023040098565708 730672, 44739148260023940935799 206928, 48467410615025936013782 474172, 50445672272782096667406 248628, 50445672272782096667406 248628, 48467410615025936013782 474172, 44739148260023940935799 206928, 39674339023040098565708 730672, 33796659167774898778196 326128, 27651812046361280818524 266832, 21726423750712434928840 495368, 16390109145274293016493 707032, 11868699725888281149874 753368, 82477404874816869007604 21832, 54984936583211246005069 47888, 35154303717135058921273 92912, 21546186149211810306587 24688, 12654109325727571132440 12912, 71179364957217587619975 7263, 38327350361578701026140 7757, 19744392610510239922557 3693, 97248500917438495140954 207, 45764000431735762419272 568, 20560637875127661376774632, 8811701946483283447189128, 3599145865465003098147672, 1399667836569723427057428, 5176853642107196237061 72, 1818894522902528407616 28, 6062981743008428025387 6, 1914625813581608850122 4, 5719012170438571889976, 1613054714739084379224, 4287866963230477463 76, 1071966740807619365 94, 2514489885845033080 6, 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, 15562041646374623865 6036, 16752080125215036278855 64, 16938214348828536681954036, 16135877879673500733861 4764, 14522290091706150660475 32876, 12378523459120956991548 018324, 10015350798743319747707 0330076, 77074656146850765014962 8192324, 56521414507690561010972 73410376, 39564990155383392707680 913872632, 26478108796295039735140 3038993768, 16965603043552006941404 71323923032, 10421727583896232835434323 846955768, 61452255753319166029629978545842632, 34822944926880860750123654509 3108248,

189841215891705337637770890712049335 2,
996666383431453022598297176238259009 8,
504373594039553499314895843732694711 02,
246252990031076120253743264881256829 498,
116090695300364456691050396301163933 9062,
528857611923882524925896249816413476 6838,
232983218225926625845732672216419991 07962,
993244246121055615447597181554211540 91838,
410031599039717830992469605718533482276562,
164012639615887132396987842287413392 9106248,
636049017046976928076123583504847060311 9352,
239275582603386558657208395699442465545 91848,
873634103923992783934458561042150397458 35352,
309743000482142896122217126187671504553 416248,
106689255721626997553208121242420184901 7322632,
357177073502838209199870666768102358149 2775768,
116272536693477119165064280884084384676 412653032,
368196366196010877356036889466262721813 473401268,
113464594480811515266860347570217040690 499665132,
340393783442434545800581042710651122071 498995396,
994483798684759751456599516938961121346 144123804,
283045388856431621568416785590319703767 7487121596,
785050418148970723972778631731641442525 6426544804,
212254372314351343888936444875591945571 74782880396,
559579708828744452070832445581106038325 51700321044,
143891925127391430532499771720855838426 561515111256,
360992022688017097651709953615480436754 356081770344,
883808055546524618388669196782727965846 871786403256,
211215145478067747784410774146380751160 0151218353544,
492835339448824744830291806341555086040 0352842824936,
112301823251453507428541903412255995015 68017133650264,
249962122720977161695786817272440763099 41715555544136,
543568425599585256386076094703561659438 41508430310264,
115508290439911866982041170124506852630 663205414409311,
239901833990586185270393199489360386232 915888168388569,
487073420526341648882313465629913511442 586803250970731,
966877088507514019423099864608634283908 418579587747869,
187687905416164486123307620776970184523 39890074350399 81,
356335008833587647567439106112798466269 061681121724981 9,
661765016405234202625244054209482865928 257407797489252 1,
120236179037007341040361243652145478457 387613529402976 79,
213753207176901939627308877603814183924 244646274494180 96,
371872017965295155242030513091567141895 60369968302412304,
633187490049016075141835738507262971335 757650811635668 96,
105531248341502679190305956417877161889 292941801939278160,
172182563083504371310499192050220632556 214799782111453840,
275044873497026463262225982106196594862 524939911684530160,
430198391879964468179379100217384417605 487726528532213840,
658911460980705071515251533244348285193 215378606992378160,
988367191471057607272877299866522427789 823067910488567240,
145204562697599821315398023066810085070 356722322652024076 0,
208952907296546084331914228315653537052 464551635035839524 0,
294548074140914359841373068830499564278 775331822881846076 0,
406756864289834115971419952194499398289 737362993503501724 0,
550318110509775568667215229439616832980 232902873563561156 0,
729491448815283893349564373908329290229 611057297514488044 0,
947500387541690574120698554616565629838 460338788725714356 0,
120590958414396982160816179678471989252 167679482201454554 40,
150399959370764775503939280273150458505512499129487207365 60,
183822172564268058949259120333850560395626387824928809002 40,
220182602302255147532629056224062759155200838163925716277 60,
258475228789603868842651500784769325964800983931564971282 40,

297385478284813053399609791225487289013265648179327440077 60,

335349581895640251705942956062783538674533603266050092002 40,

370649537884655015043410635648339700640273982557213259581 60,

401536999375042932963694855285701342360296814436981031213 40,

426374339542571361806810000973476683124851256567103569226 60,

443777373809615090860149184686679813048314573161679225113 40,

452742573280516405827020885387420819372522948377066684206 60,

452742573280516405827020885387420819372522948377066684206 60,

443777373809615090860149184686679813048314573161679225113 40,

426374339542571361806810000973476683124851256567103569226 60,

401536999375042932963694855285701342360296814436981031213 40,

370649537884655015043410635648339700640273982557213259581 60,

335349581895640251705942956062783538674533603266050092002 40,

297385478284813053399609791225487289013265648179327440077 60,

258475228789603868842651500784769325964800983931564971282 40,

220182602302255147532629056224062759155200838163925716277 60,

183822172564268058949259120333850560395626387824928809002 40,

150399959370764775503939280273150458505512499129487207365 60,

120590958414396982160816179678471989252167679482201454554 40,

947500387541690574120698554616565629838460338788725714356 0,

729491448815283893349564373908329290229611057297514488044 0,

550318110509775568667215229439616832980232902873563561156 0,

406756864289834115971419952194499398289737362993503501724 0,

294548074140914359841373068830499564278775331822881846076 0,

208952907296546084331914228315653537052464551635035839524 0,

145204562697599821315398023066810085070356722322652024076 0,

988367191471057607272877299866522427789823067910488567240,

658911460980705071515251533244348285193215378606992378160,

430198391879964468179379100217384417605487726528532213840,

275044873497026463262225982106196594862524939911684530160,

172182563083504371310499192050220632556214799782111453840,

105531248341502679190305956417877161889292941801939278160,

633187490049016075141835738507262971335757650811635668 96,

371872017965295155242030513091567141895603699683024123 04,

213753207176901939627308877603814183924244646274494180 96,

120236179037007341040361243652145478457387613529402976 79,

661765016405234202625244054209482865928257407797489252 1,

356335008833587647567439106112798466269061681121724981 9,

187687905416164486123307620776970184523398900743503998 1,

966877088507514019423099864608634283908418579587747869, 48707342052634164888231346562991351144258680325097073 1,

239901833990586185270393199489360386232915888168388569, 11550829043991186698204117012450685263066320541440931 1,

543568425599585256386076094703561659438415084303102 64, 24996212272097716169578681727244076309941715555544136,

112301823251453507428541903412255995015680171336502 64, 49283533944882474483029180634155508604003528428249 36,

211215145478067747784410774146380751160015121835354 4, 88380805554652461838866919678272796584687178640325 6,

360992022688017097651709953615480436754356081770344, 1438919251273914305324997717208558384265615151112 56, 559579708828744452070832445581106038325517003210 44, 212254372314351343888936444875591945571747828803 96, 785050418148970723972778631731641442525642654480 4, 283045388856431621568416785590319703767748712159 6, 994483798684759751456599516938961121346144123804, 340393783442434545800581042710651122071498995396, 113464594480811515266860347570217040690499665132, 368196366196010877356036889466262721813473401268, 116272536693477119165064280884084384674126530 32, 357177073502838209199870666768102358149277576 8, 106689255721626997553208121242420184901732263 2, 309743000482142896122217126187671504553416248, 873634103923992783934458561042150397458353 52, 239275582603386558657208395699442465545918 48, 636049017046976928076123583504847060311935 2, 164012639615887132396987842287413392910624 8, 410031599039717830992469605718533482276562, 993244246121055615447597181554211540918 38, 232983218225926625845732672221641999107962, 528857611923882524925896249816413476683 8, 116090695300364456691050396301163933906 2, 246252990031076120253743264881256829498, 504373594039553499314895843732694711 02, 996666383431453022598297176238259009 8, 189841215891705337637770890712049335 2, 348229449268808607501236545093108248, 614522557533191660296299785458426 32, 104217275838962328354343238469557 68, 169656030435520069414047132392303 2, 264781087962950397351403038993768, 395649901553833927076809138726 32, 565214145076905610109727341037 6, 770746561468507650149628192324, 100153507987433197477070330076, 123785234591209569915480183 24, 145222900917061506606475329 76, 161358778796735007338614764, 169382143488285366819540 36, 167520801252150362788556 4, 15562041646374623865603 6, 13532210127282281622264, 1097206226536401212616, 82585414900589338584, 5741232051912627816, 366461620334848584, 21328454093562616, 1122550215450664, 52895036330136, 2203959847089, 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:'+' '-' ' '

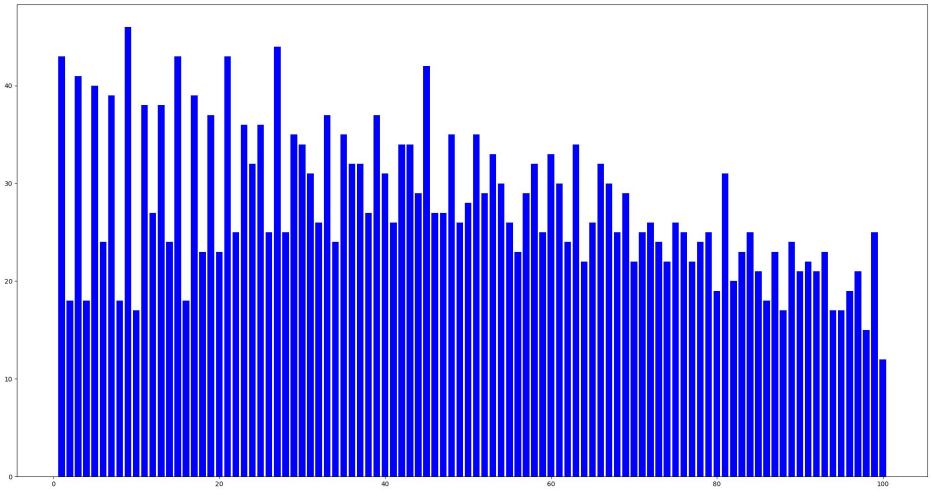
So naturally it reminds me of ternary number. With 0 stands for'+' 1 stands for '-'and 2 stands for ' '

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