

Shekhar Adhikari

1.

The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `temp.py` with the following content:

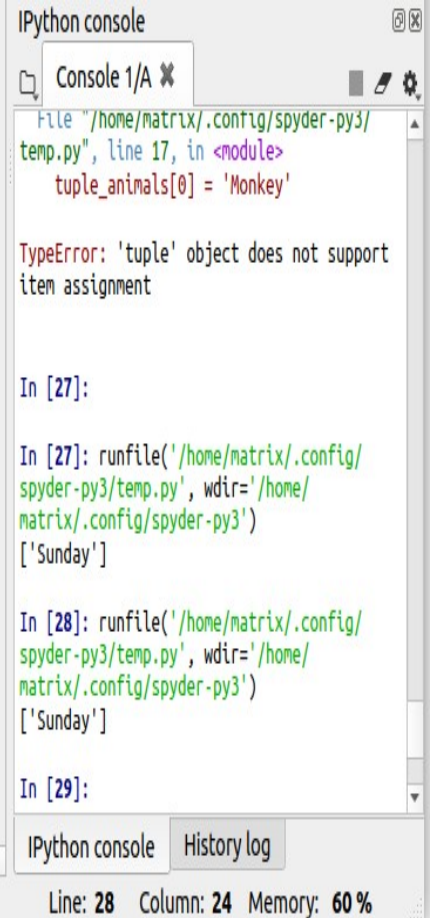
```
1 ### Shekhar Adhikari ###
2 ### Feb 7 ###
3 ### Intro to Data Science ###
4
5 ##Question 1.
6
7 #!Install Anaconda (Jupyter). Write a program to ask the user for their name and print "You are $name and welcome
8
9
10 class Nepal:
11     def __init__(self, Name):
12
13         self.Name = Name;
14
15
16 object1 = Nepal(input("Enter a name: "))
17
18 print(f'your are ${object1.Name} and welcome to CS697Ak')
```

The IPython console on the right shows the execution of the script. It displays the prompt `Enter a name: Shekhar Adhikari` and the output `your name is $Shekhar ADhikari and welcome to cs697ak`. The console also shows the execution of `runfile` commands to run the script in the current directory.

The status bar at the bottom indicates the file permissions (`RW`), end-of-line characters (`LF`), encoding (`UTF-8`), and the current line and column (`Line: 18 Column: 58`).

## Question No 2:

```
14
15 #tuple example
16 #tuple_animals = ('Zebra','Lion','Cheetah')
17 #tuple_animals[0] = 'Monkey'
18 #print(tuple_example)
19
20
21 #tuple can be beneficial to use if you already now the value such as months and dates.
22 #tuple is immutable
23 # you can use certain methods like count and index to find the particular item position and len function as well.
24 # the main difference between tuple and list is tuple do not support built in functions like pop, remove and append.
25
26 #
27 #list example
28 list_example = ['Nepal']
29 list_example[0] = 'Sunday'
30 print(list_example)
31 # in List you can use append, pop, remove.
32 # list is mutable in python
```



```
IPython console
Console 1/A X
File "/home/matrix/.config/spyder-py3/temp.py", line 17, in <module>
    tuple_animals[0] = 'Monkey'

TypeError: 'tuple' object does not support item assignment

In [27]:

In [27]: runfile('/home/matrix/.config/spyder-py3/temp.py', wdir='/home/matrix/.config/spyder-py3')
['Sunday']

In [28]: runfile('/home/matrix/.config/spyder-py3/temp.py', wdir='/home/matrix/.config/spyder-py3')
['Sunday']

In [29]:

IPython console History log
Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 28 Column: 24 Memory: 60 %
```

In this screenshot, I have pointed out the differences between tuple and list. Also, on the output screen we can see that tuple does not support item assignment. However, list supports item assignment as per the output on right hand side. Hence, tuple is immutable therefore it cannot be modified and that is one disadvantage and list are mutable and it can be modified so that is an advantage. A list has a variable size while a tuple has a fixed size. Tuples cannot be copied because it is immutable. In list you can store different data types such as ['Shekhar',10] but in tuple can only be ('Shekhar','Lion')

Question 3:

Rock, Paper and Scissor Game

Pseudo Algorithm:

- 1) Import random module
- 2) Make a variable called choices for computer(Rock, Paper, Scissors) and use the random method so that computer will shuffle between Rock, Paper and Scissors.
- 3) Assign score for computer and user wins as: 0
- 4) Start a while True loop:
- 5) make a variable and ask user to input(if they wanna choose Rock, Paper or Scissors)
- 6) run if, elif condition and check the 9 combinations for rock, paper and scissors.
- 7) Every time computer or user wins increment the value by 1.
- 8) print the new computer score and user score
- 9) After the game ends ask user if they wanna play again if they wanna play again loop runs again if they press no. We can break the loop and exit out of the game.

## Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our [tutorial](#)

Variable explorer

File explorer

Help

IPython console

Console 1/A ✖

Choose one :Paper  
Shekhar wins  
Computer Selected: Rock  
Computer Score is: 0  
Shekhar score is: 2

Do you want to play again? (Yes/No)yes

Choose one :Rock  
It is a tie!  
Computer Selected: Rock  
Computer Score is: 0  
Shekhar score is: 2

Do you want to play again? (Yes/No)yes

Choose one :Paper  
Shekhar wins  
Computer Selected: Rock  
Computer Score is: 0  
Shekhar score is: 3

Do you want to play again? (Yes/No)yes

Choose one :Scissor  
Computer wins  
Computer Selected: Rock  
Computer Score is: 1  
Shekhar score is: 3

Do you want to play again? (Yes/No)

IPython console

History log

```
7 import random
8
9 Game = 'Shekhar Vs Computer'
10 game1 = Game.center(85, '-')
11 print(game1)
12 print('')
13
14 choices_for_computer='Rock','Paper','Scissor'
15 computer_choice = random.choice(choices_for_computer)
16
17 computer_score = 0
18 shekhar_score = 0
19 computer_selects = 'Computer Selected: ' + computer_choice
20
21 while True:
22
23     shekhar_choice = input(' Choose one :')
24     if shekhar_choice == computer_choice:
25         print(' It is a tie!')
26     elif shekhar_choice == 'Rock' and computer_choice == 'Paper':
27         print(' Computer wins')
28         computer_score = computer_score + 1
29     elif shekhar_choice == 'Paper' and computer_choice == 'Rock':
30         print(' Shekhar wins')
31         shekhar_score = shekhar_score + 1
32     elif shekhar_choice == 'Paper' and computer_choice == 'Scissor':
33         print(' Computer wins')
34         computer_score = computer_score + 1
35     elif shekhar_choice == 'Scissor' and computer_choice == 'Paper':
36         print(' Shekhar wins')
37         shekhar_score = shekhar_score + 1
38     elif shekhar_choice == 'Scissor' and computer_choice == 'Rock':
39         print(' Computer wins')
40         computer_score = computer_score + 1
41     elif shekhar_choice == 'Rock' and computer_choice == 'Scissor':
42         print(' Shekhar wins')
43         shekhar_score = shekhar_score + 1
44     elif shekhar_choice != choices_for_computer:
45         print(f' Typing Error: Choose between:{choices_for_computer}')
46
47     print(computer_selects)
48     print(f' Computer Score is: {computer_score}')
49     print(f' Shekhar score is: {shekhar_score}')
50
51     shekhar_choice = input("Do you want to play again? (Yes/No)")
52     if shekhar_choice in 'yes':
53         pass
54     elif shekhar_choice in 'No':
55         break
56     else:
57         break
```

Show Applications

#### Question 4:

Write a python program that calculates the max, mean, standard deviation, median, 75 percentile of the Glucose column of the Prima data. This data can be found at <https://www.kaggle.com/uciml/pima-indians-diabetes-database#diabetes.csv>

##### 1) Finding the mean

```
import pandas as pd

new_dataset = pd.read_csv('/home/matrix/Music/data.csv')
new_dataset.head() #It has 5 rows and 9 columns
new_dataset.columns #looking at all the columns in the dataset
#new_dataset[['Glucose', 'BloodPressure']]#boht columns can be accessed
len(new_dataset['Glucose']) #to find they type

newvalue = (new_dataset['Glucose']).tolist()
print(newvalue)

new_dataset_for_sd = (new_dataset['Glucose']).tolist()
print(new_dataset_for_sd)

#data = [140, 85, 183, 89, 137, 116, 78, 115, 197, 125, 110, 160, 139, 109, 166, 100, 110, 107, 103, 115, 126, 99, 196, 119, 143, 129]
##Finding Mean

def mean(newvalue):
    total = 0
    for i in newvalue:
        total = total + i
    mean = total/len(newvalue)
    return mean

print(f'The mean is: {mean(newvalue)}')

##Finding Standard Deviation
```

parenthesis next to an object. You can activate this de  
in Preferences > Help.

New to Spyder? Read our [tutorial](#)

Variable explorer File explorer Help

Python console

Console 1/A ✖

```
In [2]: def mean(newvalue):
...:     total = 0
...:     for i in newvalue:
...:         total = total + i
...:     mean = total/len(newvalue)
...:     return mean
...:
...:
...: print(f'The mean is: {mean(newvalue)}')
The mean is: 120.89453125
```

In [3]: |



## 2) Standard Deviation:

Activities Spyder3 Feb 13 16:50

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/matrix/python-homework-1.py

```
36
37
38
39 ##Finding Standard Deviation
40
41
42 def mean(newvalue):
43
44     total = 0
45     new = 0
46     for i in newvalue:
47         total = total + i
48         mean = total/len(newvalue)
49         subtracting_mean_from_dataset = [i - mean for i in newvalue]#list(map(lambda x: x - means, newdataset))
50         squaring = [i **2 for i in subtracting_mean_from_dataset]
51         #standard_deviation = sum(squaring)/767
52         #final_standard_deviation = standard_deviation **.5
53     for j in squaring:
54         new = new + j
55         newvalues = new/767
56         final_standard_deviation = newvalues **.5
57
58     return final_standard_deviation
59
60
61
62 print(f'The standard deviation is: {mean(newvalue)}')
63
64
65
66
67
68 #standard deviation = 1022
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
```

Variable explorer

Name	Type	Size	Value
newvalue	list	768	[148, 85, 183, 89, 137, 116, 78, 115, 197, 125, ...]
new_dataset	DataFrame	(768, 9)	Column names: Pregnancies, Glucose, BloodPressure, SkinThicknes...
new_dataset_for_sd	list	768	[148, 85, 183, 89, 137, 116, 78, 115, 197, 125, ...]

Variable explorer File explorer Help

IPython console

Console 1/A

```
148, 85, 183, 89, 137, 116, 78, 115, 197, 125, 102, 101, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 880, 881, 882, 883, 884, 885, 886, 887, 888, 889, 890, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926, 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959, 960, 961, 962, 963, 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000, 1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019, 1020, 1021, 1022, 1023, 1024, 1025, 1026, 1027, 1028, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1045, 1046, 1047, 1048, 1049, 1050, 1051, 1052, 1053, 1054, 1055, 1056, 1057, 1058, 1059, 1060, 1061, 1062, 1063, 1064, 1065, 1066, 1067, 1068, 1069, 1070, 1071, 1072, 1073, 1074, 1075, 1076, 1077, 1078, 1079, 1080, 1081, 1082, 1083, 1084, 1085, 1086, 1087, 1088, 1089, 1090, 1091, 1092, 1093, 1094, 1095, 1096, 1097, 1098, 1099, 1100, 1101, 1102, 1103, 1104, 1105, 1106, 1107, 1108, 1109, 1110, 1111, 1112, 1113, 1114, 1115, 1116, 1117, 1118, 1119, 1120, 1121, 1122, 1123, 1124, 1125, 1126, 1127, 1128, 1129, 1130, 1131, 1132, 1133, 1134, 1135, 1136, 1137, 1138, 1139, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1150, 1151, 1152, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1163, 1164, 1165, 1166, 1167, 1168, 1169, 1170, 1171, 1172, 1173, 1174, 1175, 1176, 1177, 1178, 1179, 1180, 1181, 1182, 1183, 1184, 1185, 1186, 1187, 1188, 1189, 1190, 1191, 1192, 1193, 1194, 1195, 1196, 1197, 1198, 1199, 1200, 1201, 1202, 1203, 1204, 1205, 1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1214, 1215, 1216, 1217, 1218, 1219, 1220, 1221, 1222, 1223, 1224, 1225, 1226, 1227, 1228, 1229, 1230, 1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239, 1240, 1241, 1242, 1243, 1244, 1245, 1246, 1247, 1248, 1249, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1259, 1260, 1261, 1262, 1263, 1264, 1265, 1266, 1267, 1268, 1269, 1270, 1271, 1272, 1273, 1274, 1275, 1276, 1277, 1278, 1279, 1280, 1281, 1282, 1283, 1284, 1285, 1286, 1287, 1288, 1289, 1290, 1291, 1292, 1293, 1294, 1295, 1296, 1297, 1298, 1299, 1300, 1301, 1302, 1303, 1304, 1305, 1306, 1307, 1308, 1309, 1310, 1311, 1312, 1313, 1314, 1315, 1316, 1317, 1318, 1319, 1320, 1321, 1322, 1323, 1324, 1325, 1326, 1327, 1328, 1329, 1330, 1331, 1332, 1333, 1334, 1335, 1336, 1337, 1338, 1339, 1340, 1341, 1342, 1343, 1344, 1345, 1346, 1347, 1348, 1349, 1350, 1351, 1352, 1353, 1354, 1355, 1356, 1357, 1358, 1359, 1360, 1361, 1362, 1363, 1364, 1365, 1366, 1367, 1368, 1369, 1370, 1371, 1372, 1373, 1374, 1375, 1376, 1377, 1378, 1379, 1380, 1381, 1382, 1383, 1384, 1385, 1386, 1387, 1388, 1389, 1390, 1391, 1392, 1393, 1394, 1395, 1396, 1397, 1398, 1399, 1400, 1401, 1402, 1403, 1404, 1405, 1406, 1407, 1408, 1409, 1410, 1411, 1412, 1413, 1414, 1415, 1416, 1417, 1418, 1419, 1420, 1421, 1422, 1423, 1424, 1425, 1426, 1427, 1428, 1429, 1430, 1431, 1432, 1433, 1434, 1435, 1436, 1437, 1438, 1439, 1440, 1441, 1442, 1443, 1444, 1445, 1446, 1447, 1448, 1449, 1450, 1451, 1452, 1453, 1454, 1455, 1456, 1457, 1458, 1459, 1460, 1461, 1462, 1463, 1464, 1465, 1466, 1467, 1468, 1469, 1470, 1471, 1472, 1473, 1474, 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485, 1486, 1487, 1488, 1489, 1490, 1491, 1492, 1493, 1494, 1495, 1496, 1497, 1498, 1499, 1500, 1501, 1502, 1503, 1504, 1505, 1506, 1507, 1508, 1509, 1510, 1511, 1512, 1513, 1514, 1515, 1516, 1517, 1518, 1519, 1520, 1521, 1522, 1523, 1524, 1525, 1526, 1527, 1528, 1529, 1530, 1531, 1532, 1533, 1534, 1535, 1536, 1537, 1538, 1539, 1540, 1541, 1542, 1543, 1544, 1545, 1546, 1547, 1548, 1549, 1550, 1551, 1552, 1553, 1554, 1555, 1556, 1557, 1558, 1559, 1560, 1561, 1562, 1563, 1564, 1565, 1566, 1567, 1568, 1569, 1570, 1571, 1572, 1573, 1574, 1575, 1576, 1577, 1578, 1579, 1580, 1581, 1582, 1583, 1584, 1585, 1586, 1587, 1588, 1589, 1590, 1591, 1592, 1593, 1594, 1595, 1596, 1597, 1598, 1599, 1600, 1601, 1602, 1603, 1604, 1605, 1606, 1607, 1608, 1609, 1610, 1611, 1612, 1613, 1614, 1615, 1616, 1617, 1618, 1619, 1620, 1621, 1622, 1623, 1624, 1625, 1626, 1627, 1628, 1629, 1630, 1631, 1632, 1633, 1634, 1635, 1636, 1637, 1638, 1639, 1640, 1641, 1642, 1643, 1644, 1645, 1646, 1647, 1648, 1649, 1650, 1651, 1652, 1653, 1654, 1655, 1656, 1657, 1658, 1659, 1660, 1661, 1662, 1663, 1664, 1665, 1666, 1667, 1668, 1669, 1670, 1671, 1672, 1673, 1674, 1675, 1676, 1677, 1678, 1679, 1680, 1681, 1682, 1683, 1684, 1685, 1686, 1687, 1688, 1689, 1690, 1691, 1692, 1693, 1694, 1695, 1696, 1697, 1698, 1699, 1700, 1701, 1702, 1703, 1704, 1705, 1706, 1707, 1708, 1709, 1710, 1711, 1712, 1713, 1714, 1715, 1716, 1717, 1718, 1719, 1720, 1721, 1722, 1723, 1724, 1725, 1726, 1727, 1728, 1729, 1730, 1731, 1732, 1733, 1734, 1735, 1736, 1737, 1738, 1739, 1740, 1741, 1742, 1743, 1744, 1745, 1746, 1747, 1748, 1749, 1750, 1751, 1752, 1753, 1754, 1755, 1756, 1757, 1758, 1759, 1760, 1761, 1762, 1763, 1764, 1765, 1766, 1767, 1768, 1769, 1770, 1771, 1772, 1773, 1774, 1775, 1776, 1777, 1778, 1779, 1780, 1781, 1782, 1783, 1784, 1785, 1786, 1787, 1788, 1789, 1790, 1791, 1792, 1793, 1794, 1795, 1796, 1797, 1798, 1799, 1800, 1801, 1802, 1803, 1804, 1805, 1806, 1807, 1808, 1809, 1810, 1811, 1812, 1813, 1814, 1815, 1816, 1817, 1818, 1819, 1820, 1821, 1822, 1823, 1824, 1825, 1826, 1827, 1828, 1829, 1830, 1831, 1832, 1833, 1834, 1835, 1836, 1837, 1838, 1839, 1840, 1841, 1842, 1843, 1844, 1845, 1846, 1847, 1848, 1849, 1850, 1851, 1852, 1853, 1854, 1855, 1856, 1857, 1858, 1859, 1860, 1861, 1862, 1863, 1864, 1865, 1866, 1867, 1868, 1869, 1870, 1871, 1872, 1873, 1874, 1875, 1876, 1877, 1878, 1879, 1880, 1881, 1882, 1883, 1884, 1885, 1886, 1887, 1888, 1889, 1890, 1891, 1892, 1893, 1894, 1895, 1896, 1897, 1898, 1899, 1900, 1901, 1902, 1903, 1904, 1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912, 1913, 1914, 1915, 1916, 1917, 1918, 1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926, 1927, 1928, 1929, 1930, 1931, 1932, 1933, 1934, 1935, 1936, 1937, 1938, 1939, 1940, 1941, 1942, 1943, 1944, 1945, 1946, 1947, 1948, 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959, 1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 218
```

Activities

Spyder3

Feb 14 14:40

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

/home/matrix

temp.py python-homework-1.py\*

```
6 @author: matrix
7 """
8
9
10 import pandas as pd
11
12 new_dataset = pd.read_csv('/home/matrix/Music/data.csv')
13 new_dataset.head() #It has 5 rows and 9 columns
14 new_dataset.columns #looking at all the columns in the dataset
15 #new_dataset[['Glucose','BloodPressure']]#both columns can be accessed
16 type(new_dataset['Glucose']) #to find they type
17
18 newvalue = (new_dataset['Glucose'].tolist())
19 print(newvalue)
20
21 new_dataset_for_sd = (new_dataset['Glucose'].tolist())
22 print(new_dataset_for_sd)
23
24
25 #data = [148, 85, 183, 89, 137, 116, 78, 115, 197, 125, 110, 168, 139, 109, 166, 100, 118, 107, 103, 115, 126, 99, 196, 119, 143, 1
26 ###Finding Mean
27
28
29
30
31 ## Finding the median
32
33
34
35 def median(newvalue):
36     asc_order = sorted(newvalue)
37     findingthelength = len(newvalue)
38     index = (findingthelength - 1) // 2
39
40     if (findingthelength % 2):
41         return asc_order[index]
42     else:
43         return (asc_order[index] + asc_order[index + 1])/2.0
44
45 print(f'The median is: {median(newvalue)}')
46
47 #
48
49
50
51
52
53
54
55
56
57
58
59
```

Variable explorer

Name	Type	Size	Value
i	int	1	767
j	int	1	1
k	int	1	75
max	int	1	199
maximum	int	1	199
newvalue	list	768	[0, 0, 0, 0, 0, 44, 56, 57, 61, ...]
mylist	list	8	[1, 6, 7, 8, 1, 10, 15, 9]
n	int	1	768

Variable explorer File explorer Help

IPython console

Console 1/A

```
...
...
...: print(f'The median is: {median(newvalue)}')
Traceback (most recent call last):

File "<ipython-input-40-ccade008fa3d>", line 12, in <module>
    print(f'The median is: {median(newvalue)}')

File "<ipython-input-40-ccade008fa3d>", line 9, in median
    return (ascendingorder[index] + ascendingorder[index + 1])/2.0
NameError: name 'ascendingorder' is not defined

In [41]:

In [41]: def median(newvalue):
...:     asc_order = sorted(newvalue)
...:     findingthelength = len(newvalue)
...:     index = (findingthelength - 1) // 2
...:
...:     if (findingthelength % 2):
...:         return asc_order[index]
...:     else:
...:         return (asc_order[index] + asc_order[index + 1])/2.0
...:
...:     print(f'The median is: {median(newvalue)}')
The median is: 117.0

In [42]:
```

IPython console History log

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 31 Column: 23 Memory: 65 %

IPython console History log

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 39 Column: 15 Memory: 64 %

## 5) Finding the 75<sup>th</sup> percentile:

Here I have arranged the data in ascending order and after checking the length of the glucose column it is 768. By following the formula.

By using the formula:  $p/100 \times (n+1)$  which leads up to  $75/100 \times (767 + 1)$  I found 576.75 which is stored inside a variable called finding\_percentile. But there is no n called 576.75 therefore I added the value of 576 and 577 position and divided it by 2 by following formula. And answer is 141 I am not sure why using numpy it has answer of 140.25.

The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script named `python-homework-1.py` with the following code:

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Tue Feb 11 10:48:24 2020
5
6@author: matrix
7"""
8
9##Displaying histogram, bargraph of Glucose and Blood Pressure column of this data:
10##Also plot the scatter plot between Glucose and Blood Pressure column.
11##Can you decipher and relationship between these two variable
12
13
14
15import csv
16import pandas as pd
17
18
19
20
21df = pd.read_csv('/home/matrix/Music/data.csv')
22df.head()
23
24Glucose = df['Glucose'].tolist()
25ascending = sorted(Glucose)
26len(ascending)
27
28def percentile(ascending):
29    finding_percentile = 75/100 * 769
30    #Since we do not have 576.75 value
31    #we are gonna average the 576 and 577 value
32    newvalue = (141 + 141)/2
33    print(f'The 75th percentile is : {newvalue}')
34
35percentile(ascending)
```

The Variable explorer on the right shows the following variables:

Name	Type	Size	Value
Glucose	list	768	[148, 85, 183, 89, 137, 116, 78, 115, 197, 125, ...]
ascending	list	768	[0, 0, 0, 0, 0, 44, 56, 57, 57, 61, ...]
df	DataFrame	(768, 9)	Column names: Pregnancies, Glucose, BloodPressure, SkinThickness, Insu ...

The IPython console shows the following interactions:

```
In [58]: def percentile(ascending):
...:     finding_percentile = 75/100 * 769
...:     #Since we do not have 576.75 value
...:     #we are gonna average the 576 and 577 value
...:     newvalue = (141 + 141)/2
...:     print(f'The 75th percentile is : {newvalue}')
...:
...:     percentile(ascending)
The 75th percentile is : 141.0

In [59]: len(ascending)
Out[59]: 768

In [60]:
```

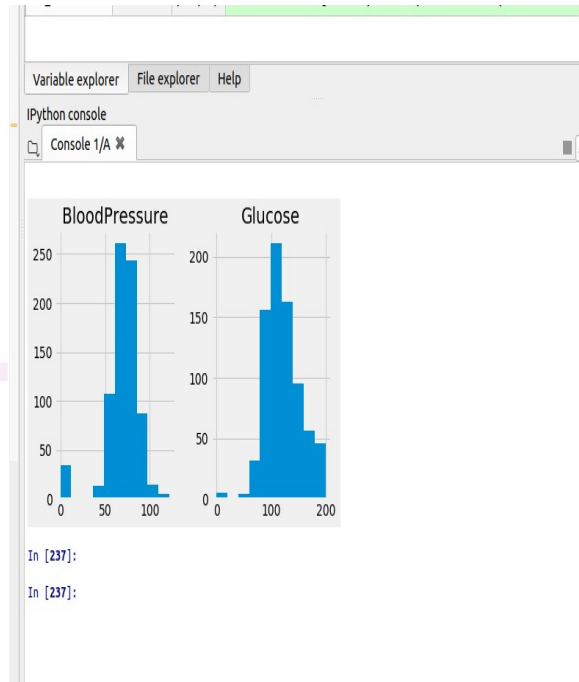
The status bar at the bottom indicates: Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 26 Column: 15 Memory: 45 %



20 Points 6. Display histogram, bar-graph of Glucose and Blood pressure column of this data. Also, plot the scatter plot between Glucose and Blood pressure column. Can you decipher and relationship between these two variables based on the scatter plot.

Histogram:

```
15
16 import pandas as pd
17 import matplotlib.pyplot as plt
18 import numpy as np
19
20 new_dataset = pd.read_csv('/home/matrix/Music/data.csv')
21
22 type(new_dataset['Glucose']) #to find they type
23
24 Glucose = new_dataset['Glucose']
25 BloodPressure = new_dataset['BloodPressure']
26
27 hist = new_dataset.hist(bins=10,column = ['Glucose','BloodPressure'])
28 hist.set_title('asdfa')
29 plt.show()
30
31
32
33
34
```



## Scatter Plot:

Scatter Plot between Blood Pressure and Glucose column. From this scatter plot we can tell that Blood Pressure is always higher than Glucose in a human body. Also, from the scatter plot it is hard to tell if Blood Pressure and Glucose are directly related. In some cases person with lower blood pressure and high blood pressure have same amount of Glucose in their body.

Activities Spyder3 Feb 21 15:29

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/matrix/python-homework-1.py

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Tue Feb 11 10:48:24 2020
5
6 @author: matrix
7 """
8
9 ##Displaying histogram, bargraph of Glucose and Blood Pressure column of this data:
10 ##Also plot the scatter plot between Glucose and Blood Pressure column.
11 ##Can you decipher and relationship between these two variable
12
13
14
15 import csv
16 import matplotlib.pyplot as plt
17 import pandas as pd
18
19
20
21 df = pd.read_csv('/home/matrix/Music/data.csv')
22 df.head()
23
24 y = df['Glucose']
25 x = df['BloodPressure']
26
27 plt.xlabel('BloodPressure'); plt.ylabel('Glucose')
28 plt.scatter(x,y)
```

Variable explorer

Name	Type	Size	Value
Glucose	list	768	[148, 85, 183, 89, 137, 116, 78, 115, 197, 125, ...]
ascending	list	768	[0, 0, 0, 0, 0, 44, 56, 57, 57, 61, ...]
df	DataFrame	(768, 9)	Column names: Pregnancies, Glucose, BloodPressure, SkinThicknes...
finding_percentile	float	1	576.75
x	Series	(768,)	Series object of pandas.core.series module
y	Series	(768,)	Series object of pandas.core.series module

Variable explorer File explorer Help

IPython console

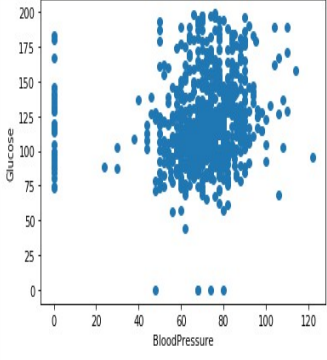
Console 1/A

```
764 122
765 121
766 126
767 93
Name: Glucose, Length: 768, dtype: int64

In [71]: y = df['Glucose']
...: x = df['BloodPressure']

In [72]: plt.xlabel('BloodPressure'); plt.ylabel('Glucose')
...: plt.scatter(x,y)
Out[72]: <matplotlib.collections.PathCollection at 0x7f956c13b358>

In [73]:
```

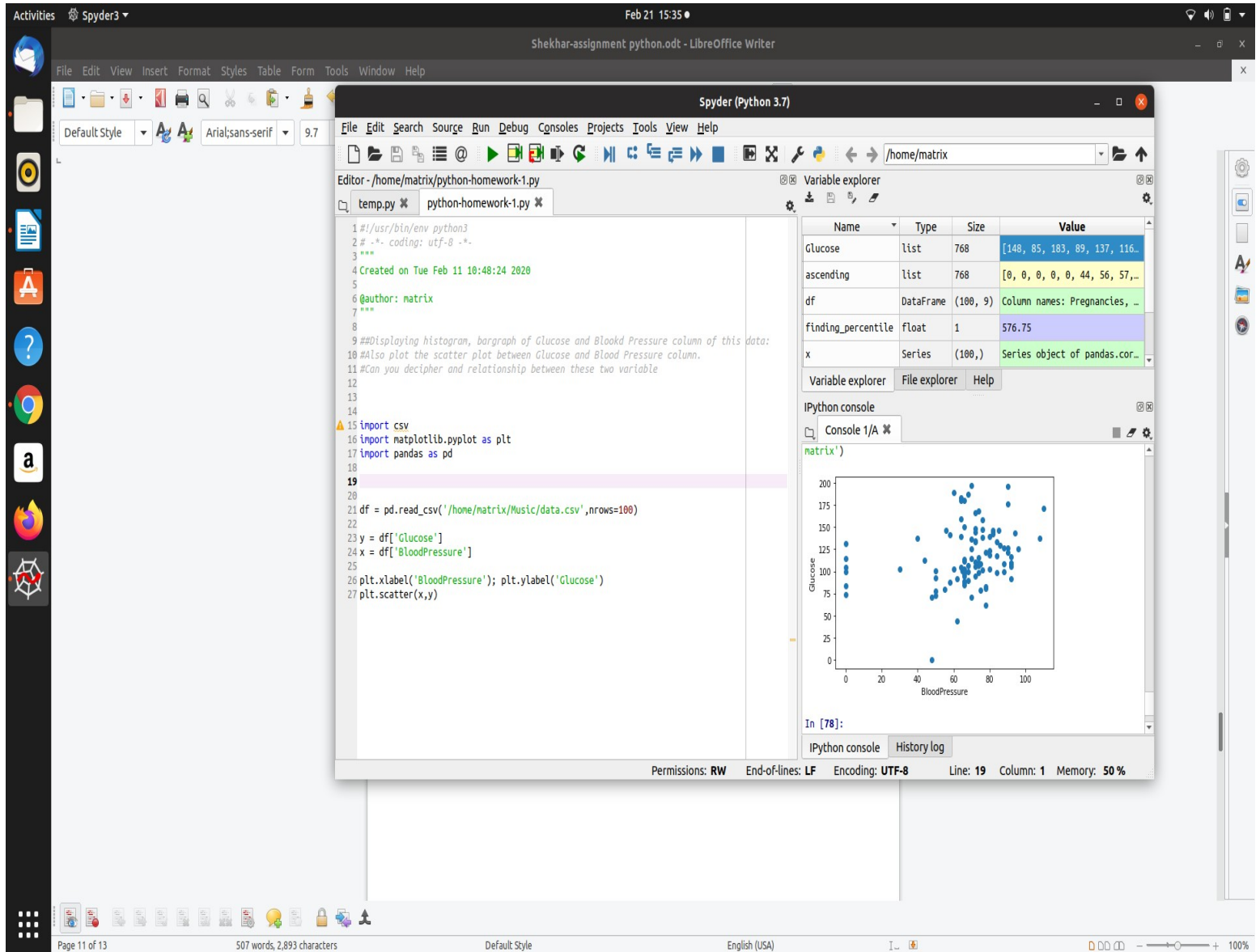


IPython console History log

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 27 Column: 51 Memory: 52 %

## Scatter Plot:

This time we will analyze only 100 rows of the data set.



Bar Graph:

Bar Graph of Blood Pressure and Glucose column

Activities Spyder3 Feb 21 15:39 Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/matrix/python-homework-1.py

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Tue Feb 11 10:48:24 2020
5
6 @author: matrix
7 """
8
9 #Displaying histogram, bargraph of Glucose and Blood Pressure column of this data:
10 #Also plot the scatter plot between Glucose and Blood Pressure column.
11 #Can you decipher and relationship between these two variable
12
13
14
15 import csv
16 import matplotlib.pyplot as plt
17 import pandas as pd
18
19
20
21 df = pd.read_csv('/home/matrix/Music/data.csv')
22
23 y = df['Glucose']
24 x = df['BloodPressure']
25
26 plt.xlabel('BloodPressure'); plt.ylabel('Glucose')
27 plt.bar(x,y)
```

Variable explorer

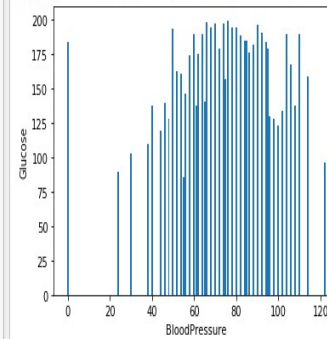
Name	Type	Size	Value
Glucose	list	768	[148, 85, 183, 89, 137, 116, 78, 115, 197, 125, ...]
ascending	list	768	[0, 0, 0, 0, 0, 44, 56, 57, 57, 61, ...]
df	DataFrame	(768, 9)	Column names: Pregnancies, Glucose, BloodPressure, SkinThicknes...
finding_percentile	float	1	576.75
x	Series	(768,)	Series object of pandas.core.series module
y	Series	(768,)	Series object of pandas.core.series module

Variable explorer File explorer Help

IPython console

Console 1/A

```
In [79]: df = pd.read_csv('/home/matrix/Music/data.csv')
...:
...: y = df['Glucose']
...: x = df['BloodPressure']
...:
...: plt.xlabel('BloodPressure'); plt.ylabel('Glucose')
...: plt.bar(x,y)
Out[79]: <BarContainer object of 768 artists>
```



IPython console History log

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 21 Column: 48 Memory: 50 %

Question6: Find the area of 500 circle.

Activities Spyder3 Feb 16 18:01

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/matrix/untitled1.py

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-import pandas as pd
3
4new_dataset = pd.read_csv('/home/matrix/Music/circle1.csv')
5new_dataset.head() #It has 5 rows and 9 columns
6new_dataset.columns #Looking at all the columns in the dataset
7new_dataset[['Glucose','BloodPressure']]#boht columns can be accessed
8
9type(new_dataset['Circle'])
10#since we know the data type is series I would like to convert it to list because easy to manipulate
11
12# now we are converting the series object to list
13newvalue = (new_dataset['Circle']).tolist())
14
15
16##Now we have radius of circle in a list now we will calculate Area of cirlice
17
18
19
20
21
22def area(newvalue):
23    pi = 3.1415
24    my_list = [pi*i * i for i in newvalue]
25    print(my_list)
26area(newvalue)
```

Variable explorer

Name	Type	Size	Value
newvalue	list	500	[6, 1, 0, 1, 0, 5, 3, 10, 2, 8, ...]
new_dataset	DataFrame	(500, 1)	Column names: Circle

Variable explorer File explorer Help

IPython console

Console 1/A

```
[113.094, 3.1415, 201.056, 3.1415, 0.0, 78.53750000000001, 28.2735, 314.15000000000003, 12.566,
201.056, 50.264, 314.15000000000003, 314.15000000000003, 3.1415, 78.53750000000001, 153.9335, 0.0,
153.9335, 3.1415, 3.1415, 28.2735, 201.056, 153.9335, 254.46150000000003, 380.12149999999997,
314.15000000000003, 153.9335, 3.1415, 530.9135, 78.53750000000001, 78.53750000000001, 28.2735, 28.2735,
113.094, 314.15000000000003, 50.264, 380.12149999999997, 254.46150000000003, 12.566, 50.264, 28.2735,
153.9335, 153.9335, 254.46150000000003, 153.9335, 0.0, 3.1415, 12.566, 153.9335, 153.9335, 3.1415,
3.1415, 78.53750000000001, 201.056, 153.9335, 3.1415, 153.9335, 0.0, 0.0, 12.566, 201.056,
78.53750000000001, 12.566, 153.9335, 78.53750000000001, 0.0, 12.566, 3.1415, 50.264, 12.566,
78.53750000000001, 530.9135, 50.264, 3.1415, 3.1415, 153.9335, 78.53750000000001, 0.0, 12.566, 28.2735,
12.566, 153.9335, 0.0, 78.53750000000001, 12.566, 530.9135, 12.566, 706.8375000000001, 3.1415, 3.1415,
50.264, 153.9335, 50.264, 12.566, 113.094, 12.566, 3.1415, 113.094, 3.1415, 3.1415, 0.0,
3.1415, 12.566, 3.1415, 3.1415, 50.264, 28.2735, 0.0, 28.2735, 201.056, 3.1415, 50.264, 153.9335,
50.264, 78.53750000000001, 78.53750000000001, 50.264, 50.264, 0.0, 113.094, 12.566, 78.53750000000001,
0.0, 3.1415, 28.2735, 3.1415, 3.1415, 0.0, 50.264, 254.46150000000003, 28.2735, 201.056, 12.566,
12.566, 0.0, 0.0, 0.0, 78.53750000000001, 28.2735, 78.53750000000001, 12.566, 314.15000000000003,
50.264, 0.0, 254.46150000000003, 12.566, 78.53750000000001, 12.566, 3.1415, 50.264, 254.46150000000003,
3.1415, 201.056, 153.9335, 12.566, 3.1415, 12.566, 907.8935, 50.264, 153.9335, 0.0, 12.566, 0.0,
113.094, 28.2735, 50.264, 50.264, 28.2735, 113.094, 113.094, 12.566, 3.1415, 12.566, 201.056, 113.094,
0.0, 78.53750000000001, 78.53750000000001, 113.094, 0.0, 3.1415, 78.53750000000001, 50.264, 153.9335,
201.056, 3.1415, 201.056, 78.53750000000001, 28.2735, 254.46150000000003, 153.9335, 380.12149999999997,
201.056, 78.53750000000001, 3.1415, 28.2735, 50.264, 50.264, 0.0, 3.1415, 0.0, 12.566, 113.094,
78.53750000000001, 201.056, 78.53750000000001, 3.1415, 153.9335, 12.566, 0.0, 153.9335, 0.0,
254.46150000000003, 452.376, 78.53750000000001, 113.094, 78.53750000000001, 78.53750000000001, 0.0,
12.566, 153.9335, 153.9335, 3.1415, 3.1415, 0.0, 28.2735, 50.264, 0.0, 50.264, 113.094, 3.1415, 50.264,
28.2735, 50.264, 153.9335, 0.0, 254.46150000000003, 0.0, 3.1415, 50.264, 28.2735, 113.094, 12.566,
254.46150000000003, 314.15000000000003, 0.0, 254.46150000000003, 3.1415, 254.46150000000003, 12.566,
12.566, 0.0, 452.376, 3.1415, 28.2735, 12.566, 3.1415, 380.12149999999997, 28.2735, 28.2735, 50.264,
28.2735, 50.264, 78.53750000000001, 0.0, 12.566, 0.0, 12.566, 314.15000000000003, 12.566, 28.2735,
3.1415, 530.9135, 12.566, 153.9335, 0.0, 78.53750000000001, 12.566, 0.0, 314.15000000000003, 153.9335,
153.9335, 12.566, 153.9335, 78.53750000000001, 3.1415, 50.264, 78.53750000000001, 0.0, 0.0, 12.566,
3.1415, 0.0, 113.094, 12.566, 0.0, 615.734, 201.056, 0.0, 12.566, 78.53750000000001, 78.53750000000001,
28.2735, 12.566, 314.15000000000003, 0.0, 0.0, 12.566, 113.094, 0.0, 12.566, 28.2735, 153.9335, 12.566,
28.2735, 28.2735, 28.2735, 113.094, 50.264, 28.2735, 0.0, 530.9135, 12.566, 3.1415, 3.1415]
```

IPython console History log

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 26 Column: 15 Memory: 59%



Here, I have imported a csv file called circle csv which has a column which has radius of 500 circle. I have stored it in a variable called mewvalue.