

Introduction to Computer Programming

Week 7.2: Modules for reading & writing files



University of
BRISTOL

Importing Python modules imports sections of pre-written code.

csv : A Python modules for reading/writing delimited files

CSV (and other delimited files)

Delimited file:

Uses a set character (tab, space, vertical bar etc) to separate values.

CSV (comma-seperated-value):

A *delimited* text file that uses a comma to separate values.

The CSV file is a widely used format for storing tabular data in plain text and is supported by software applications e.g. Microsoft Excel, Google Spreadsheet.

	A	B	C	D	E	F	G	H	I
1	sample	moisture	knotratio	treering	Edyn	density	beamheig	Estat	bstrength
2	units	%	-	mm	N/mm2	kg/m3	cm	N/mm2	N/mm2
3	DO1	13.3	0.04	3	14053	675	101	15452	58.4
4	DO2	12	0.16	2.5	20611	474	100	17272	74.35
5	DO3	12.8	0.14	3.88	18846	596	99	18456	49.82
6	DO4	11.7	0.13	2.02	18587	582	100	18940	78.52
7	DO5	12	0.16	2.13	19299	678	100	16864	79.31
8	DO6	12.4	0.04	2.98	21695	595	100	19440	64.34
9	DO7	12.5	0.32	3.67	16523	592	100	16152	58.19
10	DO8	11.5	0.07	3.67	18333	634	101	18480	88.39
11	DO9	13.1	0.19	2.44	18628	592	101	14604	33.02
12	DO10	11.7	0.25	3	15683	540	101	16628	60.28
13	DO11	13.2	0.2	3.75	18496	605	100	18476	91.86
14	DO12	11.9	0.11	1.96	19792	646	101	19212	81.88
15	DO13	12.3	0.03	2.4	20098	618	101	19608	91.02

Python's `csv` module can be used to handle files of this type.

<https://docs.python.org/3/library/csv.html> (<https://docs.python.org/3/library/csv.html>)

In [76]: 1 **import** csv

Writing CSV files

Writing files using `write` can be time-consuming.

We need to add delimiters (`,` , `' '` etc) and new line markers (`'\n'`).

```
names = ['Elena', 'Sajid', 'Tom', 'Farhad', 'Manesha']  
scores = [550, 480, 380, 305, 150]
```

```
with open('sample_data/scores_.csv', 'w') as f:
```

```
    # loop through two lists  
    for n, s in zip(names, scores):  
        f.write(n + ',' + str(s) + '\n')
```

The `writer` class from the `csv` module handles delimiters automatically.

`writerow` :

- a method belonging to the `writer` class
- writes list to csv file as row

Example: Write the high score table to a csv file `scores_.csv`

In [95]: 1

Try it yourself

Example: Use the `csv` module to write the header and first row of the high score table shown to a `.txt` file.

Hint

`.txt` file so delimiter is `' '` not `','` (default).

Specify delimiter when creating the writer object using the named argument `delimiter=' '`.

Place	Name	Score
1	Elena	550
2	Sajid	480
3	Tom	380
4	Farhad	305
5	Manesha	150

In [96]:

```
1
2
```

writerow - A word of warning!

On Windows, `writerow` introduces an additional blank row between subsequent rows.

To avoid the blank line, pass the additional named argument `newline=''` to the `open` function:

```
with open('sample_data/scores_.csv', 'w', newline='') as f:
```

Example: Write the high score table data to a csv file, `scores_.csv`

`writerows` can be used to write a data structure (containing multiple lists) to the file

```
In [97]: 1 header = ['place', 'name', 'score']
          2
          3 # data is list of lists
          4 data = [[1, 'Elena', 550],
          5             [2, 'Sajid', 480],
          6             [3, 'Tom', 380],
          7             [4, 'Farhad', 305],
          8             [5, 'Manesha', 150]
          9             ]
```

```
In [98]: 1
```

Appending CSV files

(Code structure identical to write, apart from mode specifier)

```
In [99]: 1 import csv
          2
          3 with open('sample_data/scores_.csv', 'a') as f: # open file in
          4
          5     w = csv.writer(f) # writer object
          6
          7     w.writerow([6, 'Carl', 100]) # list to row
          8     w.writerow([7, 'Theo', 105])
          9     w.writerow([8, 'Mark', 75])
         10     w.writerow([9, 'Grace', 50])
```

Reading CSV files

Reading csv files and converting them to a useful format (list of strings) can be a lengthy process!

```
In [100]: 1 with open('sample_data/scores_.csv') as f:
          2     file = list(f)                                # list of strings (
          3
          4     print(file)
          5
          6     lines = [line.strip() for line in file] # remove \n from ea
          7
          8     print(lines)
          9
         10     lists = [l.split(',') for l in lines]    # convert each line
         11
         12     print(lists)
```

```
['place,name,score\n', '1,Elena,550\n', '2,Sajid,480\n', '3,Tom,380\n', '4,Farhad,305\n', '5,Manesha,150\n', '6,Carl,100\n', '7,Theo,105\n', '8,Mark,75\n', '9,Grace,50\n']
['place,name,score', '1,Elena,550', '2,Sajid,480', '3,Tom,380', '4,Farhad,305', '5,Manesha,150', '6,Carl,100', '7,Theo,105', '8,Mark,75', '9,Grace,50']
[['place', 'name', 'score'], ['1', 'Elena', '550'], ['2', 'Sajid', '480'], ['3', 'Tom', '380'], ['4', 'Farhad', '305'], ['5', 'Manesha', '150'], ['6', 'Carl', '100'], ['7', 'Theo', '105'], ['8', 'Mark', '75'], ['9', 'Grace', '50']]
```

The writer class from the csv module removes delimiters and \n characters automatically.

Example: Read the contents of the file scores_.csv

```
In [ ]: 1
```

reader and writer objects:

- iterable but not subscriptable
- moves stream position to end of file after operation using either object

Example: Read the file scores_.csv and print the third line.

```
In [102]: 1
          2
          3
          4
          5
          6
          7
          8
          9
         10
         11
         12
         13
         14
         15
         16
         17
         18
         19
         20
         21
         22
         23
         24
         25
         26
         27
         28
         29
         30
         31
         32
         33
         34
         35
         36
         37
         38
         39
         40
         41
         42
         43
         44
         45
         46
         47
         48
         49
         50
         51
         52
         53
         54
         55
         56
         57
         58
         59
         60
         61
         62
         63
         64
         65
         66
         67
         68
         69
         70
         71
         72
         73
         74
         75
         76
         77
         78
         79
         80
         81
         82
         83
         84
         85
         86
         87
         88
         89
         90
         91
         92
         93
         94
         95
         96
         97
         98
         99
        100
        101
        102
        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

```

Reading and writing csv files

The same mode specifiers are used as for .txt files.

A `reader` and `writer` object are created.

Example: Add a new entry to the file `scores_.csv` and then print the contents.

What if we want to print the contents before adding a new line?

In [103]:

```
1 ['place', 'name', 'score']  
  ['1', 'Elena', '550']  
  ['2', 'Sajid', '480']  
  ['3', 'Tom', '380']  
  ['4', 'Farhad', '305']  
  ['5', 'Manesha', '150']  
  ['6', 'Carl', '100']  
  ['7', 'Theo', '105']  
  ['8', 'Mark', '75']  
  ['9', 'Grace', '50']  
  ['10', 'Lois', '70']
```

Summary

Functions imported from modules can shorten processes that are lengthy to produce in pure Python by importing code stored elsewhere.

`csv` : A Python modules for reading/writing delimited files

Further reading

- More ways to read and write files using packages we study later on the unit (e.g. `matplotlib`, `numpy`).
- Explore the `os` module for system-level operations (e.g. creating a new directory in your filesystem) <https://docs.python.org/3/library/os.html> (<https://docs.python.org/3/library/os.html>)
- Explore the `Pandas` package: useful for handling spreadsheet-style data https://pandas.pydata.org/docs/getting_started/index.html#getting-started (https://pandas.pydata.org/docs/getting_started/index.html#getting-started)

Extra example : Writing data as columns

Data stored as lists

```
places = [1, 2, 3, 4, 5]
names = ['Elena', 'Sajid', 'Tom', 'Farhad', 'Manesha']
scores = [550, 480, 380, 305, 150]
```

Often we want to organise our data in columns, not rows:

	A	B	C
1	place	name	score
2	1	Elena	550
3	2	Sajid	480
4	3	Tom	380
5	4	Farhad	305
6	5	Manesha	150

We can't write a column explicitly in Python, as we would in Excel, we can only write rows.

The CSV file is essentially a text file with commas to separate values.

We re-arrange the data into lists that when written to a file, will arrange the data in columns.

This can be achieved using a loop (+ list comprehension)

[An identical process can be used to to the inverse operation : we can transform imported data arranged in columns into lists so that it's easier to use in the Python program]

Example: Write places , names and scores to columns of a csv file.

```
In [104]: 1 places = [1, 2, 3, 4, 5]
          2 names = ['Elena', 'Sajid', 'Tom', 'Farhad', 'Manesha']
          3 scores = [550, 480, 380, 305, 150]
          4
          5 data = [places, names, scores]
          6
          7 with open('sample_data/scores.csv', 'w') as f: # no gap between
          8
          9     w = csv.writer(f)
         10
         11     for i in range(len(places)):
         12         w.writerow([d[i] for d in data])
         13         # OR
         14         #w.writerow([places[i], names[i], scores[i]])
         15
```

Extra example : Using zip to convert between rows and columns

Import the data from the file sample_data/scores.csv and generate a row containing the data from each column:

Alternatively, can use `zip` to transpose the data from rows to columns before writing to a CSV file.

Like when using `zip` to iterate through two lists, items from multiple lists are regrouped elementwise.

```
In [105]: 1 places = [1, 2, 3, 4, 5]
          2 names = ['Elena', 'Sajid', 'Tom', 'Farhad', 'Manesha']
          3 scores = [550, 480, 380, 305, 150]
          4
          5 data = zip(places, names, scores)
          6
          7 print(list(data)) # must be converted to a list to print, itera
```

```
[(1, 'Elena', 550), (2, 'Sajid', 480), (3, 'Tom', 380), (4, 'Farha
d', 305), (5, 'Manesha', 150)]
```

To transpose a list of lists we can use `*`.

This *unpacks* the list (removing the outer brackets).


```
In [106]: 1 data = [[1, 2, 3, 4, 5],
2           ['Elena', 'Sajid', 'Tom', 'Farhad', 'Manesha'],
3           [550, 480, 380, 305, 150]
4           ]
5
6 data_cols = list(zip(*data)) # must be converted to a list to p
7
8 print(data_cols)
```

```
[(1, 'Elena', 550), (2, 'Sajid', 480), (3, 'Tom', 380), (4, 'Farha
d', 305), (5, 'Manesha', 150)]
```

This can then be written to a .csv or .txt file

```
In [107]: 1 with open('sample_data/scores.csv', 'w', newline='') as f:
2
3           w = csv.writer(f)
4
5           w.writerow(data_cols)
```

```
In [108]: 1 with open('sample_data/scores.txt', 'w', newline='') as f:
2
3           w = csv.writer(f, delimiter=' ') # specify the delimiter
4
5           w.writerow(data_cols)
```

```
In [109]: 1 import csv
2
3 with open('sample_data/scores.csv') as f:
4
5     r = list(csv.reader(f)) # reader is iterable but not subsc
6
7     header = reader[0]      # choose first row as header
8     print(header)
9
10    data = list(zip(*r[1:])) # transpose data EXCLUDING header
11
12    print(data[0])           # place
13    print(data[1])           # name
14    print(data[2])           # score
```

```
['1', 'Elena', '550']
('2', '3', '4', '5')
('Sajid', 'Tom', 'Farhad', 'Manesha')
('480', '380', '305', '150')
```

```
In [ ]: 1
```

In []:

1