

# IPL DATA ANALYSIS PROJECT

In [244...]

```
#Import numpy
import numpy as np

#Seasons
Seasons = ["2010", "2011", "2012", "2013", "2014", "2015", "2016", "2017", "2018", "2019"]
Sdict = {"2010":0,"2011":1,"2012":2,"2013":3,"2014":4,"2015":5,"2016":6,"2017":7,"2018":8,"2019":9}

#Players
Players = ["Sachin", "Rahul", "Smith", "Sami", "Pollard", "Morris", "Samson", "Dhoni", "Kohli"]
Pdict = {"Sachin":0, "Rahul":1, "Smith":2, "Sami":3, "Pollard":4, "Morris":5, "Samson":6, "Dhoni":7, "Kohli":8}

#Salaries
Sachin_Salary = [15946875, 17718750, 19490625, 21262500, 23034375, 24806250, 25244493, 278100000, 30518750, 33290000, 36075000, 38850000, 41625000, 44400000, 47175000, 50950000, 53725000, 56500000, 59275000, 62050000, 64825000, 67500000, 70275000, 73050000, 75825000, 78500000, 81275000, 84050000, 86825000, 89500000, 92275000, 95050000, 97825000, 100500000, 103275000, 106050000, 108825000, 111500000, 114275000, 116950000, 119625000, 122300000, 124975000, 127650000, 130325000, 133000000, 135675000, 138350000, 141025000, 143695000, 146370000, 149040000, 151715000, 154385000, 157055000, 159725000, 162395000, 165065000, 167735000, 170405000, 173075000, 175745000, 178415000, 181085000, 183755000, 186425000, 189095000, 191765000, 194435000, 197105000, 199775000, 202445000, 205115000, 207785000, 210455000, 213125000, 215792500, 218460000, 221125000, 223795000, 226465000, 229135000, 231805000, 234475000, 237145000, 239815000, 242485000, 245155000, 247825000, 250495000, 253165000, 255835000, 258505000, 261175000, 263845000, 266515000, 269185000, 271855000, 274525000, 277195000, 279865000, 282535000, 285205000, 287875000, 290545000, 293215000, 295885000, 298555000, 301225000, 303895000, 306565000, 309235000, 311905000, 314575000, 317245000, 320915000, 323585000, 326255000, 328925000, 331595000, 334265000, 336935000, 339605000, 342275000, 344945000, 347615000, 350285000, 352955000, 355625000, 358295000, 360965000, 363635000, 366305000, 368975000, 371645000, 374315000, 377085000, 380755000, 383425000, 386095000, 388765000, 391435000, 394105000, 396775000, 399445000, 402115000, 404785000, 407455000, 410125000, 412795000, 415465000, 418135000, 420805000, 423475000, 426145000, 428815000, 431485000, 434155000, 436825000, 439495000, 442165000, 444835000, 447505000, 450175000, 452845000, 455515000, 458185000, 460855000, 463525000, 466195000, 468865000, 471535000, 474205000, 476875000, 479545000, 482215000, 484885000, 487555000, 490225000, 492895000, 495565000, 498235000, 500905000, 503575000, 506245000, 508915000, 511585000, 514255000, 516925000, 519595000, 522265000, 524935000, 527605000, 530275000, 532945000, 535615000, 538285000, 540955000, 543625000, 546295000, 548965000, 551635000, 554305000, 556975000, 559645000, 562315000, 565000000, 567675000, 570355000, 573035000, 575715000, 578395000, 581075000, 583755000, 586435000, 589115000, 591795000, 594475000, 597155000, 599835000, 602515000, 605195000, 607875000, 610555000, 613235000, 615915000, 618595000, 621275000, 623955000, 626635000, 629315000, 632095000, 634775000, 637455000, 640135000, 642815000, 645495000, 648175000, 650855000, 653535000, 656215000, 658895000, 661575000, 664255000, 666935000, 669615000, 672295000, 674975000, 677655000, 680335000, 683015000, 685695000, 688375000, 691055000, 693735000, 696415000, 699095000, 701775000, 704455000, 707135000, 709815000, 712495000, 715175000, 717855000, 720535000, 723215000, 725895000, 728575000, 731255000, 733935000, 736615000, 739295000, 741975000, 744655000, 747335000, 750015000, 752695000, 755375000, 758055000, 760735000, 763415000, 766095000, 768775000, 771455000, 774135000, 776815000, 779495000, 782175000, 784855000, 787535000, 790215000, 792895000, 795575000, 798255000, 800935000, 803615000, 806295000, 808975000, 811655000, 814335000, 817015000, 819695000, 822375000, 825055000, 827735000, 830415000, 833095000, 835775000, 838455000, 841135000, 843815000, 846495000, 849175000, 851855000, 854535000, 857215000, 860895000, 863575000, 866255000, 868935000, 871615000, 874295000, 876975000, 879655000, 882335000, 885015000, 887695000, 890375000, 893055000, 895735000, 898415000, 901095000, 903775000, 906455000, 909135000, 911815000, 914495000, 917175000, 920855000, 923535000, 926215000, 928895000, 931575000, 934255000, 936935000, 939615000, 942295000, 944975000, 947655000, 950335000, 953015000, 955695000, 958375000, 961055000, 963735000, 966415000, 969095000, 971775000, 974455000, 977135000, 980815000, 983495000, 986175000, 988855000, 991535000, 994215000, 996895000, 999575000, 1002250000, 1004925000, 1007600000, 1010275000, 1012950000, 1015625000, 1018300000, 1021075000, 1023750000, 1026425000, 1029100000, 1031775000, 1034450000, 1037125000, 1039795000, 1042470000, 1045145000, 1047820000, 1050495000, 1053170000, 1055845000, 1058520000, 1061195000, 1063870000, 1066545000, 1069220000, 1071895000, 1074570000, 1077245000, 1080000000, 1082775000, 1085450000, 1088125000, 1090800000, 1093475000, 1096150000, 1098825000, 1101500000, 1104175000, 1106850000, 1109525000, 1112195000, 1114870000, 1117545000, 1120215000, 1122885000, 1125555000, 1128225000, 1130895000, 1133565000, 1136235000, 1138905000, 1141575000, 1144245000, 1146915000, 1149585000, 1152255000, 1154925000, 1157595000, 1160265000, 1162935000, 1165605000, 1168275000, 1170945000, 1173615000, 1176285000, 1178955000, 1181625000, 1184295000, 1186965000, 1189635000, 1192305000, 1194975000, 1197645000, 1200315000, 1202985000, 1205655000, 1208325000, 1211000000, 1213675000, 1216345000, 1219015000, 1221685000, 1224355000, 1227025000, 1229695000, 1232365000, 1235035000, 1237705000, 1240375000, 1243045000, 1245715000, 1248385000, 1251055000, 1253725000, 1256395000, 1259065000, 1261735000, 1264405000, 1267075000, 1269745000, 1272415000, 1275085000, 1277755000, 1280425000, 1283095000, 1285765000, 1288435000, 1291105000, 1293775000, 1296445000, 1299115000, 1301785000, 1304455000, 1307125000, 1309795000, 1312465000, 1315135000, 1317805000, 1320475000, 1323145000, 1325815000, 1328485000, 1331155000, 1333825000, 1336495000, 1339165000, 1341835000, 1344505000, 1347175000, 1349845000, 1352515000, 1355185000, 1357855000, 1360525000, 1363195000, 1365865000, 1368535000, 1371205000, 1373875000, 1376545000, 1379215000, 1381885000, 1384555000, 1387225000, 1390895000, 1393565000, 1396235000, 1398905000, 1401575000, 1404245000, 1406915000, 1409585000, 1412255000, 1414925000, 1417595000, 1420265000, 1422935000, 1425605000, 1428275000, 1430945000, 1433615000, 1436285000, 1438955000, 1441625000, 1444295000, 1446965000, 1449635000, 1452305000, 1454975000, 1457645000, 1460315000, 1462985000, 1465655000, 1468325000, 1471000000, 1473675000, 1476345000, 1479015000, 1481685000, 1484355000, 1487025000, 1490695000, 1493365000, 1496035000, 1498705000, 1501375000, 1504045000, 1506715000, 1509385000, 1512055000, 1514725000, 1517395000, 1520065000, 1522735000, 1525405000, 1528075000, 1530745000, 1533415000, 1536085000, 1538755000, 1541425000, 1544095000, 1546765000, 1549435000, 1552105000, 1554775000, 1557445000, 1560115000, 1562785000, 1565455000, 1568125000, 1570795000, 1573465000, 1576135000, 1578805000, 1581475000, 1584145000, 1586815000, 1589485000, 1592155000, 1594825000, 1597495000, 1599165000, 1601835000, 1604505000, 1607175000, 1609845000, 1612515000, 1615185000, 1617855000, 1620525000, 1623195000, 1625865000, 1628535000, 1631205000, 1633875000, 1636545000, 1639215000, 1641885000, 1644555000, 1647225000, 1650895000, 1653565000, 1656235000, 1658905000, 1661575000, 1664245000, 1666915000, 1669585000, 1672255000, 1674925000, 1677595000, 1680265000, 1682935000, 1685605000, 1688275000, 1690945000, 1693615000, 1696285000, 1698955000, 1701625000, 1704295000, 1706965000, 1709635000, 1712305000, 1714975000, 1717645000, 1720315000, 1722985000, 1725655000, 1728325000, 1731000000, 1733675000, 1736345000, 1739015000, 1741685000, 1744355000, 1747025000, 1750695000, 1753365000, 1756035000, 1758705000, 1761375000, 1764045000, 1766715000, 1769385000, 1772055000, 1774725000, 1777395000, 1780065000, 1782735000, 1785405000, 1788075000, 1790745000, 1793415000, 1796085000, 1798755000, 1801425000, 1804095000, 1806765000, 1809435000, 1812105000, 1814775000, 1817445000, 1820115000, 1822785000, 1825455000, 1828125000, 1830795000, 1833465000, 1836135000, 1838805000, 1841475000, 1844145000, 1846815000, 1849485000, 1852155000, 1854825000, 1857495000, 1860165000, 1862835000, 1865505000, 1868175000, 1870845000, 1873515000, 1876185000, 1878855000, 1881525000, 1884195000, 1886865000, 1889535000, 1892205000, 1894875000, 1897545000, 1900215000, 1902885000, 1905555000, 1908225000, 1910895000, 1913565000, 1916235000, 1918905000, 1921575000, 1924245000, 1926915000, 1929585000, 1932255000, 1934925000, 1937595000, 1940265000, 1942935000, 1945605000, 1948275000, 1950945000, 1953615000, 1956285000, 1958955000, 1961625000, 1964295000, 1966965000, 1969635000, 1972305000, 1974975000, 1977645000, 1980315000, 1982985000, 1985655000, 1988325000, 1991000000, 1993675000, 1996345000, 1999015000, 2001685000, 2004355000, 2007025000, 2009695000, 2012365000, 2015035000, 2017705000, 2020375000, 2023045000, 2025715000, 2028385000, 2031055000, 2033725000, 2036395000, 2039065000, 2041735000, 2044405000, 2047075000, 2049745000, 2052415000, 2055085000, 2057755000, 2060425000, 2063095000, 2065765000, 2068435000, 2071105000, 2073775000, 2076445000, 2079115000, 2081785000, 2084455000, 2087125000, 2090795000, 2093465000, 2096135000, 2098805000, 2101475000, 2104145000, 2106815000, 2109485000, 2112155000, 2114825000, 2117495000, 2120165000, 2122835000, 2125505000, 2128175000, 2130845000, 2133515000, 2136185000, 2138855000, 2141525000, 2144195000, 2146865000, 2149535000, 2152205000, 2154875000, 2157545000, 2160215000, 2162885000, 2165555000, 2168225000, 2170895000, 2173565000, 2176235000, 2178905000, 2181575000, 2184245000, 2186915000, 2189585000, 2192255000, 2194925000, 2197595000, 2200265000, 2202935000, 2205605000, 2208275000, 2210945000, 2213615000, 2216285000, 2218955000, 2221625000, 2224295000, 2226965000, 2229635000, 2232305000, 2234975000, 2237645000, 2240315000, 2242985000, 2245655000, 2248325000, 2251000000, 2253675000, 2256345000, 2259015000, 2261685000, 2264355000, 2267025000, 2269695000, 2272365000, 2275035000, 2277705000, 2280375000, 2283045000, 2285715000, 2288385000, 2291055000, 2293725000, 2296395000, 2299065000, 2301735000, 2304405000, 2307075000, 2310745000, 2313415000, 2316085000, 2318755000, 2321425000, 2324095000, 2326765000, 2329435000, 2332105000, 2334775000, 2337445000, 2340115000, 2342785000, 2345455000, 2348125000, 2350795000, 2353465000, 2356135000, 2358805000, 2361475000, 2364145000, 2366815000, 2369485000, 2372155000, 2374825000, 2377495000, 2380165000, 2382835000, 2385505000, 2388175000, 2390845000, 2393515000, 2396185000, 2398855000, 2401525000, 2404195000, 2406865000, 2409535000, 2412205000, 2414875000, 2417545000, 2420215000, 2422885000, 2425555000, 2428225000, 2430895000, 2433565000, 2436235000, 2438905000, 2441575000, 2444245000, 2446915000, 2449585000, 2452255000, 2454925000, 2457595000, 2460265000, 2462935000, 2465605000, 2468275000, 2470945000, 2473615000, 2476285000, 2478955000, 2481625000, 2484295000, 2486965000, 2489635000, 2492305000, 2494975000, 2497645000, 2500315000, 2502985000, 2505655000, 2508325000, 2511000000, 2513675000, 2516345000, 2519015000, 2521685000, 2524355000, 2527025000, 2529695000, 2532365000, 2535035000, 2537705000, 2540375000, 2543045000, 2545715000, 2548385000
```

```
#Matrix
Points = np.array([Sachin PTS, Rahul PTS, Smith PTS, Sami PTS, Pollard PTS, Morris_
```

In [246... Salary # martrix format

```
Out[246... array([[15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
       25244493, 27849149, 30453805, 23500000],
      [12000000, 12744189, 13488377, 14232567, 14976754, 16324500,
       18038573, 19752645, 21466718, 23180790],
      [ 4621800, 5828090, 13041250, 14410581, 15779912, 14500000,
       16022500, 17545000, 19067500, 20644400],
      [ 3713640, 4694041, 13041250, 14410581, 15779912, 17149243,
       18518574, 19450000, 22407474, 22458000],
      [ 4493160, 4806720, 6061274, 13758000, 15202590, 16647180,
       18091770, 19536360, 20513178, 21436271],
      [ 3348000, 4235220, 12455000, 14410581, 15779912, 14500000,
       16022500, 17545000, 19067500, 20644400],
      [ 3144240, 3380160, 3615960, 4574189, 13520500, 14940153,
       16359805, 17779458, 18668431, 20068563],
      [ 0, 0, 4171200, 4484040, 4796880, 6053663,
       15506632, 16669630, 17832627, 18995624],
      [ 0, 0, 0, 4822800, 5184480, 5546160,
       6993708, 16402500, 17632688, 18862875],
      [ 3031920, 3841443, 13041250, 14410581, 15779912, 14200000,
       15691000, 17182000, 18673000, 15000000]])
```

In [248... # Building your first matrix -  
Games

```
Out[248... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
       [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
       [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
       [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
       [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
       [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
       [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
       [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
       [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
       [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]]))
```

In [250... Points

```
Out[250... array([[2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782],
       [1653, 1426, 1779, 1688, 1619, 1312, 1129, 1170, 1245, 1154],
       [2478, 2132, 2250, 2304, 2258, 2111, 1683, 2036, 2089, 1743],
       [2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
       [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
       [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928],
       [1258, 1104, 1684, 1781, 841, 1268, 1189, 1186, 1185, 1564],
       [ 903,  903, 1624, 1871, 2472, 2161, 1850, 2280, 2593, 686],
       [ 597,  597,  597, 1361, 1619, 2026,  852,  0, 159,  904],
       [2040, 1397, 1254, 2386, 2045, 1941, 1082, 1463, 1028, 1331]]))
```

In [252... mydata = np.arange(0,20)
print(mydata)

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

In [253...]: `np.reshape(mydata,(4,5)) # 5 rows & 4 columns`

Out[253...]: `array([[ 0, 1, 2, 3, 4],  
 [ 5, 6, 7, 8, 9],  
 [10, 11, 12, 13, 14],  
 [15, 16, 17, 18, 19]])`

In [254...]: `mydata`

Out[254...]: `array([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,  
 17, 18, 19])`

In [255...]: `#np.reshape(mydata,(5,4), order = 'c') #'C' means to read / write the elements using column-major order`  
`MATR1 = np.reshape(mydata, (5,4), order = 'c')`  
`MATR1`

Out[255...]: `array([[ 0, 1, 2, 3],  
 [ 4, 5, 6, 7],  
 [ 8, 9, 10, 11],  
 [12, 13, 14, 15],  
 [16, 17, 18, 19]])`

In [256...]: `MATR1`

Out[256...]: `array([[ 0, 1, 2, 3],  
 [ 4, 5, 6, 7],  
 [ 8, 9, 10, 11],  
 [12, 13, 14, 15],  
 [16, 17, 18, 19]])`

In [257...]: `# If i want to get only no.3`  
`MATR1[4,3]`

Out[257...]: `19`

In [258...]: `MATR1[3,3]`

Out[258...]: `15`

In [259...]: `MATR1`

Out[259...]: `array([[ 0, 1, 2, 3],  
 [ 4, 5, 6, 7],  
 [ 8, 9, 10, 11],  
 [12, 13, 14, 15],  
 [16, 17, 18, 19]])`

In [260...]: `MATR1[-3,-1]`

Out[260...]: `11`

In [261...]: `MATR1`

```
Out[261... array([[ 0,  1,  2,  3],
                  [ 4,  5,  6,  7],
                  [ 8,  9, 10, 11],
                  [12, 13, 14, 15],
                  [16, 17, 18, 19]])
```

```
In [262... mydata
```

```
Out[262... array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
                  17, 18, 19])
```

```
In [263... MATR2 = np.reshape(mydata, (5,4), order = 'F') # reshape behaviour are - 'C', 'F', '
MATR2
```

```
Out[263... array([[ 0,  5, 10, 15],
                  [ 1,  6, 11, 16],
                  [ 2,  7, 12, 17],
                  [ 3,  8, 13, 18],
                  [ 4,  9, 14, 19]])
```

```
In [264... MATR2[4,3]
```

```
Out[264... 19
```

```
In [265... MATR2[0,2]
```

```
Out[265... 10
```

```
In [266... MATR2[0:2]
```

```
Out[266... array([[ 0,  5, 10, 15],
                  [ 1,  6, 11, 16]])
```

```
In [267... MATR2
```

```
Out[267... array([[ 0,  5, 10, 15],
                  [ 1,  6, 11, 16],
                  [ 2,  7, 12, 17],
                  [ 3,  8, 13, 18],
                  [ 4,  9, 14, 19]])
```

```
In [268... MATR2[1:2]
```

```
Out[268... array([[ 1,  6, 11, 16]])
```

```
In [269... MATR2[1,2]
```

```
Out[269... 11
```

```
In [270... MATR2
```

```
Out[270... array([[ 0,  5, 10, 15],  
                  [ 1,  6, 11, 16],  
                  [ 2,  7, 12, 17],  
                  [ 3,  8, 13, 18],  
                  [ 4,  9, 14, 19]])
```

```
In [271... MATR2[-2,-1]
```

```
Out[271... 18
```

```
In [272... MATR2[-3,-3]
```

```
Out[272... 7
```

```
In [273... MATR2
```

```
Out[273... array([[ 0,  5, 10, 15],  
                  [ 1,  6, 11, 16],  
                  [ 2,  7, 12, 17],  
                  [ 3,  8, 13, 18],  
                  [ 4,  9, 14, 19]])
```

```
In [274... MATR2[0:2]
```

```
Out[274... array([[ 0,  5, 10, 15],  
                  [ 1,  6, 11, 16]])
```

```
In [275... mydata
```

```
Out[275... array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,  
                  17, 18, 19])
```

```
In [276... MATR3 = np.reshape(mydata, (5,4), order = 'A')  
MATR3
```

```
Out[276... array([[ 0,  1,  2,  3],  
                  [ 4,  5,  6,  7],  
                  [ 8,  9, 10, 11],  
                  [12, 13, 14, 15],  
                  [16, 17, 18, 19]])
```

```
In [277... MATR2 ## F shaped
```

```
Out[277... array([[ 0,  5, 10, 15],  
                  [ 1,  6, 11, 16],  
                  [ 2,  7, 12, 17],  
                  [ 3,  8, 13, 18],  
                  [ 4,  9, 14, 19]])
```

```
In [278... MATR1 # C shaped
```

```
Out[278... array([[ 0,  1,  2,  3],
                  [ 4,  5,  6,  7],
                  [ 8,  9, 10, 11],
                  [12, 13, 14, 15],
                  [16, 17, 18, 19]])
```

```
In [279... a1 = ['welcome', 'to', 'datascience']
a2 = ['required', 'hard', 'work' ]
a3 = [1,2,3]
```

```
In [280... [a1,a2,a3] # List same datatype
```

```
Out[280... [['welcome', 'to', 'datascience'], ['required', 'hard', 'work'], [1, 2, 3]]
```

```
In [281... np.array([a1,a2,a3]) # u11 - unicode 11 characer : 3*3 matrix
```

```
Out[281... array([['welcome', 'to', 'datascience'],
                  ['required', 'hard', 'work'],
                  ['1', '2', '3']], dtype='<U11')
```

```
In [282... Games
```

```
Out[282... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
                  [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
                  [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
                  [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
                  [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
                  [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
                  [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
                  [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
                  [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
                  [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

```
In [283... Games[0]
```

```
Out[283... array([80, 77, 82, 82, 73, 82, 58, 78, 6, 35])
```

```
In [284... Games[5]
```

```
Out[284... array([70, 69, 67, 77, 70, 77, 57, 74, 79, 44])
```

```
In [285... Games[0:5]
```

```
Out[285... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
                  [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
                  [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
                  [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
                  [82, 82, 82, 79, 82, 78, 54, 76, 71, 41]])
```

```
In [286... Games[0,5]
```

```
Out[286... 82
```

```
In [287... Games[0,2]
```

```
Out[287... 82
```

```
In [288... Games
```

```
Out[288... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],  
[82, 57, 82, 79, 76, 72, 60, 72, 79, 80],  
[79, 78, 75, 81, 76, 79, 62, 76, 77, 69],  
[80, 65, 77, 66, 69, 77, 55, 67, 77, 40],  
[82, 82, 82, 79, 82, 78, 54, 76, 71, 41],  
[70, 69, 67, 77, 70, 77, 57, 74, 79, 44],  
[78, 64, 80, 78, 45, 80, 60, 70, 62, 82],  
[35, 35, 80, 74, 82, 78, 66, 81, 81, 27],  
[40, 40, 40, 81, 78, 81, 39, 0, 10, 51],  
[75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

```
In [289... Games[0:2]
```

```
Out[289... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],  
[82, 57, 82, 79, 76, 72, 60, 72, 79, 80]])
```

```
In [290... Games
```

```
Out[290... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],  
[82, 57, 82, 79, 76, 72, 60, 72, 79, 80],  
[79, 78, 75, 81, 76, 79, 62, 76, 77, 69],  
[80, 65, 77, 66, 69, 77, 55, 67, 77, 40],  
[82, 82, 82, 79, 82, 78, 54, 76, 71, 41],  
[70, 69, 67, 77, 70, 77, 57, 74, 79, 44],  
[78, 64, 80, 78, 45, 80, 60, 70, 62, 82],  
[35, 35, 80, 74, 82, 78, 66, 81, 81, 27],  
[40, 40, 40, 81, 78, 81, 39, 0, 10, 51],  
[75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

```
In [291... Games[1:2]
```

```
Out[291... array([[82, 57, 82, 79, 76, 72, 60, 72, 79, 80]])
```

```
In [292... Games[2]
```

```
Out[292... array([79, 78, 75, 81, 76, 79, 62, 76, 77, 69])
```

```
In [293... Games
```

```
Out[293... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],  
[82, 57, 82, 79, 76, 72, 60, 72, 79, 80],  
[79, 78, 75, 81, 76, 79, 62, 76, 77, 69],  
[80, 65, 77, 66, 69, 77, 55, 67, 77, 40],  
[82, 82, 82, 79, 82, 78, 54, 76, 71, 41],  
[70, 69, 67, 77, 70, 77, 57, 74, 79, 44],  
[78, 64, 80, 78, 45, 80, 60, 70, 62, 82],  
[35, 35, 80, 74, 82, 78, 66, 81, 81, 27],  
[40, 40, 40, 81, 78, 81, 39, 0, 10, 51],  
[75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

```
In [294... Games[2,8]
```

Out[294... 77

In [295... Games

```
Out[295... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
   [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
   [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
   [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
   [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
   [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
   [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
   [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
   [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
   [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

In [296... Games[-3:-1]

```
Out[296... array([[35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
   [40, 40, 40, 81, 78, 81, 39, 0, 10, 51]])
```

In [297... Games[-3,-1]

Out[297... 27

In [298... Points

```
Out[298... array([[2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782],
   [1653, 1426, 1779, 1688, 1619, 1312, 1129, 1170, 1245, 1154],
   [2478, 2132, 2250, 2304, 2258, 2111, 1683, 2036, 2089, 1743],
   [2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
   [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
   [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928],
   [1258, 1104, 1684, 1781, 841, 1268, 1189, 1186, 1185, 1564],
   [903, 903, 1624, 1871, 2472, 2161, 1850, 2280, 2593, 686],
   [597, 597, 597, 1361, 1619, 2026, 852, 0, 159, 904],
   [2040, 1397, 1254, 2386, 2045, 1941, 1082, 1463, 1028, 1331]])
```

In [299... Points[0]

```
Out[299... array([2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782])
```

In [300... Points

```
Out[300... array([[2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782],
   [1653, 1426, 1779, 1688, 1619, 1312, 1129, 1170, 1245, 1154],
   [2478, 2132, 2250, 2304, 2258, 2111, 1683, 2036, 2089, 1743],
   [2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
   [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
   [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928],
   [1258, 1104, 1684, 1781, 841, 1268, 1189, 1186, 1185, 1564],
   [903, 903, 1624, 1871, 2472, 2161, 1850, 2280, 2593, 686],
   [597, 597, 597, 1361, 1619, 2026, 852, 0, 159, 904],
   [2040, 1397, 1254, 2386, 2045, 1941, 1082, 1463, 1028, 1331]])
```

In [301... Points[6,1]

Out[301... 1104

In [302... Points[3:6]

```
Out[302... array([[2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
   [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
   [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928]])
```

In [303... Points

```
Out[303... array([[2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782],
   [1653, 1426, 1779, 1688, 1619, 1312, 1129, 1170, 1245, 1154],
   [2478, 2132, 2250, 2304, 2258, 2111, 1683, 2036, 2089, 1743],
   [2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
   [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
   [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928],
   [1258, 1104, 1684, 1781, 841, 1268, 1189, 1186, 1185, 1564],
   [903, 903, 1624, 1871, 2472, 2161, 1850, 2280, 2593, 686],
   [597, 597, 597, 1361, 1619, 2026, 852, 0, 159, 904],
   [2040, 1397, 1254, 2386, 2045, 1941, 1082, 1463, 1028, 1331]])
```

In [304... Points[-6,-1]

Out[304... 646

In [305... ##### DICTIONARY #####

```
# dict does not maintain the order

dict1 = {'key1': 'val1', 'key2': 'val2', 'key3': 'val3'}
```

In [306... dict1

Out[306... {'key1': 'val1', 'key2': 'val2', 'key3': 'val3'}

In [307... dict1['key2']

Out[307... 'val2'

In [308... dict2 = {'bang': 2, 'hyd': 'we are hear', 'pune': True}

In [309... dict2

Out[309... {'bang': 2, 'hyd': 'we are hear', 'pune': True}

In [310... dict3 = {'Germany': 'I have been here', 'France': 2, 'Spain': True}

In [311... dict3

Out[311... {'Germany': 'I have been here', 'France': 2, 'Spain': True}

In [312... dict3['Germany']

Out[312... 'I have been here'

In [313... # if you check theat dataset seasons & players are dictionary type of data  
# if you Look at the pdict players names are key part:nos are the values  
# dictionary can guide us which player at which Level and which row  
# main advantage of the dictionary is we dont required to count which no row which

In [314... Games

Out[314... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],  
[82, 57, 82, 79, 76, 72, 60, 72, 79, 80],  
[79, 78, 75, 81, 76, 79, 62, 76, 77, 69],  
[80, 65, 77, 66, 69, 77, 55, 67, 77, 40],  
[82, 82, 82, 79, 82, 78, 54, 76, 71, 41],  
[70, 69, 67, 77, 70, 77, 57, 74, 79, 44],  
[78, 64, 80, 78, 45, 80, 60, 70, 62, 82],  
[35, 35, 80, 74, 82, 78, 66, 81, 81, 27],  
[40, 40, 40, 81, 78, 81, 39, 0, 10, 51],  
[75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])

In [315... Pdict

Out[315... {'Sachin': 0,  
'Rahul': 1,  
'Smith': 2,  
'Sami': 3,  
'Pollard': 4,  
'Morris': 5,  
'Samson': 6,  
'Dhoni': 7,  
'Kohli': 8,  
'Sky': 9}

In [316... # how do i know player kobe bryant is at

Pdict['Sachin']

Out[316... 0

In [317... Games[0]

Out[317... array([80, 77, 82, 82, 73, 82, 58, 78, 6, 35])

In [318... Games

```
Out[318... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
   [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
   [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
   [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
   [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
   [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
   [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
   [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
   [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
   [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]]))
```

In [319... Pdict['Rahul']]

Out[319... 1

In [320... Games[1]]

Out[320... array([82, 57, 82, 79, 76, 72, 60, 72, 79, 80])

## Games

In [322... Games[Pdict['Rahul']]])

Out[322... array([82, 57, 82, 79, 76, 72, 60, 72, 79, 80])

In [323... Points]

```
Out[323... array([[2832, 2430, 2323, 2201, 1970, 2078, 1616, 2133, 83, 782],
   [1653, 1426, 1779, 1688, 1619, 1312, 1129, 1170, 1245, 1154],
   [2478, 2132, 2250, 2304, 2258, 2111, 1683, 2036, 2089, 1743],
   [2122, 1881, 1978, 1504, 1943, 1970, 1245, 1920, 2112, 966],
   [1292, 1443, 1695, 1624, 1503, 1784, 1113, 1296, 1297, 646],
   [1572, 1561, 1496, 1746, 1678, 1438, 1025, 1232, 1281, 928],
   [1258, 1104, 1684, 1781, 841, 1268, 1189, 1186, 1185, 1564],
   [903, 903, 1624, 1871, 2472, 2161, 1850, 2280, 2593, 686],
   [597, 597, 597, 1361, 1619, 2026, 852, 0, 159, 904],
   [2040, 1397, 1254, 2386, 2045, 1941, 1082, 1463, 1028, 1331]]))
```

In [324... Salary]

```
Out[324... array([[15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000],
   [12000000, 12744189, 13488377, 14232567, 14976754, 16324500,
   18038573, 19752645, 21466718, 23180790],
   [ 4621800, 5828090, 13041250, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3713640, 4694041, 13041250, 14410581, 15779912, 17149243,
   18518574, 19450000, 22407474, 22458000],
   [ 4493160, 4806720, 6061274, 13758000, 15202590, 16647180,
   18091770, 19536360, 20513178, 21436271],
   [ 3348000, 4235220, 12455000, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3144240, 3380160, 3615960, 4574189, 13520500, 14940153,
   16359805, 17779458, 18668431, 20068563],
   [ 0, 0, 4171200, 4484040, 4796880, 6053663,
   15506632, 16669630, 17832627, 18995624],
   [ 0, 0, 4822800, 5184480, 5546160,
   6993708, 16402500, 17632688, 18862875],
   [ 3031920, 3841443, 13041250, 14410581, 15779912, 14200000,
   15691000, 17182000, 18673000, 15000000]])
```

In [325... Salary[2,4]

Out[325... 15779912

In [326... Salary

```
Out[326... array([[15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000],
   [12000000, 12744189, 13488377, 14232567, 14976754, 16324500,
   18038573, 19752645, 21466718, 23180790],
   [ 4621800, 5828090, 13041250, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3713640, 4694041, 13041250, 14410581, 15779912, 17149243,
   18518574, 19450000, 22407474, 22458000],
   [ 4493160, 4806720, 6061274, 13758000, 15202590, 16647180,
   18091770, 19536360, 20513178, 21436271],
   [ 3348000, 4235220, 12455000, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3144240, 3380160, 3615960, 4574189, 13520500, 14940153,
   16359805, 17779458, 18668431, 20068563],
   [ 0, 0, 4171200, 4484040, 4796880, 6053663,
   15506632, 16669630, 17832627, 18995624],
   [ 0, 0, 4822800, 5184480, 5546160,
   6993708, 16402500, 17632688, 18862875],
   [ 3031920, 3841443, 13041250, 14410581, 15779912, 14200000,
   15691000, 17182000, 18673000, 15000000]])
```

In [395... Salary[Pdict['Sky']][Sdict['2019']]

Out[395... 15000000

In [445... Salary

```
Out[445... array([[15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000],
  [12000000, 12744189, 13488377, 14232567, 14976754, 16324500,
  18038573, 19752645, 21466718, 23180790],
  [ 4621800, 5828090, 13041250, 14410581, 15779912, 14500000,
  16022500, 17545000, 19067500, 20644400],
  [ 3713640, 4694041, 13041250, 14410581, 15779912, 17149243,
  18518574, 19450000, 22407474, 22458000],
  [ 4493160, 4806720, 6061274, 13758000, 15202590, 16647180,
  18091770, 19536360, 20513178, 21436271],
  [ 3348000, 4235220, 12455000, 14410581, 15779912, 14500000,
  16022500, 17545000, 19067500, 20644400],
  [ 3144240, 3380160, 3615960, 4574189, 13520500, 14940153,
  16359805, 17779458, 18668431, 20068563],
  [ 0, 0, 4171200, 4484040, 4796880, 6053663,
  15506632, 16669630, 17832627, 18995624],
  [ 0, 0, 0, 4822800, 5184480, 5546160,
  6993708, 16402500, 17632688, 18862875],
  [ 3031920, 3841443, 13041250, 14410581, 15779912, 14200000,
  15691000, 17182000, 18673000, 15000000]])
```

In [447... Games

```
Out[447... array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
   [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
   [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
   [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
   [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
   [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
   [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
   [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
   [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
   [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

In [448... Salary/Games

```
Out[448... array([[ 199335.9375 ,  230113.63636364,  237690.54878049,
   259298.7804878 ,  315539.38356164,  302515.24390244,
   435249.87931034,  357040.37179487,  5075634.16666667,
   671428.57142857],
 [ 146341.46341463,  223582.26315789,  164492.40243902,
  180159.07594937,  197062.55263158,  226729.16666667,
  300642.88333333,  274342.29166667,  271730.60759494,
  289759.875     ],
 [ 58503.79746835,  74719.1025641 ,  173883.33333333,
  177908.40740741,  207630.42105263,  183544.30379747,
  258427.41935484,  230855.26315789,  247629.87012987,
  299194.20289855],
 [ 46420.5       ,  72216.01538462,  169366.88311688,
  218342.13636364,  228694.37681159,  222717.44155844,
  336701.34545455,  290298.50746269,  291006.15584416,
  561450.      ],
 [ 54794.63414634,  58618.53658537,  73917.97560976,
  174151.89873418,  185397.43902439,  213425.38461538,
  335032.77777778,  257057.36842105,  288918.      ,
  522835.87804878],
 [ 47828.57142857,  61380.        ,  185895.52238806,
  187150.4025974 ,  225427.31428571,  188311.68831169,
  281096.49122807,  237094.59459459,  241360.75949367,
  469190.90909091],
 [ 40310.76923077,  52815.        ,  45199.5       ,
  58643.44871795,  300455.55555556,  186751.9125       ,
  272663.41666667,  253992.25714286,  301103.72580645,
  244738.57317073],
 [ 0.        ,  0.        ,  52140.        ,
  60595.13513514,  58498.53658537,  77611.06410256,
  234948.96969697,  205797.90123457,  220155.88888889,
  703541.62962963],
 [ 0.        ,  0.        ,  0.        ,
  59540.74074074,  66467.69230769,  68471.11111111,
  179325.84615385,  inf,  1763268.8       ,
  369860.29411765],
 [ 40425.6       ,  75322.41176471,  255710.78431373,
  182412.41772152,  204933.92207792,  186842.10526316,
  320224.48979592,  249014.49275362,  345796.2962963 ,
  241935.48387097]])
```

In [449... np.round(Salary/Games)

```
Out[449... array([[ 199336.,  230114.,  237691.,  259299.,  315539.,  302515.,
   435250.,  357040.,  5075634.,  671429.],
   [ 146341.,  223582.,  164492.,  180159.,  197063.,  226729.,
   300643.,  274342.,  271731.,  289760.],
   [ 58504.,  74719.,  173883.,  177908.,  207630.,  183544.,
   258427.,  230855.,  247630.,  299194.],
   [ 46420.,  72216.,  169367.,  218342.,  228694.,  222717.,
   336701.,  290299.,  291006.,  561450.],
   [ 54795.,  58619.,  73918.,  174152.,  185397.,  213425.,
   335033.,  257057.,  288918.,  522836.],
   [ 47829.,  61380.,  185896.,  187150.,  225427.,  188312.,
   281096.,  237095.,  241361.,  469191.],
   [ 40311.,  52815.,  45200.,  58643.,  300456.,  186752.,
   272663.,  253992.,  301104.,  244739.],
   [ 0.,  0.,  52140.,  60595.,  58499.,  77611.,
   234949.,  205798.,  220156.,  703542.],
   [ 0.,  0.,  0.,  59541.,  66468.,  68471.,
   179326.,  inf,  1763269.,  369860.],
   [ 40426.,  75322.,  255711.,  182412.,  204934.,  186842.,
   320224.,  249014.,  345796.,  241935.]])
```

```
In [450... import warnings
warnings.filterwarnings('ignore')
#np.round(FieldGoals/Games)
#FieldGoals/Games # this matrix is lot of decimal points yo can not round
#round()
```

```
In [451... ## --- First visualization ----##
```

```
In [452... import numpy as np
import matplotlib.pyplot as plt
```

```
In [453... %matplotlib inline # keep the plot inside jupyter nots insted of getting in other s
UsageError: unrecognized arguments: # keep the plot inside jupyter nots insted of ge
tting in other screen
```

```
In [454... Salary
```

```
Out[454... array([[15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000],
   [12000000, 12744189, 13488377, 14232567, 14976754, 16324500,
   18038573, 19752645, 21466718, 23180790],
   [ 4621800, 5828090, 13041250, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3713640, 4694041, 13041250, 14410581, 15779912, 17149243,
   18518574, 19450000, 22407474, 22458000],
   [ 4493160, 4806720, 6061274, 13758000, 15202590, 16647180,
   18091770, 19536360, 20513178, 21436271],
   [ 3348000, 4235220, 12455000, 14410581, 15779912, 14500000,
   16022500, 17545000, 19067500, 20644400],
   [ 3144240, 3380160, 3615960, 4574189, 13520500, 14940153,
   16359805, 17779458, 18668431, 20068563],
   [ 0, 0, 4171200, 4484040, 4796880, 6053663,
   15506632, 16669630, 17832627, 18995624],
   [ 0, 0, 0, 4822800, 5184480, 5546160,
   6993708, 16402500, 17632688, 18862875],
   [ 3031920, 3841443, 13041250, 14410581, 15779912, 14200000,
   15691000, 17182000, 18673000, 15000000]])
```

In [455... Salary[0]

```
Out[455... array([15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000])
```

In [531... plt.plot(Salary[0])

```
Out[531... []
```

In [533... plt.plot(Salary[0], c='r', ls='--')

```
Out[533... [

```

In [535... plt.plot(Salary[0], c='k', ls=':')

```
Out[535... [

```

In [537... %matplotlib inline  
plt.rcParams['figure.figsize'] = 7,3

In [539... plt.plot(Salary[0], c='Blue', ls = 'dashed')

```
Out[539... [

```

In [541... plt.plot(Salary[0], c='Green', ls = '--', marker = 's') # s - squares

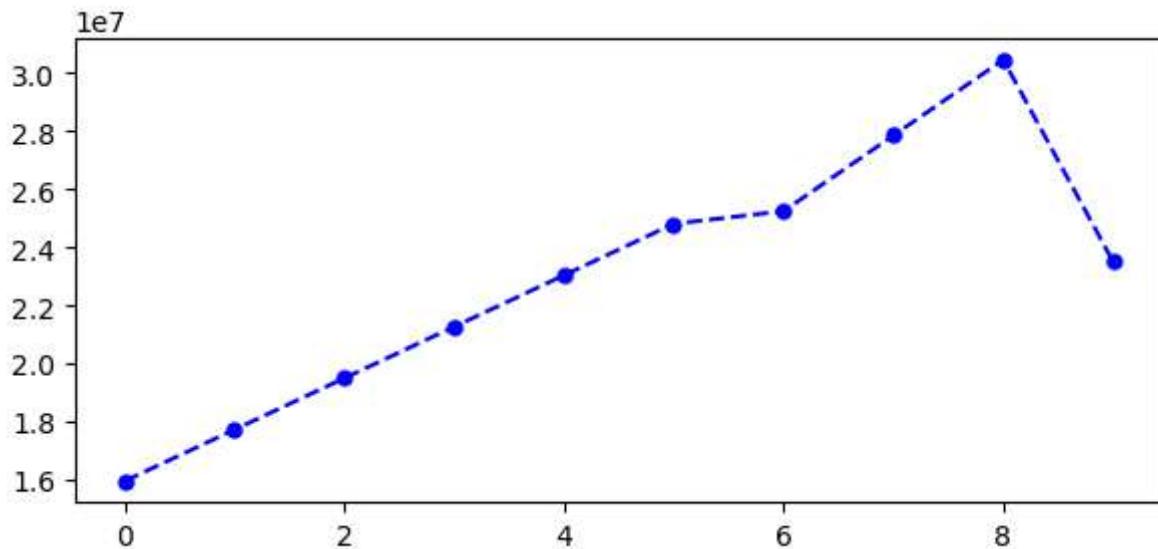
```
Out[541... [

```

In [543... %matplotlib inline  
plt.rcParams['figure.figsize'] = 7,3 #runtime configuration parameter

In [563... plt.plot(Salary[0], c='blue', ls = '--', marker = 'o', ms = 5)

```
plt.show()
```



```
In [464...]: list(range(0,10))
```

```
Out[464...]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

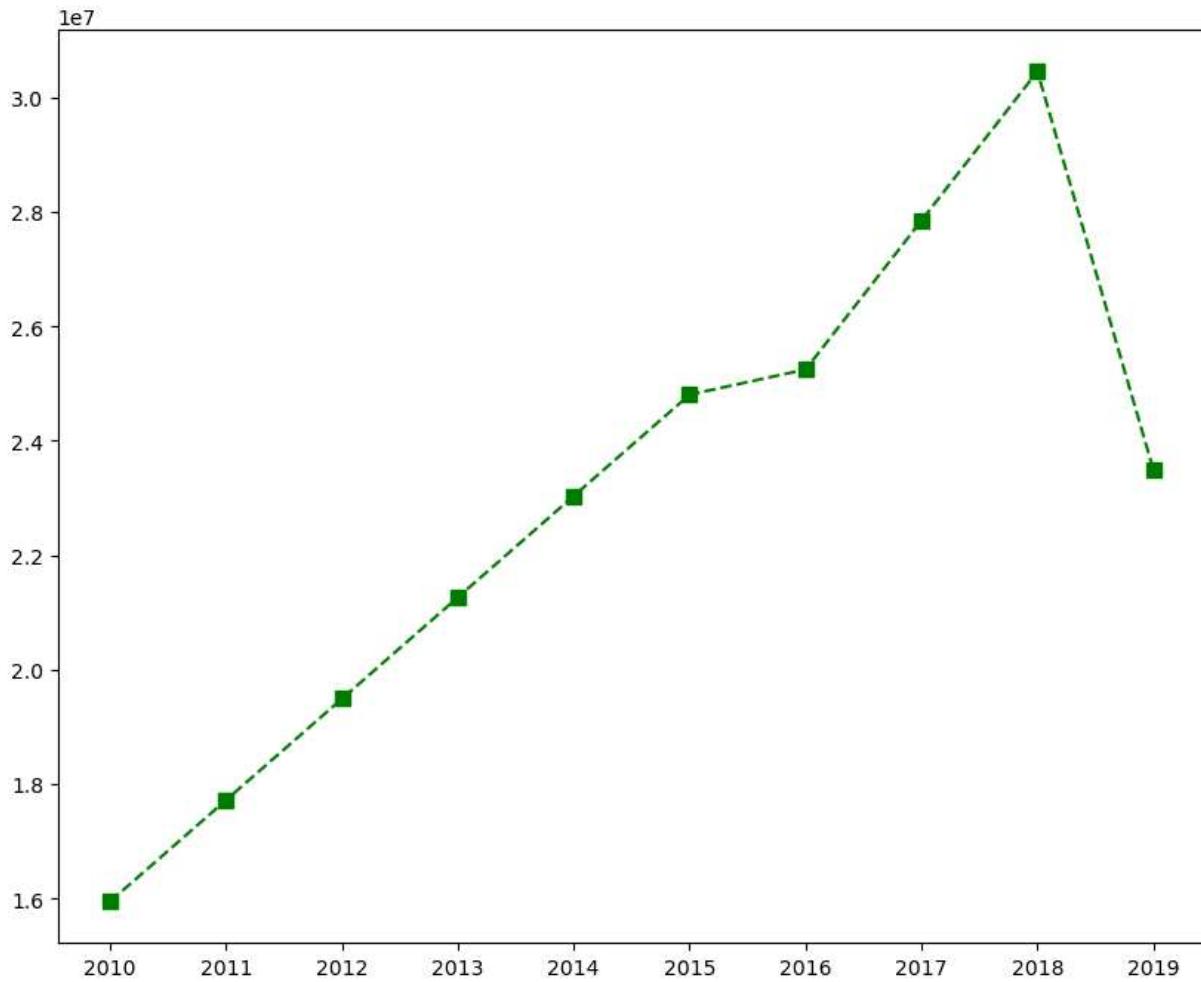
```
In [565...]: Sdict
```

```
Out[565...]: {'2010': 0,
 '2011': 1,
 '2012': 2,
 '2013': 3,
 '2014': 4,
 '2015': 5,
 '2016': 6,
 '2017': 7,
 '2018': 8,
 '2019': 9}
```

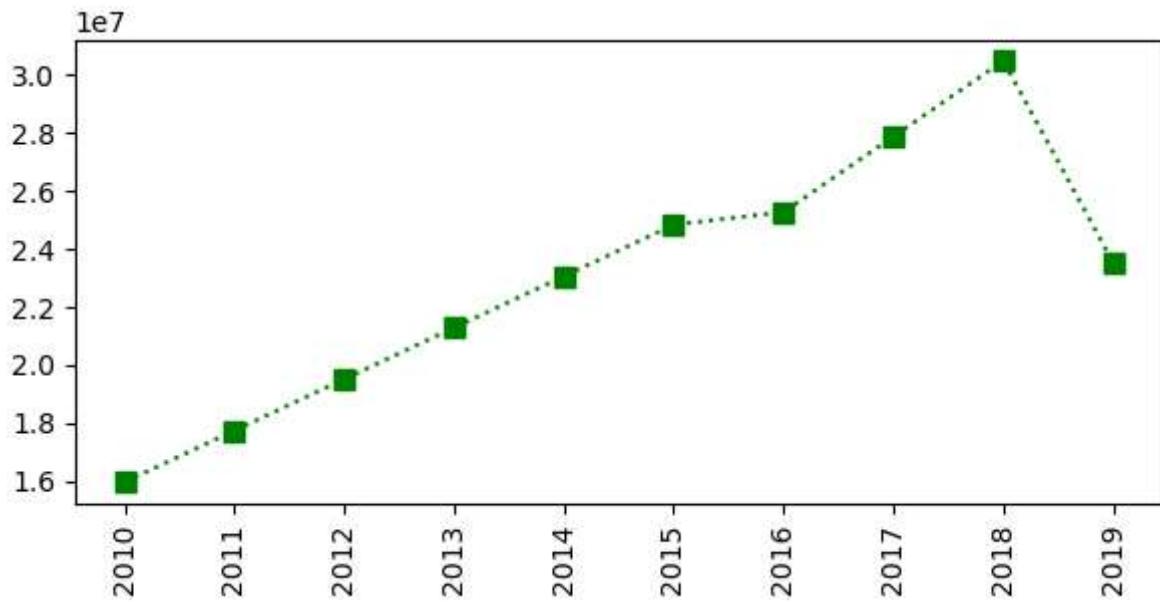
```
In [466...]: Pdict
```

```
Out[466...]: {'Sachin': 0,
 'Rahul': 1,
 'Smith': 2,
 'Sami': 3,
 'Pollard': 4,
 'Morris': 5,
 'Samson': 6,
 'Dhoni': 7,
 'Kohli': 8,
 'Sky': 9}
```

```
In [497...]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7)
plt.xticks(list(range(0,10)), Seasons)
plt.show()
```



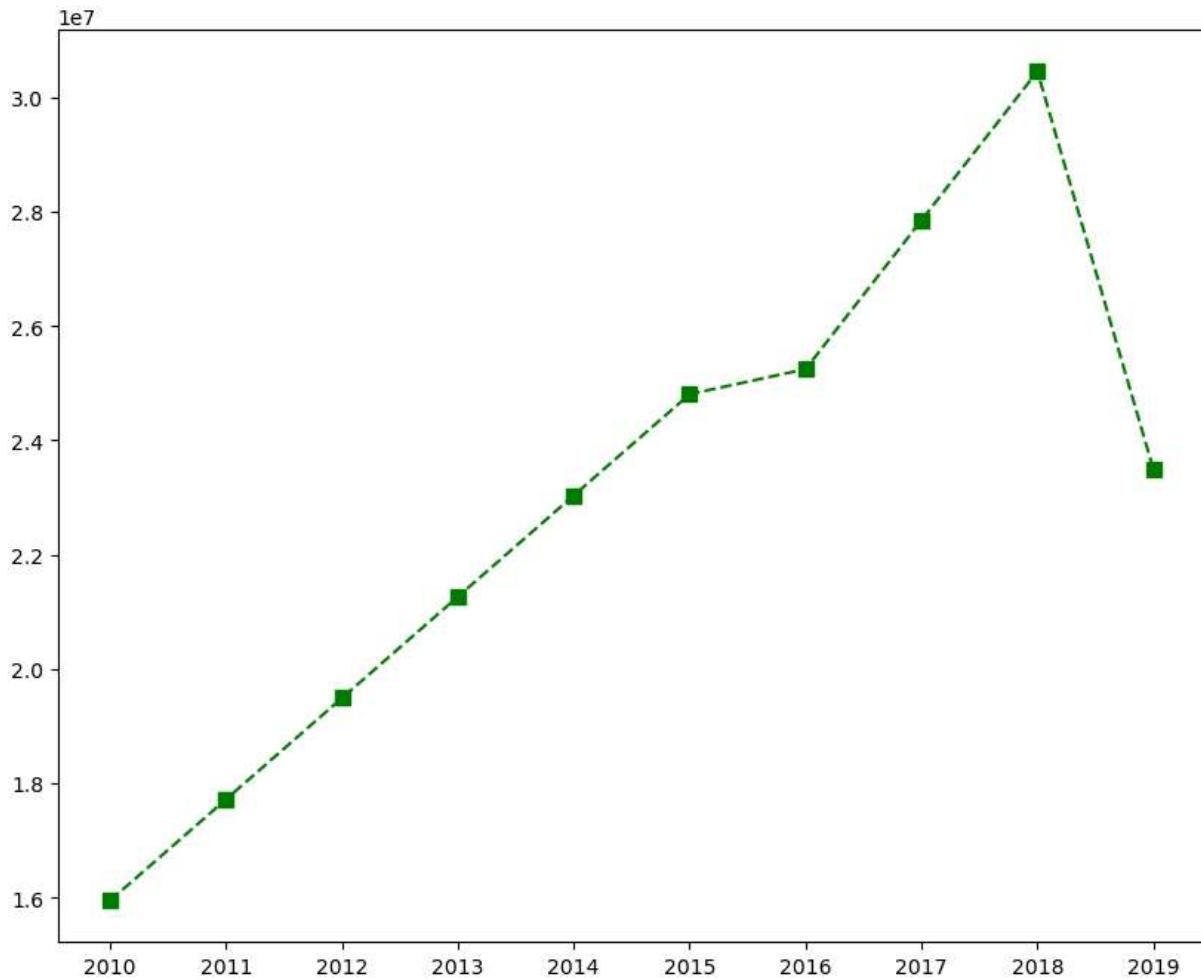
```
In [573]: plt.plot(Salary[0], c='Green', ls = ':', marker = 's', ms = 7, label = Players[0])
plt.xticks(list(range(0,10)), Seasons, rotation='vertical')
plt.show()
```



```
In [469]: Games
```

```
Out[469]: array([[80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
   [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
   [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
   [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
   [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
   [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
   [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
   [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
   [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
   [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]])
```

```
In [470]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.xticks(list(range(0,10)), Seasons, rotation='horizontal')
plt.show()
```



```
In [471]: Salary[0]
```

```
Out[471]: array([15946875, 17718750, 19490625, 21262500, 23034375, 24806250,
   25244493, 27849149, 30453805, 23500000])
```

```
In [ ]: Salary[1]
```

```
In [ ]: plt.plot(Salary[1], c='Blue', ls = ':', marker = 'o', ms = 10, label = Players[1])
```

```
In [ ]: # More visualization
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 10, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = ':', marker = 'o', ms = 10, label = Players[1])

plt.xticks(list(range(0,10)), Seasons, rotation='vertical')

plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '--', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='purple', ls = '--', marker = '^', ms = 8, label = Players[2])

plt.xticks(list(range(0,10)), Seasons, rotation='vertical')

plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '-.', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='purple', ls = '--', marker = '^', ms = 8, label = Players[2])
plt.plot(Salary[3], c='Red', ls = ':', marker = 'd', ms = 8, label = Players[3])

plt.xticks(list(range(0,10)), Seasons, rotation='vertical')

plt.show()
```

```
In [ ]: # how to add Legned in visualisation
```

```
plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = ':', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='purple', ls = '-.', marker = '^', ms = 8, label = Players[2])
plt.plot(Salary[3], c='Red', ls = '--', marker = 'd', ms = 8, label = Players[3])
plt.legend()
plt.xticks(list(range(0,10)), Seasons, rotation='vertical')

plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '--', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='purple', ls = '--', marker = '^', ms = 8, label = Players[2])
plt.plot(Salary[3], c='Red', ls = '--', marker = 'd', ms = 8, label = Players[3])
plt.legend(loc = 'upper left', bbox_to_anchor=(0,0) )
plt.xticks(list(range(0,10)), Seasons, rotation='vertical')

plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '--', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='Green', ls = '--', marker = '^', ms = 8, label = Players[2])
plt.plot(Salary[3], c='Red', ls = '--', marker = 'd', ms = 8, label = Players[3])
plt.legend(loc = 'upper right', bbox_to_anchor=(1,0) )
plt.xticks(list(range(0,10)), Seasons, rotation='vertical')
```

```
plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '--', marker = 'o', ms = 5, label = Players[1])
plt.plot(Salary[2], c='Green', ls = '--', marker = '^', ms = 8, label = Players[2])
plt.plot(Salary[3], c='Red', ls = '--', marker = 'd', ms = 8, label = Players[3])
plt.legend(loc = 'lower right',bbox_to_anchor=(0.5,1) )
plt.xticks(list(range(0,10)), Seasons,rotation='vertical')

plt.show()
```

```
In [ ]: plt.plot(Salary[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Salary[1], c='Blue', ls = '--', marker = 'o', ms = 7, label = Players[1])
plt.plot(Salary[2], c='Green', ls = '--', marker = '^', ms = 7, label = Players[2])
plt.plot(Salary[3], c='Purple', ls = '--', marker = 'D', ms = 7, label = Players[3])
plt.plot(Salary[4], c='Black', ls = '--', marker = 's', ms = 7, label = Players[4])
plt.plot(Salary[5], c='Red', ls = '--', marker = 'o', ms = 7, label = Players[5])
plt.plot(Salary[6], c='Red', ls = '--', marker = '^', ms = 7, label = Players[6])
plt.plot(Salary[7], c='Red', ls = '--', marker = 'd', ms = 7, label = Players[7])
plt.plot(Salary[8], c='Red', ls = '--', marker = 's', ms = 7, label = Players[8])
plt.plot(Salary[9], c='Red', ls = '--', marker = 'o', ms = 7, label = Players[9])

plt.legend(loc = 'lower right',bbox_to_anchor=(0.5,1) )
plt.xticks(list(range(0,10)), Seasons,rotation='vertical')

plt.show()
```

```
In [ ]: # we can visualize the how many games played by a player
```

```
plt.plot(Games[0], c='Green', ls = '--', marker = 's', ms = 7, label = Players[0])
plt.plot(Games[1], c='Blue', ls = '--', marker = 'o', ms = 7, label = Players[1])
plt.plot(Games[2], c='Green', ls = '--', marker = '^', ms = 7, label = Players[2])
plt.plot(Games[3], c='Red', ls = '--', marker = 'D', ms = 7, label = Players[3])
plt.plot(Games[4], c='Black', ls = '--', marker = 's', ms = 7, label = Players[4])
plt.plot(Games[5], c='Blue', ls = '--', marker = 'o', ms = 7, label = Players[5])
plt.plot(Games[6], c='red', ls = '--', marker = '^', ms = 7, label = Players[6])
plt.plot(Games[7], c='Green', ls = '--', marker = 'd', ms = 7, label = Players[7])
plt.plot(Games[8], c='Red', ls = '--', marker = 's', ms = 7, label = Players[8])
plt.plot(Games[9], c='Blue', ls = '--', marker = 'o', ms = 7, label = Players[9])

plt.legend(loc = 'lower right',bbox_to_anchor=(0.5,1) )
plt.xticks(list(range(0,10)), Seasons,rotation='vertical')

plt.show()
```

- In this section we learned -

1>Matrices 2>Building matrices - np.reshape 3>Dictionaried in python (order doesnot matter) (keys & values) 4>visualizaing using pyplot 5>Basket ball analysis

```
In [ ]:
```