In [5]:

```
#Dear Student,
#
#Welcome to the world of Basketball Data!
#I'm sure you will enjoy this section of the Python Programming course.
#
#Instructions for this dataset:
# Simply copy ALL the lines in this script by pressing
# CTRL+A on Windows or CMND+A on Mac and run the Jupyter cell
# Once you have executed the commands the following objects
# will be created:
# Matrices:
# - Salary
# - Games
# - MinutesPlayed
# - FieldGoals
# - FieldGoalAttempts
# - Points
# Lists:
# - Players
# - Seasons
# Dictionaries:
# - Sdict
# - Pdict
#We will understand these inside the course.
#
#Sincerely,
#Kirill Eremenko
#www.superdatascience.com

#Copyright: These datasets were prepared using publicly available data.
#           However, theses scripts are subject to Copyright Laws.
#           If you wish to use these Python scripts outside of the Python Programming Course
#           by Kirill Eremenko, you may do so by referencing www.superdatascience.com in your work.

#Comments:
#Seasons are labeled based on the first year in the season
#E.g. the 2012-2013 season is preseneted as simply 2012

#Notes and Corrections to the data:
#Kevin Durant: 2006 - College Data Used
#Kevin Durant: 2005 - Proxied With 2006 Data
#Derrick Rose: 2012 - Did Not Play
#Derrick Rose: 2007 - College Data Used
#Derrick Rose: 2006 - Proxied With 2007 Data
#Derrick Rose: 2005 - Proxied With 2007 Data

#Import numpy
import numpy as np

#Seasons
Seasons = ["2005","2006","2007","2008","2009","2010","2011","2012","2013","2014"]
Sdict = {"2005":0,"2006":1,"2007":2,"2008":3,"2009":4,"2010":5,"2011":6,"2012":7,"2013":8,"2014":9}

#Players
Players = ["KobeBryant","JoeJohnson","LeBronJames","CarmeloAnthony","DwightHoward","ChrisBosh","ChrisP
ç
Pdict = {"KobeBryant":0,"JoeJohnson":1,"LeBronJames":2,"CarmeloAnthony":3,"DwightHoward":4,"ChrisBosh"



#Salaries
KobeBryant_Salary = [15946875,17718750,19490625,21262500,23034375,24806250,25244493,27849149,30453805,
JoeJohnson_Salary = [12000000,12744189,13488377,14232567,14976754,16324500,18038573,19752645,21466718,
LeBronJames_Salary = [4621800,5828090,13041250,14410581,15779912,14500000,16022500,17545000,19067500,2
CarmeloAnthony_Salary = [3713640,4694041,13041250,14410581,15779912,17149243,18518574,19450000,2240747
DwightHoward_Salary = [4493160,4806720,6061274,13758000,15202590,16647180,18091770,19536360,20513178,2
ChrisBosh_Salary = [3348000,4235220,12455000,14410581,15779912,14500000,16022500,17545000,19067500,206
ChrisPaul_Salary = [3144240,3380160,3615960,4574189,13520500,14940153,16359805,17779458,18668431,20068
KevinDurant_Salary = [0,0,4171200,4484040,4796880,6053663,15506632,16669630,17832627,18995624]
DerrickRose_Salary = [0,0,0,4822800,5184480,5546160,6993708,16402500,17632688,18862875]
DwayneWade_Salary = [3031920,3841443,13041250,14410581,15779912,14200000,15691000,17182000,18673000,15
#Matrix
Salary = np.array([KobeBryant_Salary, JoeJohnson_Salary, LeBronJames_Salary, CarmeloAnthony_Salary, Dw
#insight:individual players salary list put into anaother list and apply the np.array function to byil
#Games
KobeBryant_G = [80,77,82,82,73,82,58,78,6,35]
```

```
JoeJohnson_G = [82,57,82,79,76,72,60,72,79,80]
LeBronJames_G = [79,78,75,81,76,79,62,76,77,69]
CarmeloAnthony_G = [80,65,77,66,69,77,55,67,77,40]
DwightHoward_G = [82,82,82,79,82,78,54,76,71,41]
ChrisBosh_G = [70,69,67,77,70,77,57,74,79,44]
ChrisPaul_G = [78,64,80,78,45,80,60,70,62,82]
KevinDurant_G = [35,35,80,74,82,78,66,81,81,27]
DerrickRose_G = [40,40,40,81,78,81,39,0,10,51]
DwayneWade_G = [75,51,51,79,77,76,49,69,54,62]
#Matrix
Games = np.array([KobeBryant_G, JoeJohnson_G, LeBronJames_G, CarmeloAnthony_G, DwightHoward_G, ChrisBo

#Minutes Played
KobeBryant_MP = [3277,3140,3192,2960,2835,2779,2232,3013,177,1207]
JoeJohnson_MP = [3340,2359,3343,3124,2886,2554,2127,2642,2575,2791]
LeBronJames_MP = [3361,3190,3027,3054,2966,3063,2326,2877,2902,2493]
CarmeloAnthony_MP = [2941,2486,2806,2277,2634,2751,1876,2482,2982,1428]
DwightHoward_MP = [3021,3023,3088,2821,2843,2935,2070,2722,2396,1223]
ChrisBosh_MP = [2751,2658,2425,2928,2526,2795,2007,2454,2531,1556]
ChrisPaul_MP = [2808,2353,3006,3002,1712,2880,2181,2335,2171,2857]
KevinDurant_MP = [1255,1255,2768,2885,3239,3038,2546,3119,3122,913]
DerrickRose_MP = [1168,1168,1168,3000,2871,3026,1375,0,311,1530]
DwayneWade_MP = [2892,1931,1954,3048,2792,2823,1625,2391,1775,1971]
#Matrix
MinutesPlayed = np.array([KobeBryant_MP, JoeJohnson_MP, LeBronJames_MP, CarmeloAnthony_MP, DwightHowar

#Field Goals
KobeBryant_FG = [978,813,775,800,716,740,574,738,31,266]
JoeJohnson_FG = [632,536,647,620,635,514,423,445,462,446]
LeBronJames_FG = [875,772,794,789,768,758,621,765,767,624]
CarmeloAnthony_FG = [756,691,728,535,688,684,441,669,743,358]
DwightHoward_FG = [468,526,583,560,510,619,416,470,473,251]
ChrisBosh_FG = [549,543,507,615,600,524,393,485,492,343]
ChrisPaul_FG = [407,381,630,631,314,430,425,412,406,568]
KevinDurant_FG = [306,306,587,661,794,711,643,731,849,238]
DerrickRose_FG = [208,208,208,574,672,711,302,0,58,338]
DwayneWade_FG = [699,472,439,854,719,692,416,569,415,509]
#Matrix
FieldGoals  = np.array([KobeBryant_FG, JoeJohnson_FG, LeBronJames_FG, CarmeloAnthony_FG, DwightHoward_

#Field Goal Attempts
KobeBryant_FGA = [2173,1757,1690,1712,1569,1639,1336,1595,73,713]
JoeJohnson_FGA = [1395,1139,1497,1420,1386,1161,931,1052,1018,1025]
LeBronJames_FGA = [1823,1621,1642,1613,1528,1485,1169,1354,1353,1279]
CarmeloAnthony_FGA = [1572,1453,1481,1207,1502,1503,1025,1489,1643,806]
DwightHoward_FGA = [881,873,974,979,834,1044,726,813,800,423]
ChrisBosh_FGA = [1087,1094,1027,1263,1158,1056,807,907,953,745]
ChrisPaul_FGA = [947,871,1291,1255,637,928,890,856,870,1170]
KevinDurant_FGA = [647,647,1366,1390,1668,1538,1297,1433,1688,467]
DerrickRose_FGA = [436,436,436,1208,1373,1597,695,0,164,835]
DwayneWade_FGA = [1413,962,937,1739,1511,1384,837,1093,761,1084]
#Matrix
FieldGoalAttempts = np.array([KobeBryant_FGA, JoeJohnson_FGA, LeBronJames_FGA, CarmeloAnthony_FGA, Dwi

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


```

Matrix :is a array presentation of table data one way to build a matrix in python is np.reshape(row,column),np.array(row,column)

In [6]:

```
Salary
```

Out[6]:

```
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]])
```

Insight:its a matrix, matrixis one of the way to work with table in python

In [5]:

```
Games
```

Out[5]:

```
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 [8]:

```
import numpy as np
mydata=np.arange(5,30)
mydata

```

Out[8]:

```
array([ 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])
```

In [11]:

```
#converting 1d array to multidimensional array
np.reshape(mydata, (5,5))
```

Out[11]:

```
array([[ 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]])
```

In [13]:

```
Mt=np.reshape(mydata, (5,5),order="c")#so defaukt is c(row order)
Mt
```

Out[13]:

```
array([[ 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]])
```

In [14]:

```
#get the no 23
Mt[3,3]
```

Out[14]:

```
23
```

In [16]:

```
Mt1=np.reshape(mydata, (5,5),order="F")# see the differnce
Mt1
```

Out[16]:

```
array([[ 5, 10, 15, 20, 25],
       [ 6, 11, 16, 21, 26],
       [ 7, 12, 17, 22, 27],
       [ 8, 13, 18, 23, 28],
       [ 9, 14, 19, 24, 29]])
```

In [17]:

```
#accces 23 again
Mt1[3,3]
```

Out[17]:

```
23
```

In [18]:

```
#OOP concept, as python is object oriented , we dont need to call module ,
#we just call function
mydata.reshape(5,5)
```

Out[18]:

```
array([[ 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]])
```

In [20]:

```
#create 3 list, make one list using it and convert it to array
a1=["pada","gafd","gfr"]
b=[4,5,6]
c=["si","pi","ni"]
```

In [22]:

```
[a1,b,c]
```

Out[22]:

```
[['pada', 'gafd', 'gfr'], [4, 5, 6], ['si', 'pi', 'ni']]
```

In [23]:

```
np.array([a1,b,c])#datatype unicode 21, and you see your intiger chage to string, because array contai
```

Out[23]:

```
array([['pada', 'gafd', 'gfr'],
       ['4', '5', '6'],
       ['si', 'pi', 'ni']], dtype='<U21')
```

Games

In [24]:

```
Games
```

Out[24]:

```
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 [25]:

```
Games[2]
```

Out[25]:

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

In [26]:

```
Games[5][-1]
```

Out[26]:

```
44
```

In [27]:

```
Games[5][9]
```

Out[27]:

```
44
```

In [28]:

```
Points[6]
```

Out[28]:

```
array([1258, 1104, 1684, 1781,  841, 1268, 1189, 1186, 1185, 1564])
```

In [30]:

```
Points[6][5]
```

Out[30]:

```
1268
```

#Dictionaries in python

In [36]:

```
Dict = {"key1": 1, "key2": 2, "key3": 3, "key4": 4, "key5": 5}

Dict #dictinories not ordered instesad it use key
```

Out[36]:

```
{'key1': 1, 'key2': 2, 'key3': 3, 'key4': 4, 'key5': 5}
```

In [37]:

```
Dict["key2"]
```

Out[37]:

```
2
```

In [38]:

```
Games
```

...

In [39]:

```
#Find the postion of player
Pdict["JoeJohnson"]
```

Out[39]:

```
1
```

In [41]:

```
Games[1]
```

Out[41]:

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

In [43]:

```
Games[1][4]
```

Out[43]:

```
76
```

In [46]:

```
Games[Pdict["JoeJohnson"],Sdict["2011"]]
```

Out[46]:

```
60
```

Matrix operation:

In [48]:

```
FieldGoals
```

Out[48]:

```
array([[978, 813, 775, 800, 716, 740, 574, 738,  31, 266],
       [632, 536, 647, 620, 635, 514, 423, 445, 462, 446],
       [875, 772, 794, 789, 768, 758, 621, 765, 767, 624],
       [756, 691, 728, 535, 688, 684, 441, 669, 743, 358],
       [468, 526, 583, 560, 510, 619, 416, 470, 473, 251],
       [549, 543, 507, 615, 600, 524, 393, 485, 492, 343],
       [407, 381, 630, 631, 314, 430, 425, 412, 406, 568],
       [306, 306, 587, 661, 794, 711, 643, 731, 849, 238],
       [208, 208, 208, 574, 672, 711, 302,   0,  58, 338],
       [699, 472, 439, 854, 719, 692, 416, 569, 415, 509]])
```

In [49]:

```
Games
```

Out[49]:

```
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 [51]:

```
#how many fieldsgoal per game each player score?
import warnings
warnings.filterwarnings("ignore")

FieldGoals/Games
```

Out[51]:

```
array([[12.225     , 10.55844156,  9.45121951,  9.75609756,  9.80821918,
         9.02439024,  9.89655172,  9.46153846,  5.16666667,  7.6       ],
       [ 7.70731707,  9.40350877,  7.8902439 ,  7.84810127,  8.35526316,
         7.13888889,  7.05      ,  6.18055556,  5.84810127,  5.575     ],
       [11.07594937,  9.8974359 , 10.58666667,  9.74074074, 10.10526316,
         9.59493671, 10.01612903, 10.06578947,  9.96103896,  9.04347826],
       [ 9.45      , 10.63076923,  9.45454545,  8.10606061,  9.97101449,
         8.88311688,  8.01818182,  9.98507463,  9.64935065,  8.95      ],
       [ 5.70731707,  6.41463415,  7.1097561 ,  7.08860759,  6.2195122 ,
         7.93589744,  7.7037037 ,  6.18421053,  6.66197183,  6.12195122],
       [ 7.84285714,  7.86956522,  7.56716418,  7.98701299,  8.57142857,
         6.80519481,  6.89473684,  6.55405405,  6.2278481 ,  7.79545455],
       [ 5.21794872,  5.953125  ,  7.875     ,  8.08974359,  6.97777778,
         5.375     ,  7.08333333,  5.88571429,  6.5483871 ,  6.92682927],
       [ 8.74285714,  8.74285714,  7.3375    ,  8.93243243,  9.68292683,
         9.11538462,  9.74242424,  9.02469136, 10.48148148,  8.81481481],
       [ 5.2       ,  5.2       ,  5.2       ,  7.08641975,  8.61538462,
         8.77777778,  7.74358974,         nan,  5.8       ,  6.62745098],
       [ 9.32      ,  9.25490196,  8.60784314, 10.81012658,  9.33766234,
         9.10526316,  8.48979592,  8.24637681,  7.68518519,  8.20967742]])
```

In [54]:

```
Goal_per_game=np.matrix.round(FieldGoals/Games)
Goal_per_game
```

Out[54]:

```
array([[12., 11.,  9., 10., 10.,  9., 10.,  9.,  5.,  8.],
       [ 8.,  9.,  8.,  8.,  8.,  7.,  7.,  6.,  6.,  6.],
       [11., 10., 11., 10., 10., 10., 10., 10., 10.,  9.],
       [ 9., 11.,  9.,  8., 10.,  9.,  8., 10., 10.,  9.],
       [ 6.,  6.,  7.,  7.,  6.,  8.,  8.,  6.,  7.,  6.],
       [ 8.,  8.,  8.,  8.,  9.,  7.,  7.,  7.,  6.,  8.],
       [ 5.,  6.,  8.,  8.,  7.,  5.,  7.,  6.,  7.,  7.],
       [ 9.,  9.,  7.,  9., 10.,  9., 10.,  9., 10.,  9.],
       [ 5.,  5.,  5.,  7.,  9.,  9.,  8., nan,  6.,  7.],
       [ 9.,  9.,  9., 11.,  9.,  9.,  8.,  8.,  8.,  8.]])
```

In [55]:

```
Goal_per_game[Pdict["JoeJohnson"],Sdict["2011"]]
```

Out[55]:

```
7.0
```

In [56]:

```
#how many played per game each player
minutes_per_game=np.matrix.round(MinutesPlayed/Games)
minutes_per_game
```

Out[56]:

```
array([[41., 41., 39., 36., 39., 34., 38., 39., 30., 34.],
       [41., 41., 41., 40., 38., 35., 35., 37., 33., 35.],
       [43., 41., 40., 38., 39., 39., 38., 38., 38., 36.],
       [37., 38., 36., 34., 38., 36., 34., 37., 39., 36.],
       [37., 37., 38., 36., 35., 38., 38., 36., 34., 30.],
       [39., 39., 36., 38., 36., 36., 35., 33., 32., 35.],
       [36., 37., 38., 38., 38., 36., 36., 33., 35., 35.],
       [36., 36., 35., 39., 40., 39., 39., 39., 39., 34.],
       [29., 29., 29., 37., 37., 37., 35., nan, 31., 30.],
       [39., 38., 38., 39., 36., 37., 33., 35., 33., 32.]])
```

In [57]:

```
minutes_per_game[Pdict["JoeJohnson"],Sdict["2011"]]
```

Out[57]:

```
35.0
```

In [59]:

```
Percent_of_succes=np.matrix.round(FieldGoals/FieldGoalAttempts,2)*100
Percent_of_succes
```

Out[59]:

```
array([[45., 46., 46., 47., 46., 45., 43., 46., 42., 37.],
       [45., 47., 43., 44., 46., 44., 45., 42., 45., 44.],
       [48., 48., 48., 49., 50., 51., 53., 56., 57., 49.],
       [48., 48., 49., 44., 46., 46., 43., 45., 45., 44.],
       [53., 60., 60., 57., 61., 59., 57., 58., 59., 59.],
       [51., 50., 49., 49., 52., 50., 49., 53., 52., 46.],
       [43., 44., 49., 50., 49., 46., 48., 48., 47., 49.],
       [47., 47., 43., 48., 48., 46., 50., 51., 50., 51.],
       [48., 48., 48., 48., 49., 45., 43., nan, 35., 40.],
       [49., 49., 47., 49., 48., 50., 50., 52., 55., 47.]])
```

Visualization

In [4]:

```
!pip install matplotlib
import matplotlib.pyplot as plt
%matplotlib inline

```

```
Requirement already satisfied: matplotlib in ./anaconda3/lib/python3.10/site-packages (3.7.0)
Requirement already satisfied: packaging>=20.0 in ./anaconda3/lib/python3.10/site-packages (f
rom matplotlib) (22.0)
Requirement already satisfied: numpy>=1.20 in ./anaconda3/lib/python3.10/site-packages (from
matplotlib) (1.23.5)
Requirement already satisfied: python-dateutil>=2.7 in ./anaconda3/lib/python3.10/site-packag
es (from matplotlib) (2.8.2)
Requirement already satisfied: fonttools>=4.22.0 in ./anaconda3/lib/python3.10/site-packages
(from matplotlib) (4.25.0)
Requirement already satisfied: pillow>=6.2.0 in ./anaconda3/lib/python3.10/site-packages (fro
m matplotlib) (9.4.0)
Requirement already satisfied: cycler>=0.10 in ./anaconda3/lib/python3.10/site-packages (from
matplotlib) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in ./anaconda3/lib/python3.10/site-packages
(from matplotlib) (1.4.4)
Requirement already satisfied: pyparsing>=2.3.1 in ./anaconda3/lib/python3.10/site-packages
(from matplotlib) (3.0.9)
Requirement already satisfied: contourpy>=1.0.1 in ./anaconda3/lib/python3.10/site-packages
(from matplotlib) (1.0.5)
Requirement already satisfied: six>=1.5 in ./anaconda3/lib/python3.10/site-packages (from pyt
hon-dateutil>=2.7->matplotlib) (1.16.0)
```

In [10]:

```
!pip --version
```

```
pip 22.3.1 from /Users/myyntiimac/anaconda3/lib/python3.10/site-packages/pip (python 3.10)
```

In [11]:

```
!pip show matplotlib
```

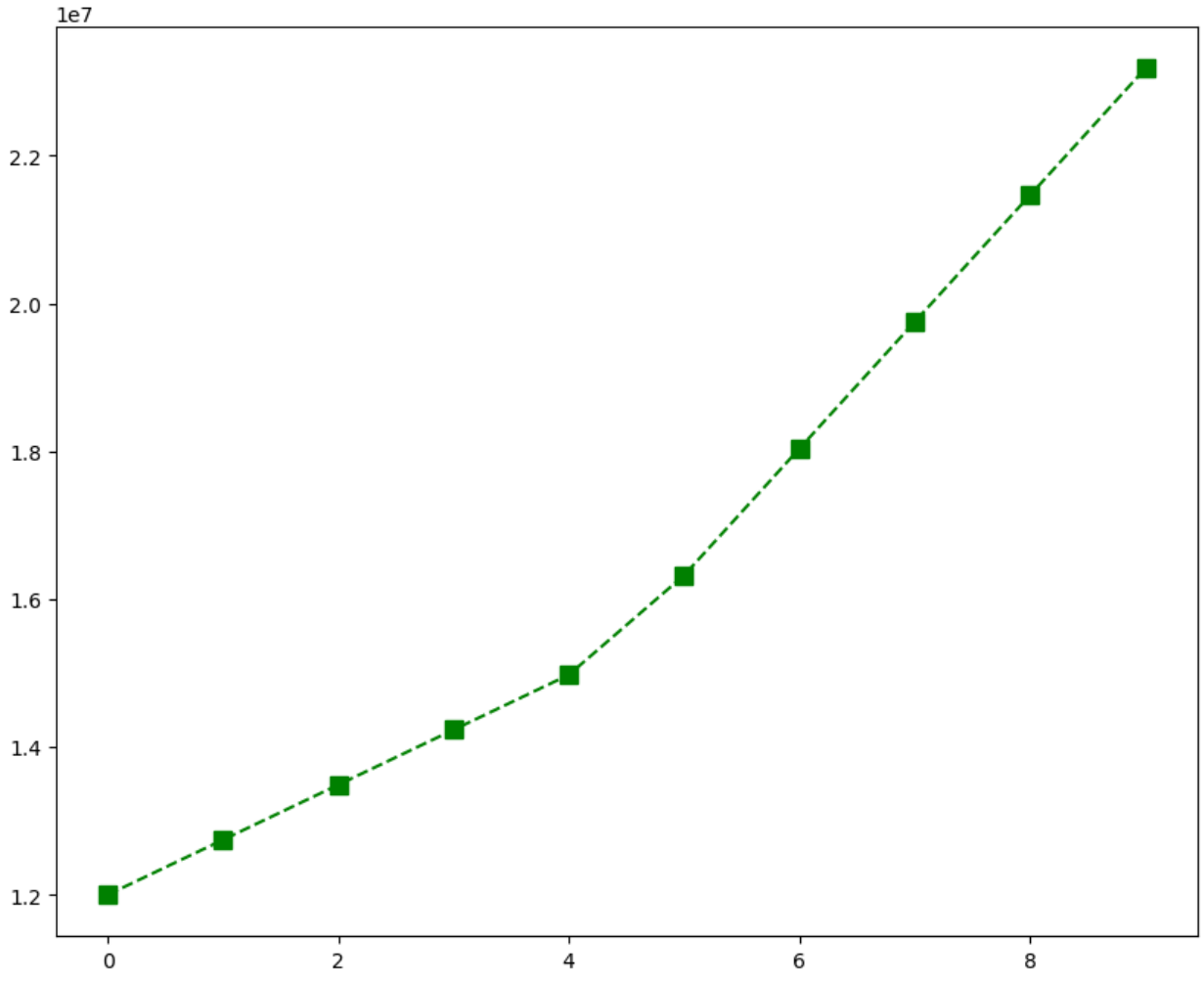
```
Name: matplotlib
Version: 3.7.0
Summary: Python plotting package
Home-page: https://matplotlib.org (https://matplotlib.org)
Author: John D. Hunter, Michael Droettboom
Author-email: matplotlib-users@python.org
License: PSF
Location: /Users/myyntiimac/anaconda3/lib/python3.10/site-packages
Requires: contourpy, cycler, fonttools, kiwisolver, numpy, packaging, pillow, pyparsing, pyth
on-dateutil
Required-by: missingno, pywaffle, seaborn
```

In [1]:

```
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["figure.figsize"] = (10, 8)
```

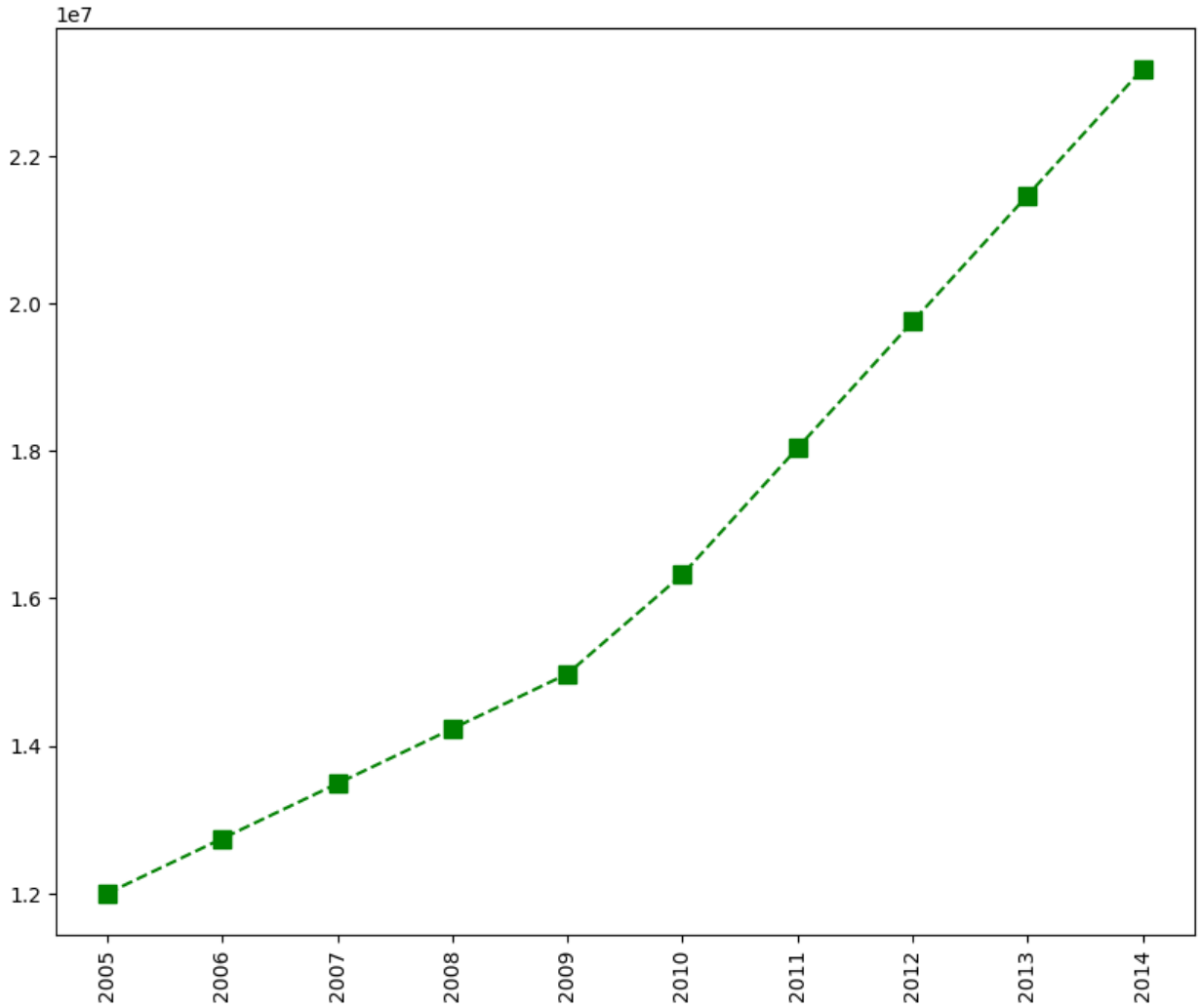
In [72]:

```
plt.plot(Salary[1],c="green",ls="--",marker="s",ms=8)
plt.show()
```



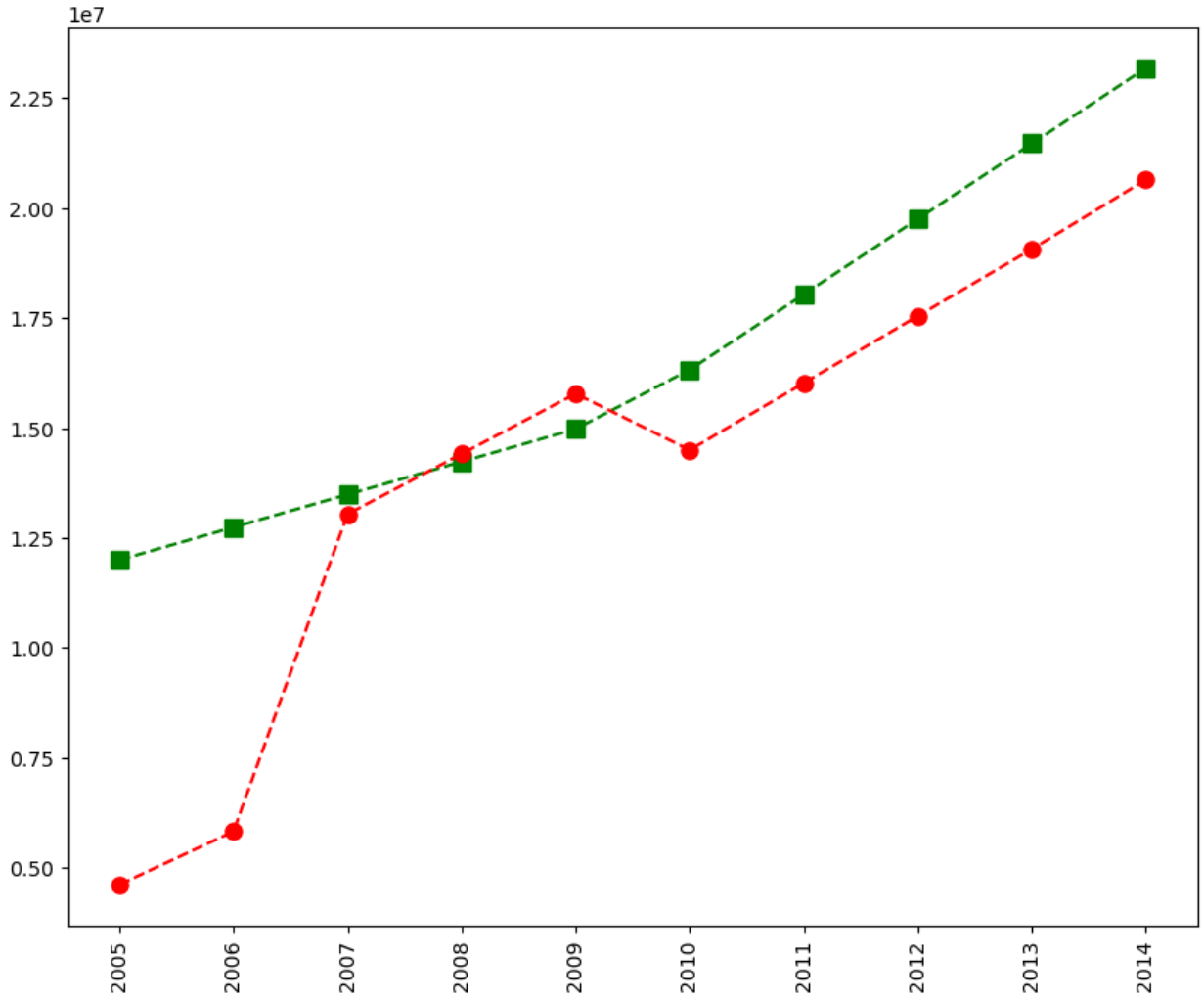
In [74]:

```
#inserting X-ticks as seasons
plt.plot(Salary[1],c="green",ls="--",marker="s",ms=8)
plt.xticks(list(range(0,10)),Seasons,rotation="vertical")
plt.show()
```



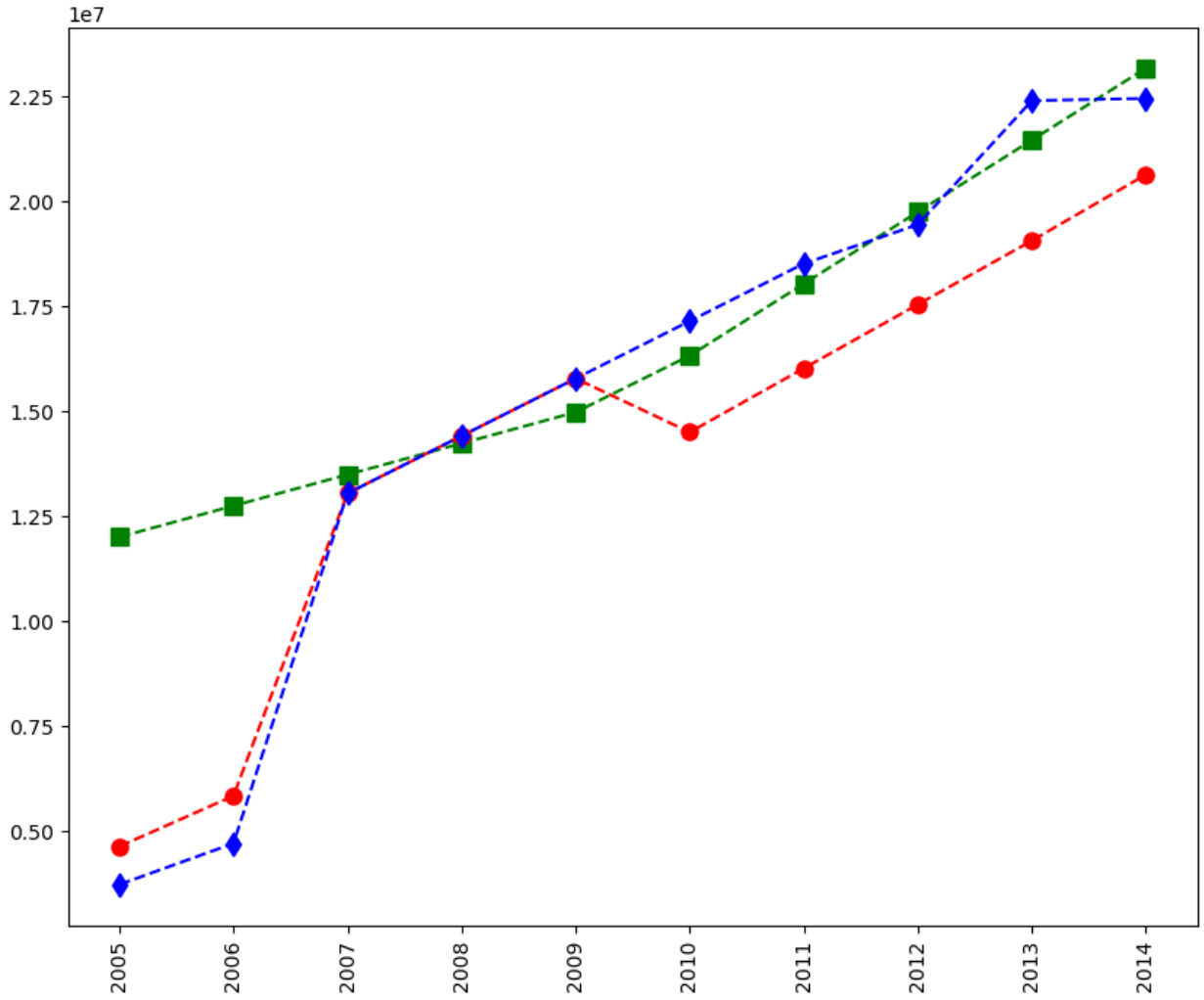
In [76]:

```
plt.plot(Salary[1],c="green",ls="--",marker="s",ms=8)
plt.plot(Salary[2],c="red",ls="--",marker="o",ms=8)
plt.xticks(list(range(0,10)),Seasons,rotation="vertical")
plt.show()
```



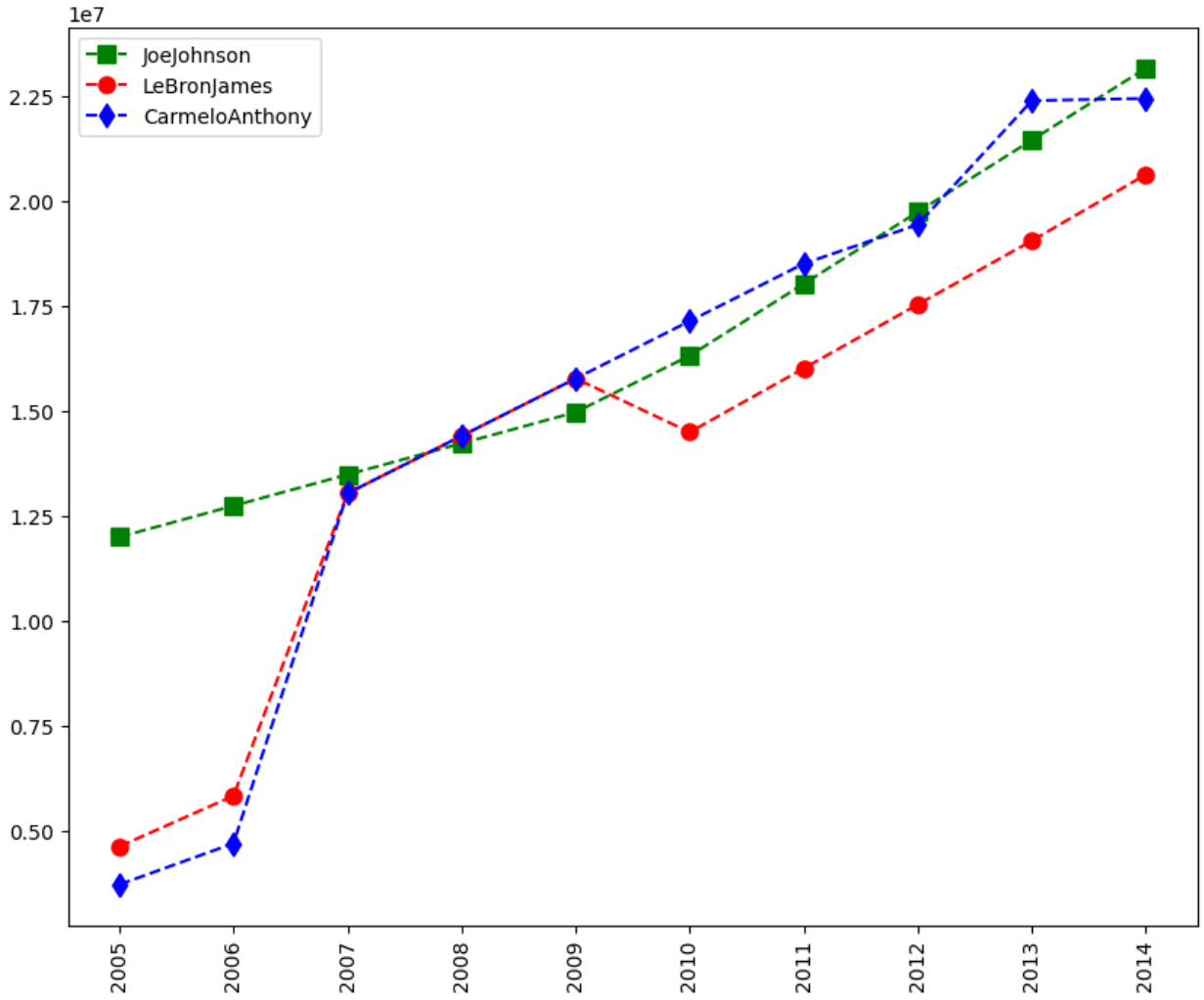
In [78]:

```
plt.plot(Salary[1],c="green",ls="--",marker="s",ms=8)
plt.plot(Salary[2],c="red",ls="--",marker="o",ms=8)
plt.plot(Salary[3],c="blue",ls="--",marker="d",ms=8)
plt.xticks(list(range(0,10)),Seasons,rotation="vertical")
plt.show()
```



In [7]:

```
plt.plot(Salary[1],c="green",ls="--",marker="s",ms=8,label=Players[1])
plt.plot(Salary[2],c="red",ls="--",marker="o",ms=8,label=Players[2])
plt.plot(Salary[3],c="blue",ls="--",marker="d",ms=8,label=Players[3])
plt.xticks(list(range(0,10)),Seasons,rotation="vertical",label='Line Plot')
plt.legend(loc="upper left")
plt.show()
```



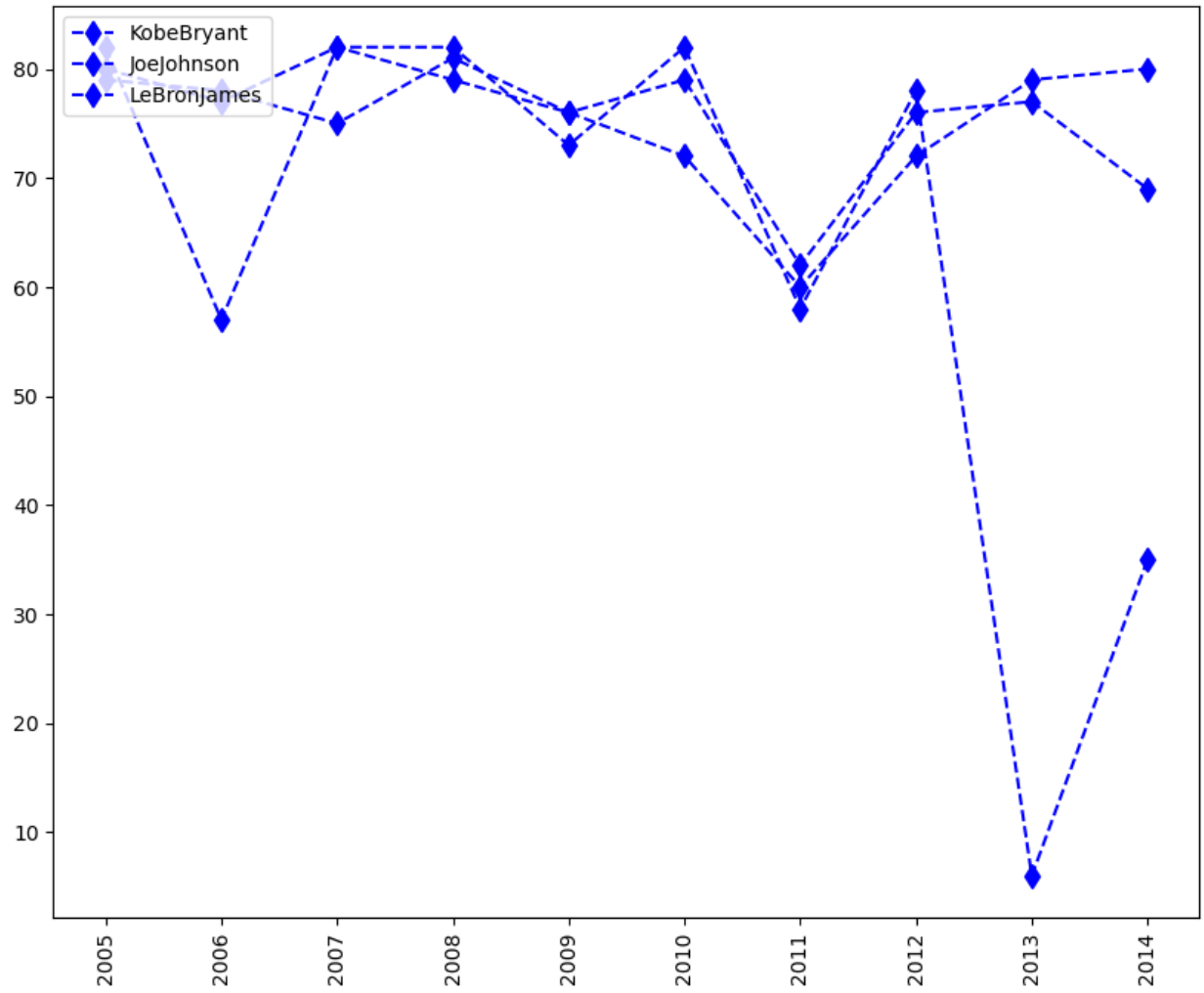
In [14]:

```
# how can we simplyfy the avobe code?
# creating our own function
def myplot(playerlist):
    for name in playerlist:
        plt.plot(Games[Pdict[name]],c="blue",ls="--",marker="d",ms=8,label=name)
    plt.xticks(list(range(0,10)),Seasons,rotation="vertical",label='Line Plot')
    plt.legend(loc="upper left")
    plt.show()

```

In [15]:

```
myplot(["KobeBryant","JoeJohnson","LeBronJames"])
```



#only one drawback about markker color and shape is same , otherwise this for loop , saves time , Lets fix this issure

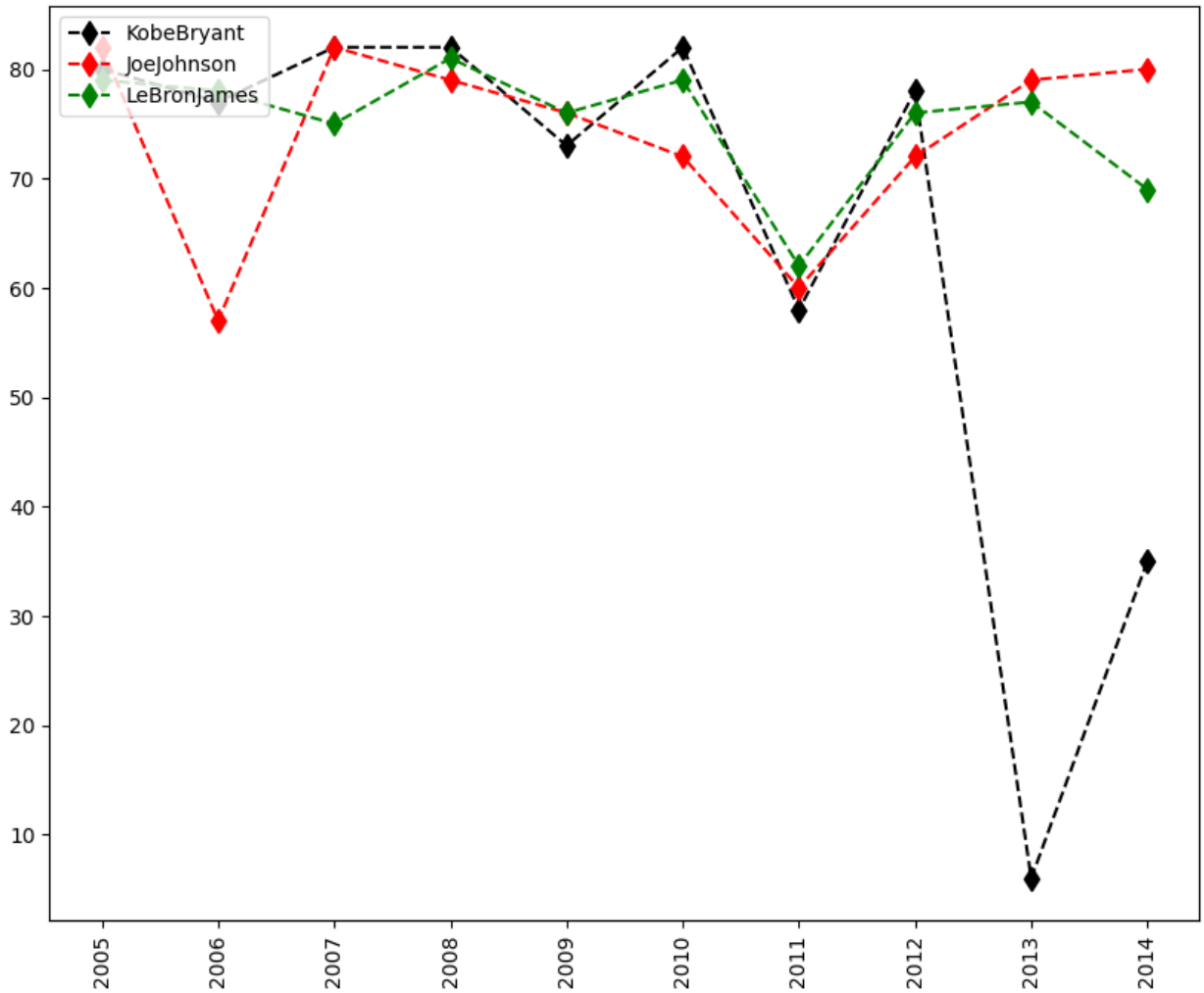
In [21]:

```
def myplot(playerlist):
    col = {"KobeBryant":"Black","JoeJohnson":"Red","LeBronJames":"Green","CarmeloAnthony":"Blue","Dwigh
    for name in playerlist:
        plt.plot(Games[Pdict[name]],c=col[name],ls="--",marker="d",ms=8,label=name)
    plt.xticks(list(range(0,10)),Seasons,rotation="vertical",label='Line Plot')
    plt.legend(loc="upper left")
    plt.show()

```

In [22]:

```
1 myplot(["KobeBryant","JoeJohnson","LeBronJames"])
```

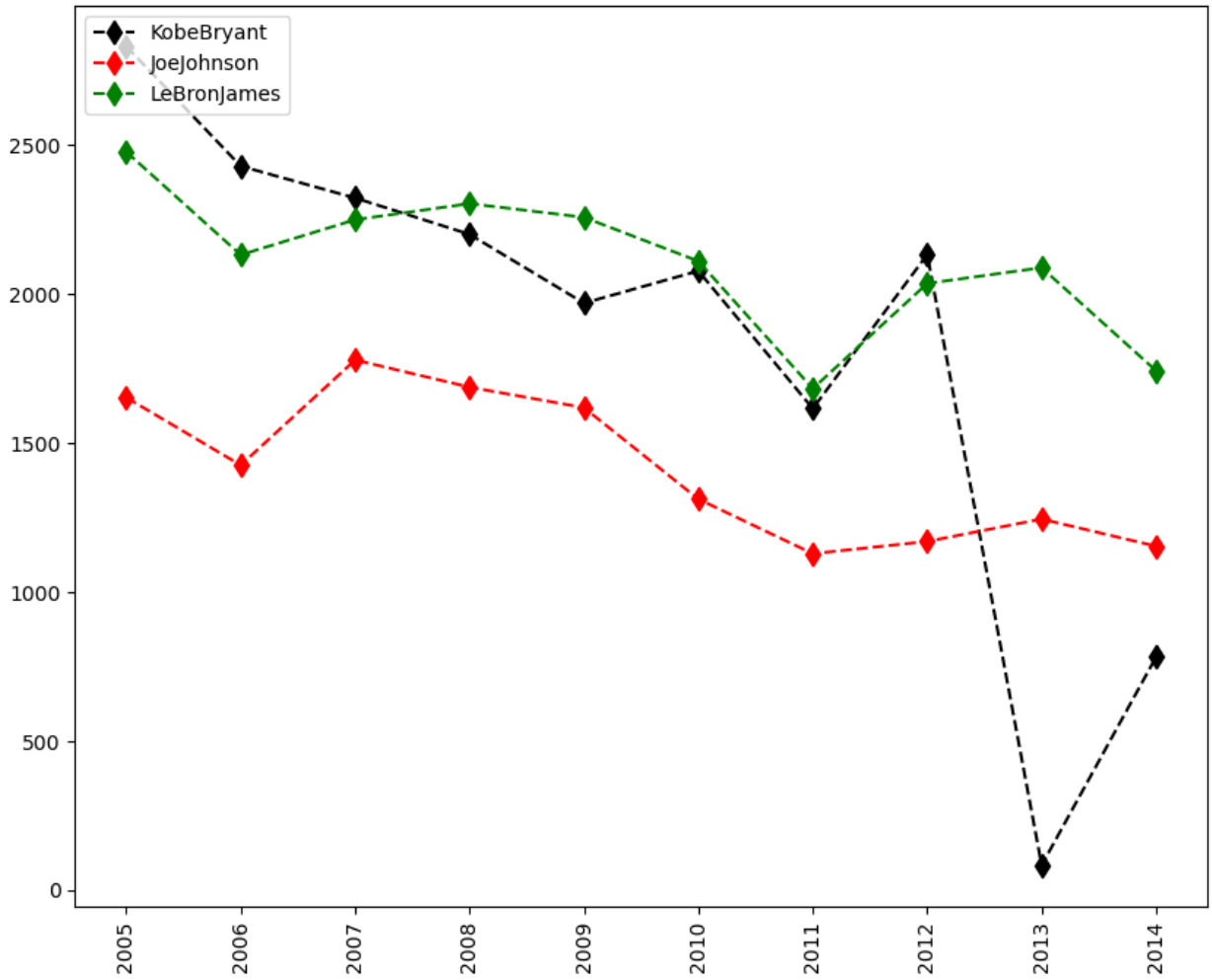


In [23]:

```
1 #univerasal function for any data not only game
2 def myplot(data,playerlist):
3     col = {"KobeBryant":"Black","JoeJohnson":"Red","LeBronJames":"Green","CarmeloAnthony":"Blue","Dwigh
4     for name in playerlist:
5         plt.plot(data[Pdict[name]],c=col[name],ls="--",marker="d",ms=8,label=name)
6     plt.xticks(list(range(0,10)),Seasons,rotation="vertical",label='Line Plot')
7     plt.legend(loc="upper left")
8     plt.show()
9
```

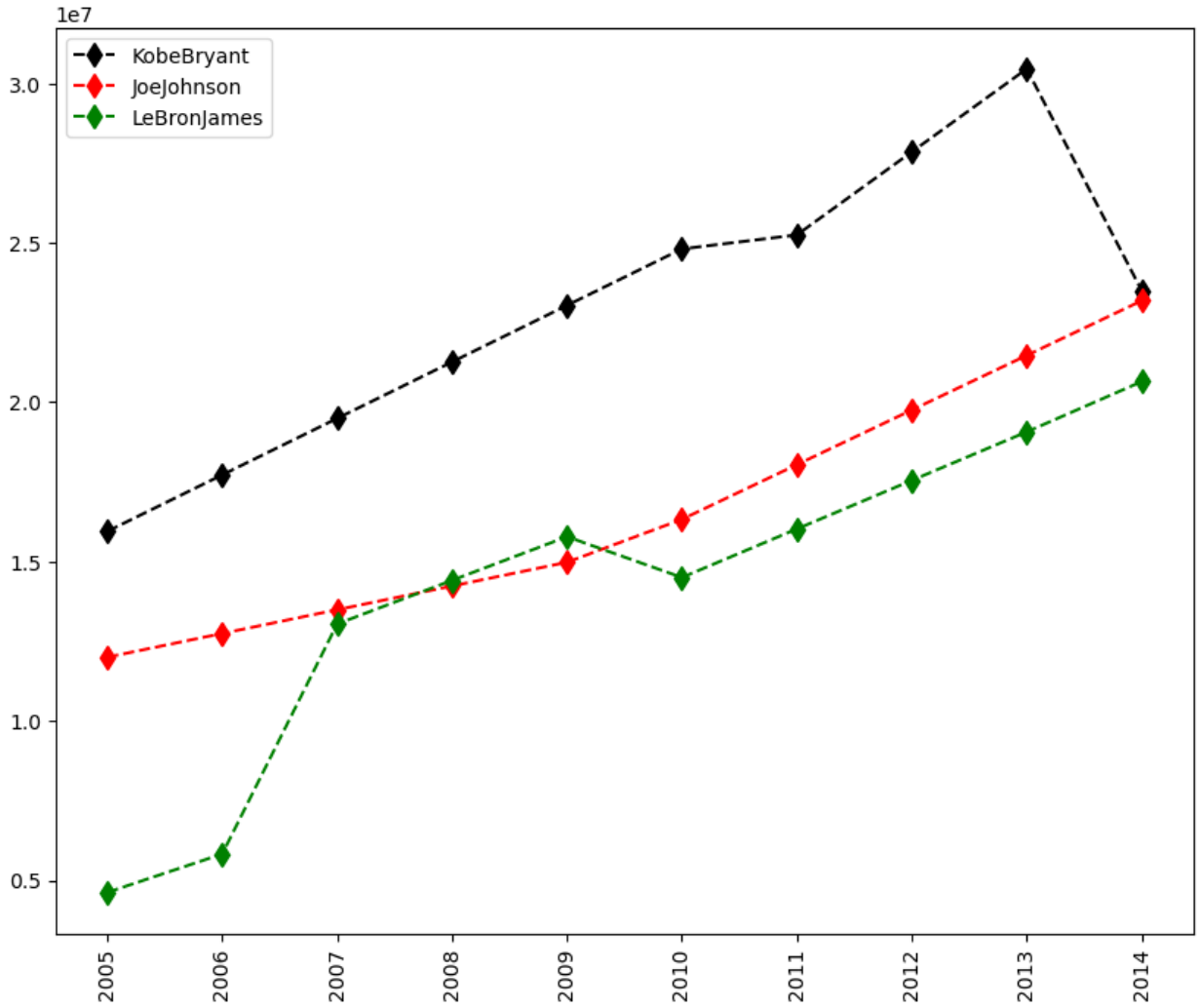
In [24]:

```
1 myplot(Points,["KobeBryant","JoeJohnson","LeBronJames"])
```



In [25]:

```
1 myplot(Salary,["KobeBryant","JoeJohnson","LeBronJames"])
```

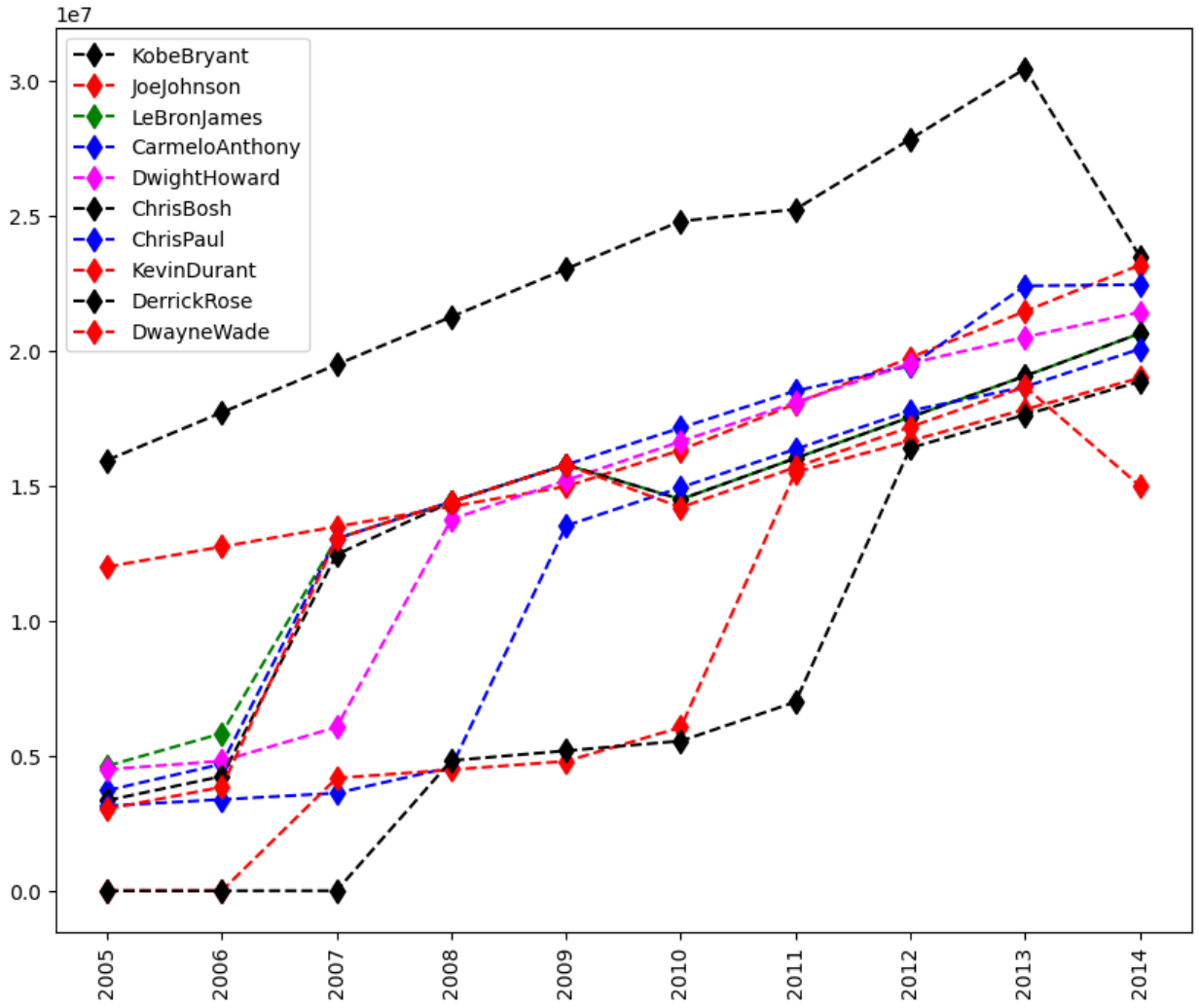


In [30]:

```
1
2
3 tHoward":"Magenta","ChrisBosh":"Black","ChrisPaul":"Blue","KevinDurant":"Red","DerrickRose":"Black","Dwayne
4
5
6
7
8
9
```

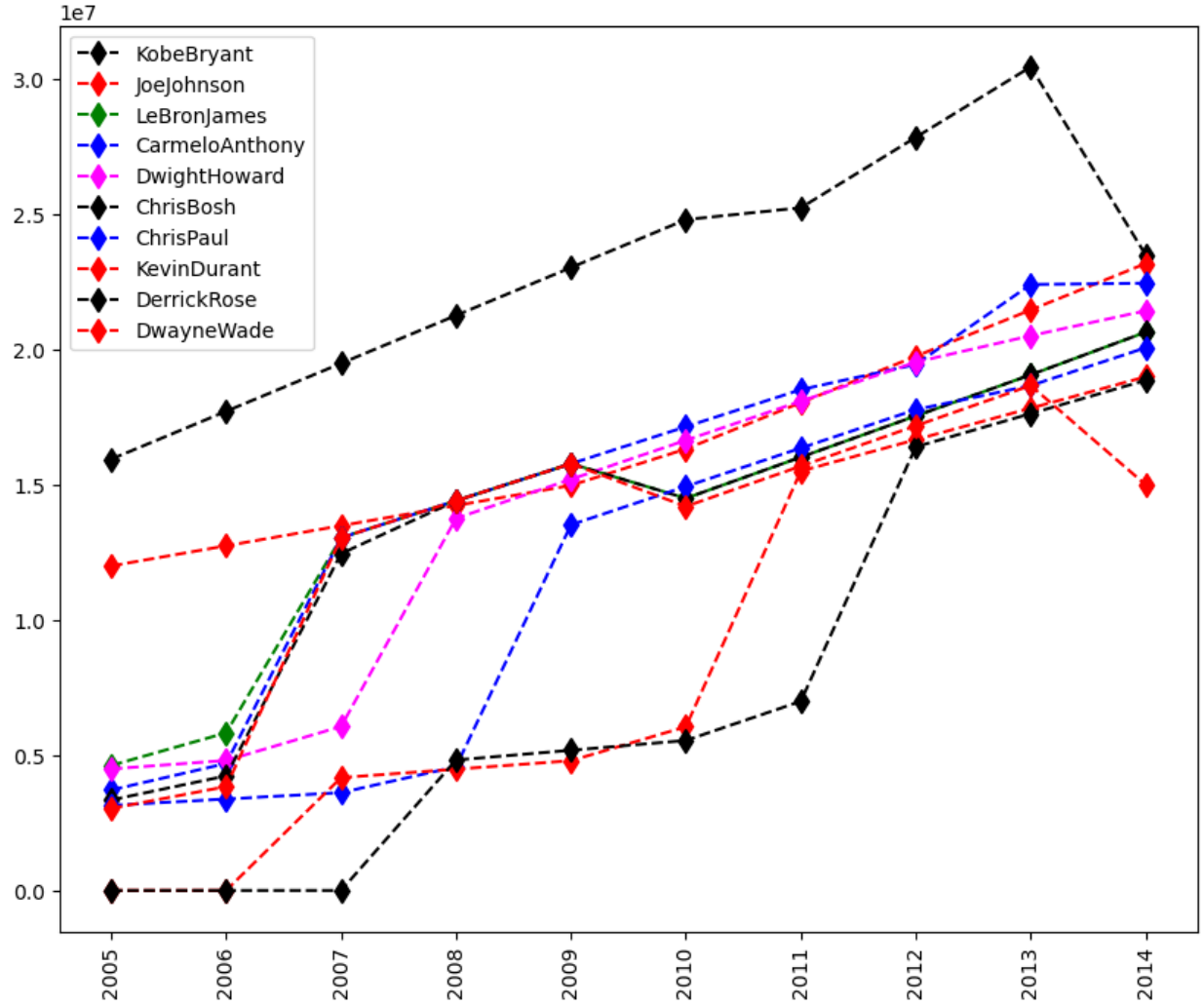
In [31]:

```
1 myplot(Salary)
```

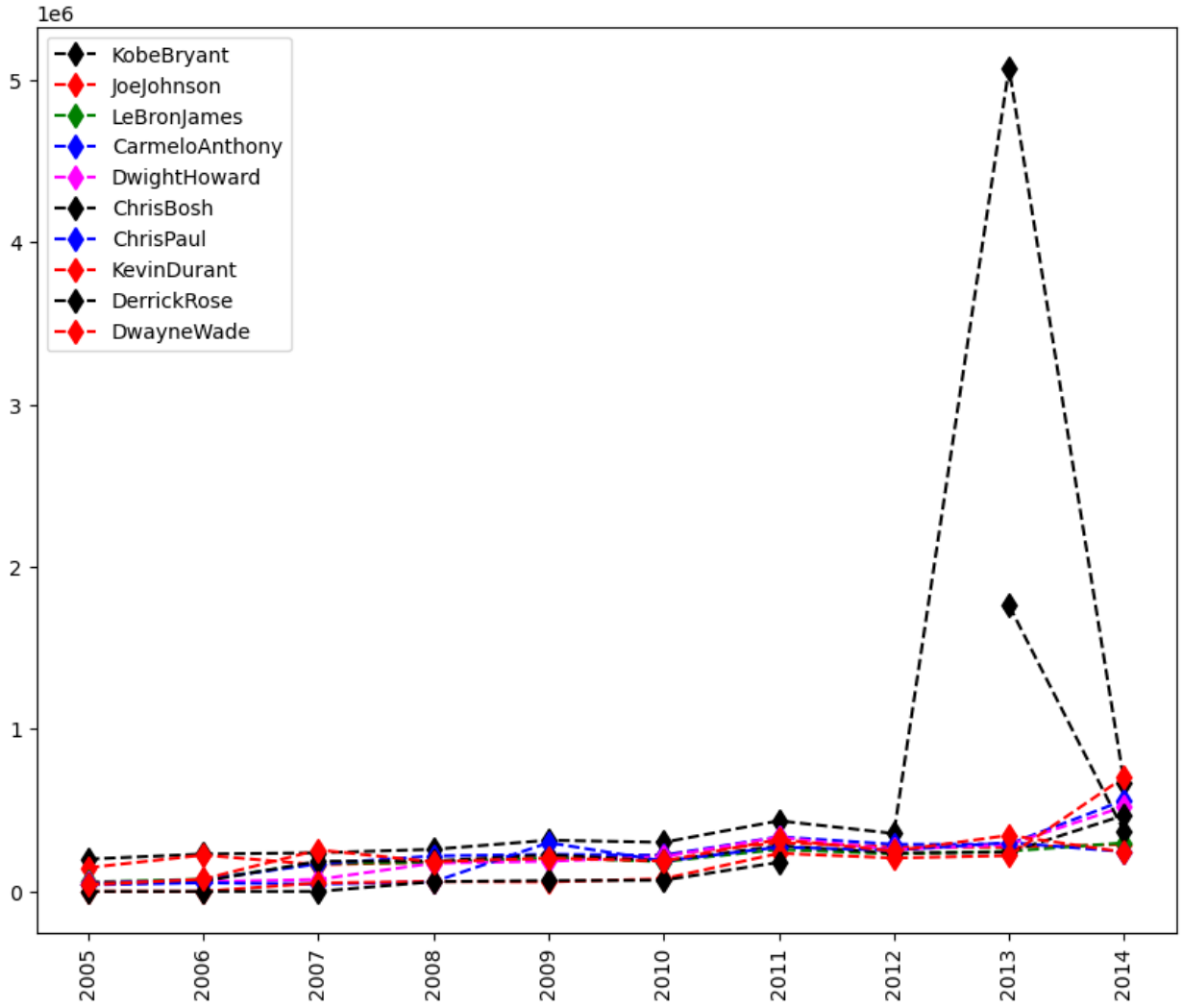


In [33]:

```
1 myplot(Salary)
2 myplot(Salary/Games)
```

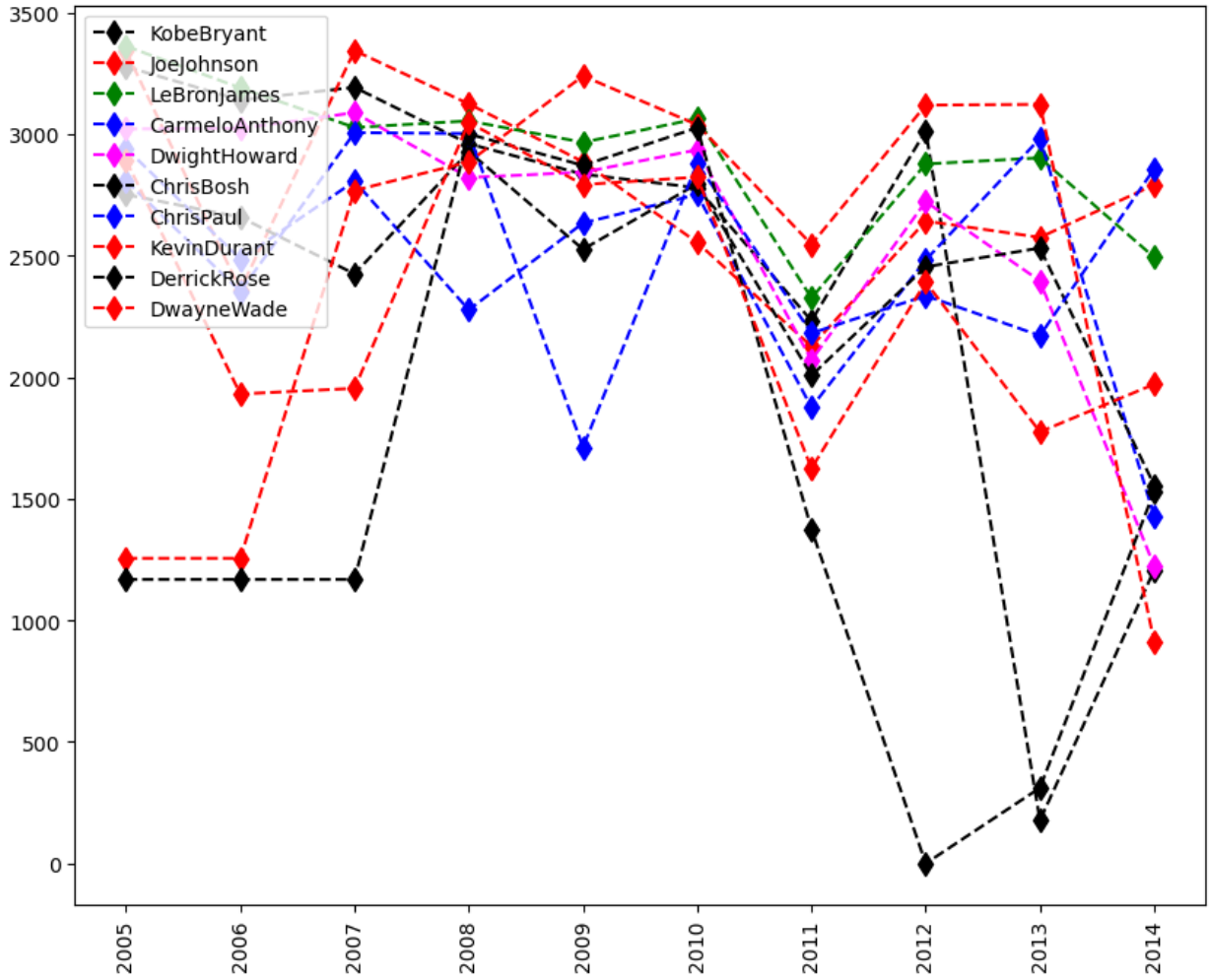


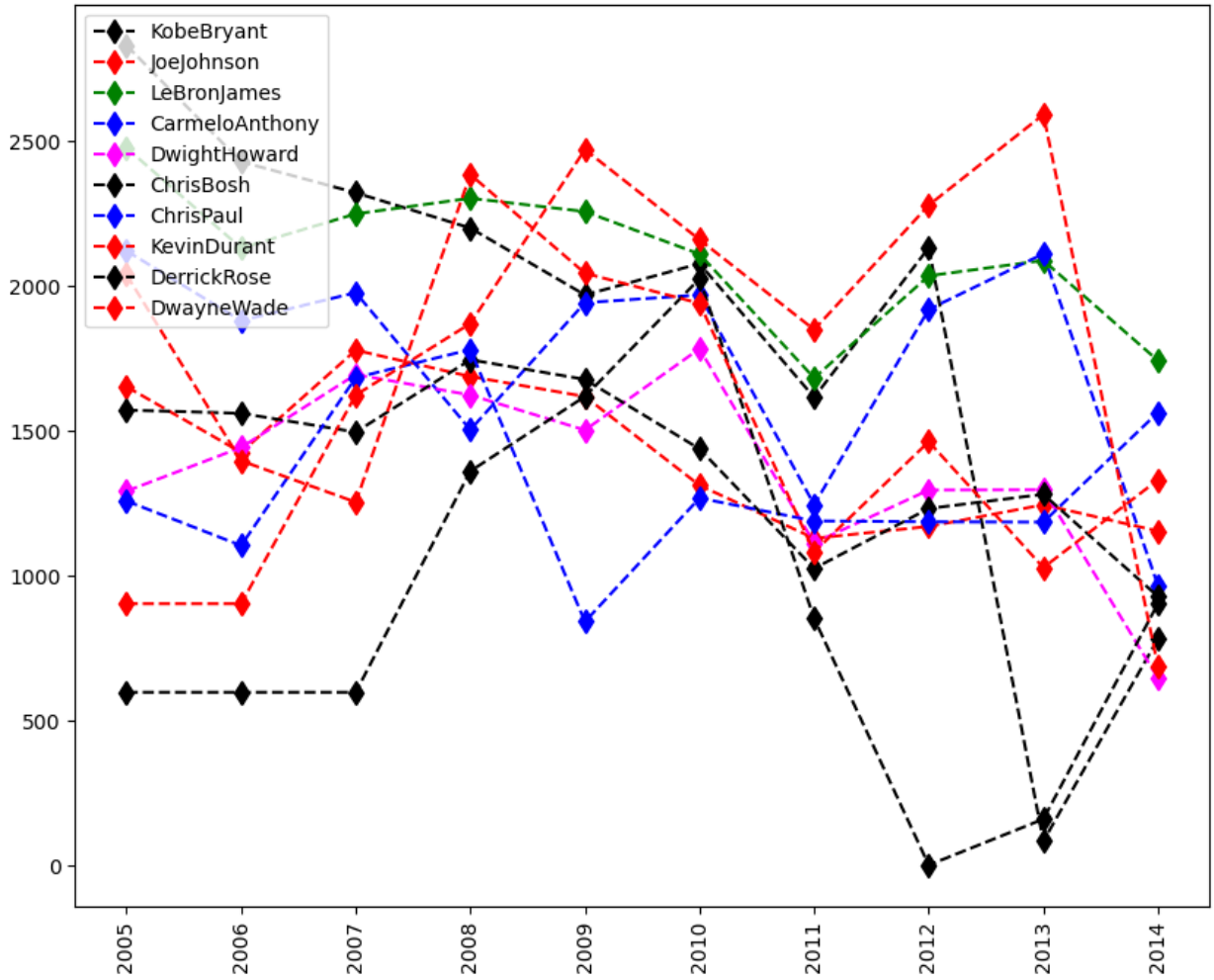

```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/3355333756.py:2: RuntimeWarn
ing: divide by zero encountered in divide
  myplot(Salary/Games)
```


1e6
KobeBryant
JoeJohnson
LeBronJames
CarmeloAnthony
DwightHoward
ChrisBosh
ChrisPaul
KevinDurant
DerrickRose
DwayneWade
5
4
3
2
1
0
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014

In [37]:

```
1 myplot(MinutesPlayed)
2 myplot(Points)
```

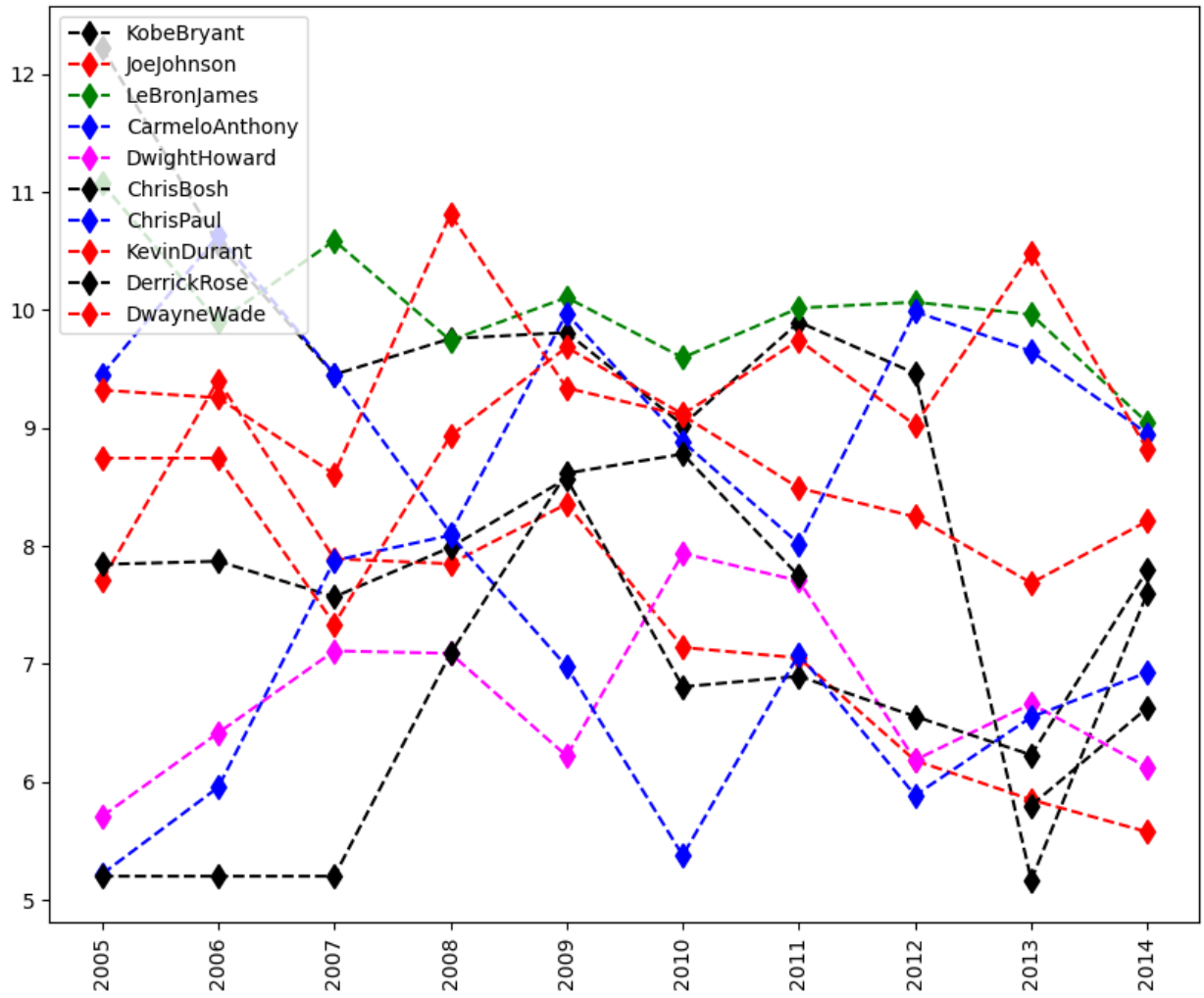



KobeBryant
JoeJohnson
LeBronJames
CarmeloAnthony
DwightHoward
ChrisBosh
ChrisPaul
KevinDurant
DerrickRose
DwayneWade
2500
2000
1500
1000
500
0
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014

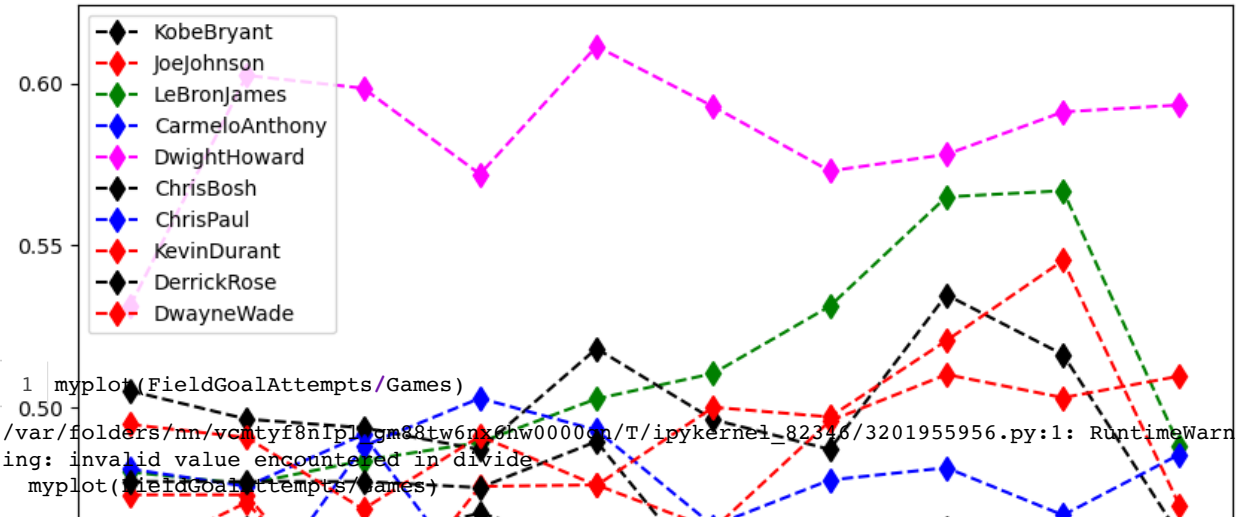
In [41]:

```
1 myplot(FieldGoals /Games)
2 myplot(FieldGoals /FieldGoalAttempts)
```

```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/1166618461.py:1: RuntimeWarn
ing: invalid value encountered in divide
  myplot(FieldGoals /Games)
```

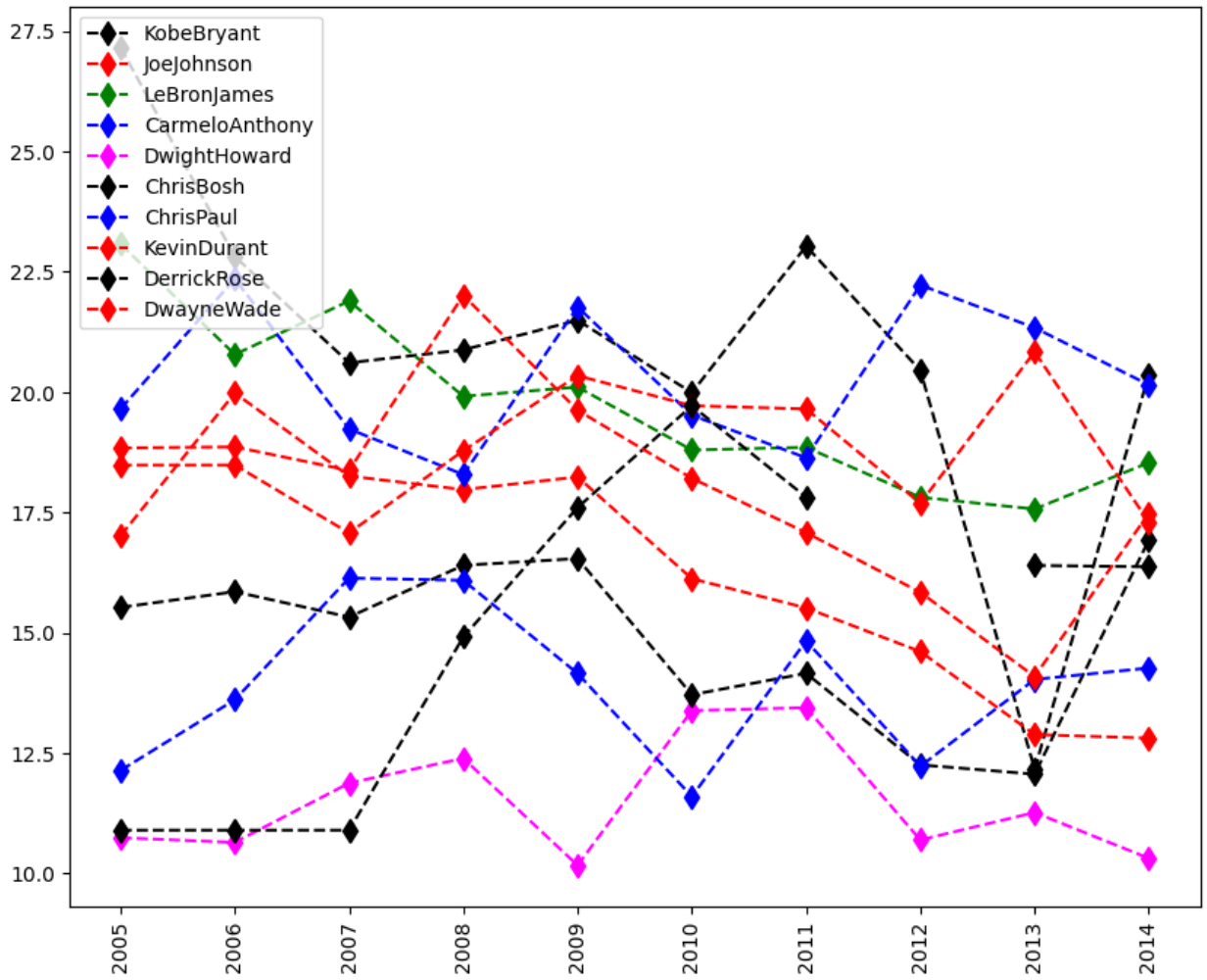



```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/1166618461.py:2: RuntimeWarn
ing: invalid value encountered in divide
  myplot(FieldGoals /FieldGoalAttempts)
```




```
1 myplot(FieldGoalAttempts/Games)
```

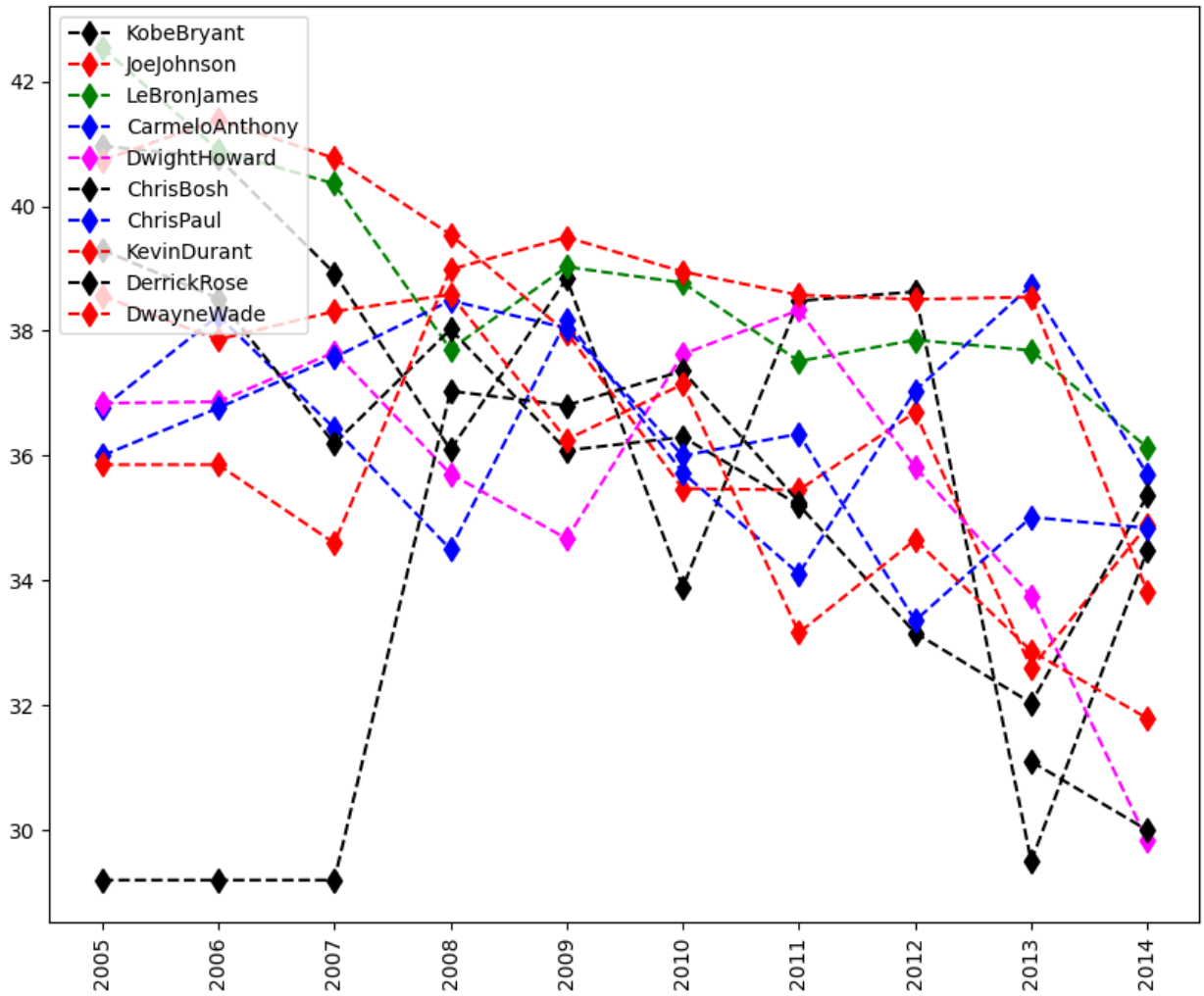
```
/var/folders/nn/vcntyf8n1pl1gm88tw6nx6hw0000gn/T/ipykernel_82346/3201955956.py:1: RuntimeWarn
ing: invalid value encountered in divide
  myplot(FieldGoalAttempts/Games)
```



In [43]:

```
1 myplot(MinutesPlayed/Games)
```

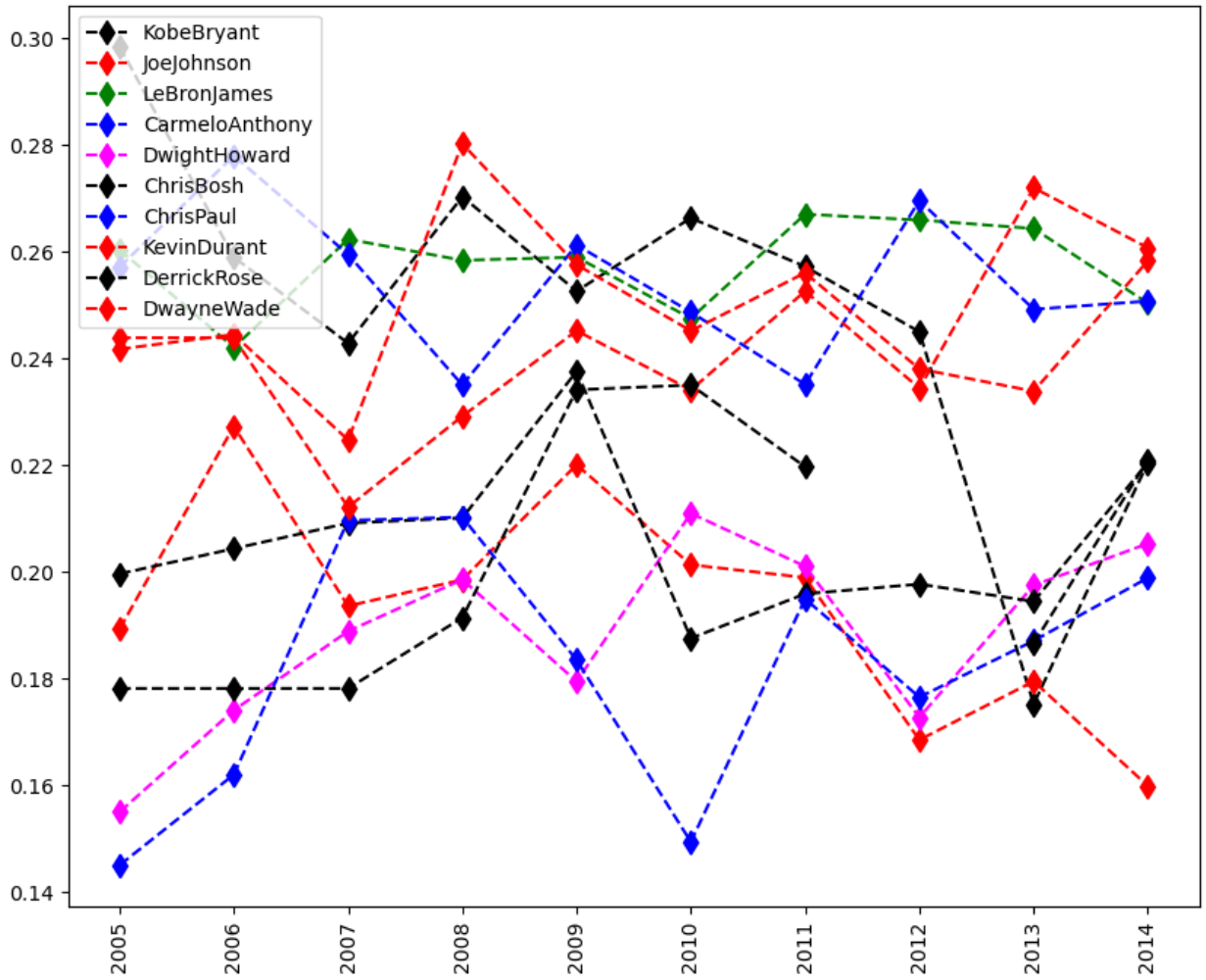
```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/3137096660.py:1: RuntimeWarn
ing: invalid value encountered in divide
  myplot(MinutesPlayed/Games)
```



In [44]:

```
1 myplot(FieldGoals/MinutesPlayed)
```

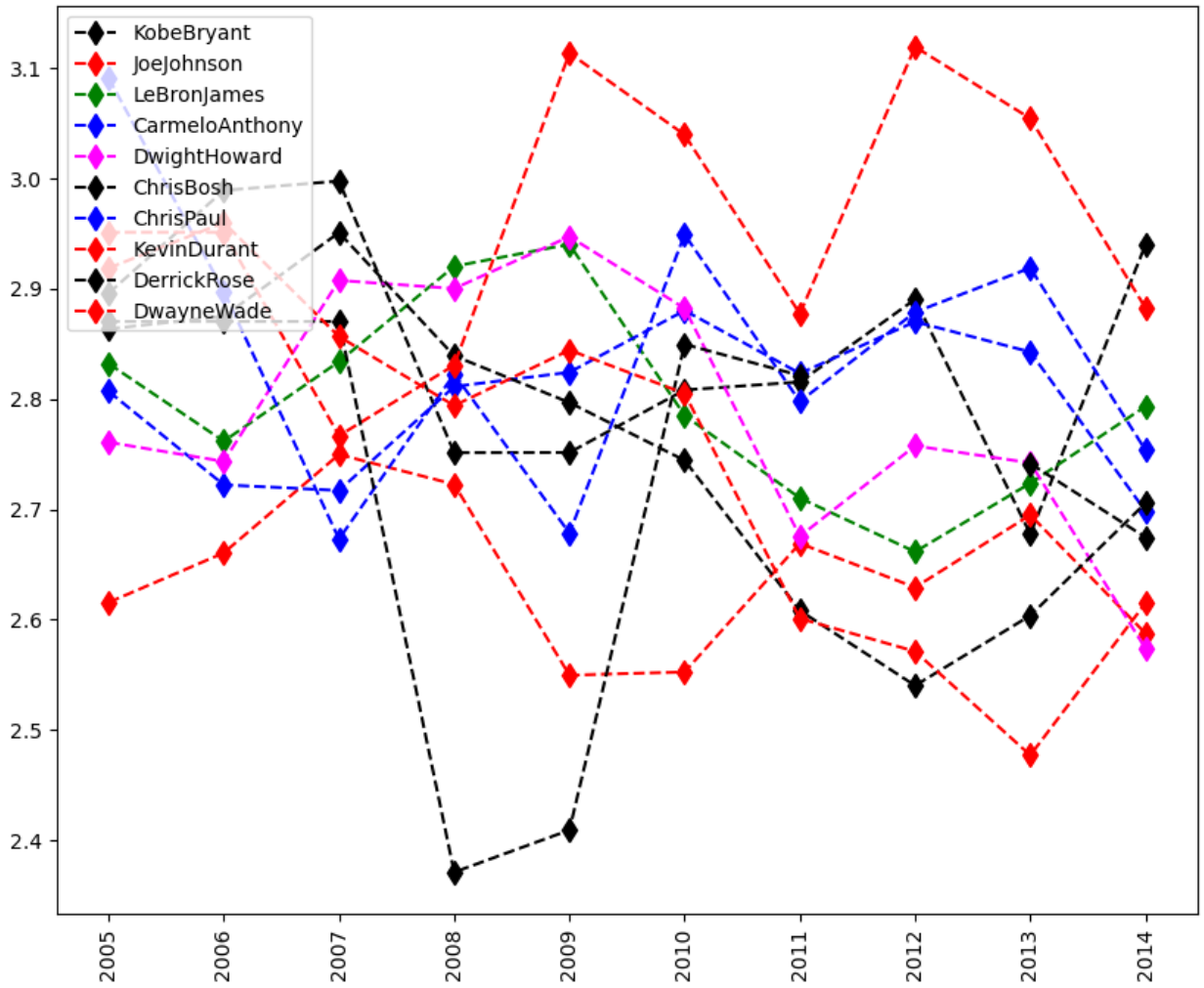
```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/2811993915.py:1: RuntimeWarn
ing: invalid value encountered in divide
  myplot(FieldGoals/MinutesPlayed)
```



In [47]:

```
1 myplot(Points/FieldGoals)
```

```
/var/folders/nn/vcmtyf8n1pl_gm88tw6nx6hw0000gn/T/ipykernel_82346/2640421752.py:1: RuntimeWarn
ing: invalid value encountered in divide
  myplot(Points/FieldGoals)
```



In [ ]:

```
1
```