In [1]:

```
#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,"20

#Players
Players = ["KobeBryant","JoeJohnson","LeBronJames","CarmeloAnthony","DwightHoward","
Pdict = {"KobeBryant":0,"JoeJohnson":1,"LeBronJames":2,"CarmeloAnthony":3,"DwightHow

#Salaries
KobeBryant_Salary = [15946875,17718750,19490625,21262500,23034375,24806250,25244493,
JoeJohnson_Salary = [12000000,12744189,13488377,14232567,14976754,16324500,18038573,
LeBronJames_Salary = [4621800,5828090,13041250,14410581,15779912,14500000,16022500,1
CarmeloAnthony_Salary = [3713640,4694041,13041250,14410581,15779912,17149243,1851857
DwightHoward_Salary = [4493160,4806720,6061274,13758000,15202590,16647180,18091770,1
ChrisBosh_Salary = [3348000,4235220,12455000,14410581,15779912,14500000,16022500,175
ChrisPaul_Salary = [3144240,3380160,3615960,4574189,13520500,14940153,16359805,17779
KevinDurant_Salary = [0,0,4171200,4484040,4796880,6053663,15506632,16669630,17832627
DerrickRose_Salary = [0,0,0,4822800,5184480,5546160,6993708,16402500,17632688,188628
DwayneWade_Salary = [3031920,3841443,13041250,14410581,15779912,14200000,15691000,17
#Matrix
Salary = np.array([KobeBryant_Salary, JoeJohnson_Salary, LeBronJames_Salary, Carmelo

#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, Dwigh

#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, CarmeloAntho

#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

#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, Carme

#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_P
```

In [2]:

```
# we want to operate in matrix form
Games
```

Out[2]:

```
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 [3]:

```
Games[0]
```

Out[3]:

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

In [4]:

```
Games[3]
```

Out[4]:

```
array([80, 65, 77, 66, 69, 77, 55, 67, 77, 40])
```

In [5]:

```
# WE WANT particular value
Games[3,0]
```

Out[5]:

```
80
```

In [6]:

```
Games[3,1]
```

Out[6]:

```
65
```

In [8]:

```
# printing all together
print(Games[3,0])
print(Games[3,1])
```

```
80
65
```

In [9]:

```
#----
```

In [9]:

```
# operating in many matrices
Points
```

Out[9]:

```
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 [11]:

```
#printing for each variables
#CASE-1
FieldGoals
```

Out[11]:

```
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 [12]:

```
#case-2
Salary
```

Out[12]:

```
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 [13]:

```
# step 1: we want field goal per game player make.
FieldGoals / Games
```

```
<ipython-input-13-8ee2c66d5b0a>:2: RuntimeWarning: invalid value encou
ntered in true_divide
  FieldGoals / Games
```

Out[13]:

```
array([[12.225     , 10.55844156,  9.45121951,  9.75609756,  9.8082191
8,
         9.02439024,  9.89655172,  9.46153846,  5.16666667,  7.6
],
       [ 7.70731707,  9.40350877,  7.8902439 ,  7.84810127,  8.3552631
6,
         7.13888889,  7.05      ,  6.18055556,  5.84810127,  5.575
],
       [11.07594937,  9.8974359 , 10.58666667,  9.74074074, 10.1052631
6,
         9.59493671, 10.01612903, 10.06578947,  9.96103896,  9.0434782
6],
       [ 9.45      , 10.63076923,  9.45454545,  8.10606061,  9.9710144
9,
         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,  6.1219512
2],
       [ 7.84285714,  7.86956522,  7.56716418,  7.98701299,  8.5714285
7,
         6.80519481,  6.89473684,  6.55405405,  6.2278481 ,  7.7954545
5],
       [ 5.21794872,  5.953125  ,  7.875     ,  8.08974359,  6.9777777
8,
         5.375     ,  7.08333333,  5.88571429,  6.5483871 ,  6.9268292
7],
       [ 8.74285714,  8.74285714,  7.3375    ,  8.93243243,  9.6829268
3,
         9.11538462,  9.74242424,  9.02469136, 10.48148148,  8.8148148
1],
       [ 5.2       ,  5.2       ,  5.2       ,  7.08641975,  8.6153846
2,
         8.77777778,  7.74358974,         nan,  5.8       ,  6.6274509
8],
       [ 9.32      ,  9.25490196,  8.60784314, 10.81012658,  9.3376623
4,
         9.10526316,  8.48979592,  8.24637681,  7.68518519,  8.2096774
2]])
```

In [14]:

```
# step2:wants to avoid error
import warnings
warnings.filterwarnings("ignore")
FieldGoals / Games
```

Out[14]:

```
array([[12.225     , 10.55844156,  9.45121951,  9.75609756,  9.8082191
8,
         9.02439024,  9.89655172,  9.46153846,  5.16666667,  7.6
],
       [ 7.70731707,  9.40350877,  7.8902439 ,  7.84810127,  8.3552631
6,
         7.13888889,  7.05      ,  6.18055556,  5.84810127,  5.575
],
       [11.07594937,  9.8974359 , 10.58666667,  9.74074074, 10.1052631
6,
         9.59493671, 10.01612903, 10.06578947,  9.96103896,  9.0434782
6],
       [ 9.45      , 10.63076923,  9.45454545,  8.10606061,  9.9710144
9,
         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,  6.1219512
2],
       [ 7.84285714,  7.86956522,  7.56716418,  7.98701299,  8.5714285
7,
         6.80519481,  6.89473684,  6.55405405,  6.2278481 ,  7.7954545
5],
       [ 5.21794872,  5.953125  ,  7.875     ,  8.08974359,  6.9777777
8,
         5.375     ,  7.08333333,  5.88571429,  6.5483871 ,  6.9268292
7],
       [ 8.74285714,  8.74285714,  7.3375    ,  8.93243243,  9.6829268
3,
         9.11538462,  9.74242424,  9.02469136, 10.48148148,  8.8148148
1],
       [ 5.2       ,  5.2       ,  5.2       ,  7.08641975,  8.6153846
2,
         8.77777778,  7.74358974,         nan,  5.8       ,  6.6274509
8],
       [ 9.32      ,  9.25490196,  8.60784314, 10.81012658,  9.3376623
4,
         9.10526316,  8.48979592,  8.24637681,  7.68518519,  8.2096774
2]])
```

In [15]:

```
# step3: now wants to nearest digit possible
np.matrix.round(FieldGoals / Games)
```

Out[15]:

```
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 [16]:

```
FieldGoalAttempts
```

Out[16]:

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

In [17]:

```
# step 4: for simplicit
fieldgoalpergame = np.matrix.round(FieldGoals / Games)
print(fieldgoalpergame)
```

```
[[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 [18]:

```
# Q2 WE WANT TO FIND ACCURACY OF PLAYER
# FORMULA = FIELD GOALS/ FIELD GOAL ATTEMPT
FieldGoals / FieldGoalAttempts
```

Out[18]:

```
array([[0.45006903, 0.46272055, 0.45857988, 0.46728972, 0.45634162,
        0.45149481, 0.42964072, 0.46269592, 0.42465753, 0.37307153],
       [0.45304659, 0.47058824, 0.43219773, 0.43661972, 0.45815296,
        0.44272179, 0.45435016, 0.4230038 , 0.45383104, 0.43512195],
       [0.47997806, 0.47624923, 0.48355664, 0.48915065, 0.5026178 ,
        0.51043771, 0.53122327, 0.56499261, 0.5668884 , 0.48788116],
       [0.48091603, 0.47556779, 0.49155976, 0.44324772, 0.45805593,
        0.45508982, 0.4302439 , 0.44929483, 0.45222155, 0.44416873],
       [0.53121453, 0.60252005, 0.59856263, 0.57201226, 0.61151079,
        0.59291188, 0.57300275, 0.57810578, 0.59125   , 0.59338061],
       [0.5050598 , 0.49634369, 0.49367089, 0.48693587, 0.51813472,
        0.49621212, 0.48698885, 0.53472988, 0.51626443, 0.46040268],
       [0.42977825, 0.43742824, 0.4879938 , 0.50278884, 0.49293564,
        0.46336207, 0.47752809, 0.48130841, 0.46666667, 0.48547009],
       [0.47295209, 0.47295209, 0.42972182, 0.47553957, 0.47601918,
        0.46228869, 0.49575944, 0.51011863, 0.50296209, 0.50963597],
       [0.47706422, 0.47706422, 0.47706422, 0.47516556, 0.48943918,
        0.44520977, 0.43453237,        nan, 0.35365854, 0.40479042],
       [0.49469214, 0.49064449, 0.46851654, 0.49108683, 0.47584381,
        0.5       , 0.49701314, 0.52058554, 0.54533509, 0.4695572 ]])
```

In [19]:

```
# STEP 2 ROUND OFF
np.matrix.round(FieldGoals / FieldGoalAttempts)
```

Out[19]:

```
array([[ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  1.,  1.,  1.,  1.,  1.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
       [ 1.,  0.,  0.,  0.,  1.,  0.,  0.,  1.,  1.,  0.],
       [ 0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  1.,  1.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0., nan,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  1.,  0.]])
```

In [20]:

```
# simplicity
accuracy = np.matrix.round(FieldGoals / FieldGoalAttempts)
print(accuracy)
```

```
[[ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  1.  1.  1.  1.  1.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
 [ 1.  0.  0.  0.  1.  0.  0.  1.  1.  0.]
 [ 0.  0.  0.  1.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  1.  1.  1.]
 [ 0.  0.  0.  0.  0.  0.  0. nan  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  1.  1.  0.]]
```

In [ ]:

```
# these are the results of accuracy of player in each seasons.
```