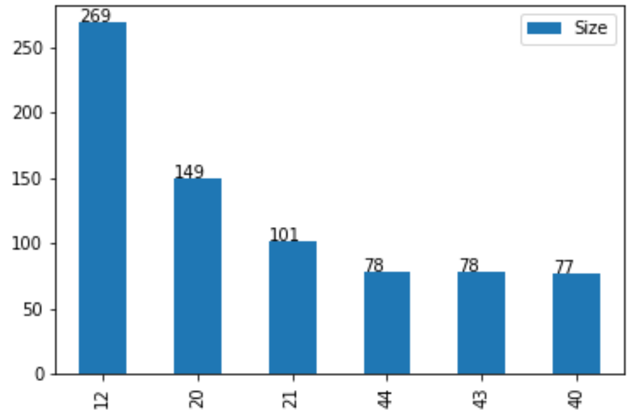
In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


schedule=pd.read_excel('Marshall_Course_Enrollment_1516_1617.xlsx')
cancelled=pd.read_excel('Cancelled_Courses_1516_1617.xlsx')
master=schedule.append(cancelled)
capacities=pd.read_excel('Marshall_Room_Capacity_Chart.xlsx')


capacities.head()


capacities = capacities.drop(capacities.columns[[2,3,4,5,6,7,8]],axis = 1)

 # 45 obversations so 45 classrooms
```

In [76]:
```
capacities_descending = capacities.sort_values(by = "Size", ascending = False)
capacities_descending = capacities_descending.head(6)
capacities_descending
```

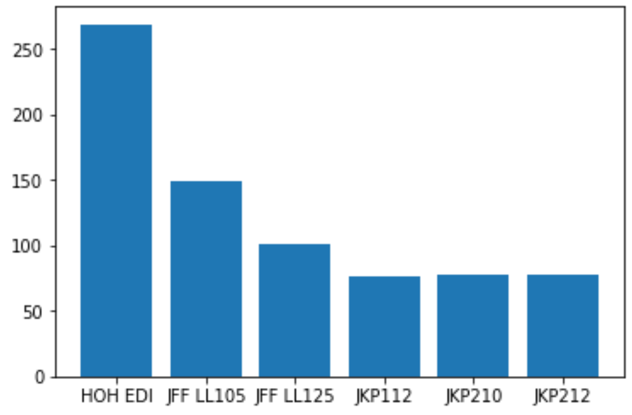
Out[76]:

	Room	Size	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
12	HOH EDI	269	NaN	NaN	NaN	NaN	NaN	NaN	NaN
20	JFF LL105	149	NaN	NaN	NaN	NaN	NaN	NaN	NaN
21	JFF LL125	101	NaN	NaN	NaN	NaN	NaN	NaN	NaN
44	JKP212	78	NaN	NaN	NaN	NaN	NaN	NaN	NaN
43	JKP210	78	NaN	NaN	NaN	NaN	NaN	NaN	NaN
40	JKP112	77	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [34]:
```
df = capacities_descending

ax = df.plot(kind='bar')
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() , p.get_height() ))

plt.show()
```



In [77]:
```
#Showing which room has the highest capacity
plt.bar(capacities_descending['Room'], capacities_descending['Size'])

plt.show()
```



In [79]:
```
# Finding prime time, looking how many classes start at which "begin time"

schedule_group = schedule.groupby( by = ['First Begin Time', 'First End Time'])

schedule_group_primetime = schedule_group.size()

schedule_group_primetime
```

```
Out[79]: First Begin Time  First End Time
         08:00:00          08:50:00             7
                           09:10:00             3
                           09:20:00            51
                           09:40:00             8
                           09:45:00             2
                           09:50:00           147
                           17:00:00             9
                           17:45:00             2
                           18:00:00             1
         08:15:00          10:45:00             2
         08:30:00          09:40:00             5
                           11:00:00             4
                           11:30:00             1
                           16:30:00            15
         08:45:00          11:15:00             1
         08:50:00          16:30:00            55
                           16:50:00             2
         09:00:00          09:50:00             5
                           10:20:00             1
                           10:50:00             1
                           11:00:00             2
                           11:20:00             8
                           11:25:00             3
                           11:30:00            16
                           11:40:00             1
                           11:45:00             1
                           11:50:00            15
                           12:00:00             9
                           12:50:00            12
                           13:00:00             2
                                              ...
         17:30:00          20:30:00             1
         18:00:00          19:00:00             2
                           19:10:00            11
                           19:20:00             1
                           19:30:00             2
                           19:50:00           136
                           20:20:00             3
                           20:30:00             4
                           20:50:00             7
                           21:00:00             7
                           21:10:00             1
                           21:20:00             5
                           21:50:00             2
                           22:00:00            91
         18:30:00          19:30:00             1
                           19:50:00            18
                           20:00:00             1
                           20:20:00             4
                           20:30:00             1
                           21:20:00             4
                           21:30:00           189
         18:40:00          19:50:00             6
         19:00:00          20:00:00             1
                           20:30:00             2
                           20:50:00             1
                           21:50:00             1
         19:20:00          20:30:00            10
         20:00:00          21:50:00            16
         20:40:00          21:50:00             7
         TBA               TBA                 98
```

In [82]:
```
schedule_group_primetime.sort_values(ascending = False)

#top 10 begin prime time

top5_primetime = schedule_group_primetime.sort_values(ascending = False).head(5)


top5_primetime

# the prime time is from 14:00 pm - 15:50 pm with 245 course on a weekly basis
```

Out[82]:
```
First Begin Time  First End Time
14:00:00          15:50:00          245
12:00:00          13:50:00          243
10:00:00          11:50:00          243
16:00:00          17:50:00          219
18:30:00          21:30:00          189
dtype: int64
```

In [85]:
```
capacities
```

Out[85]:

	Room	Size
0	ACC 306B	16
1	ACC201	48
2	ACC205	36
3	ACC236	39
4	ACC303	46
5	ACC306B	16
6	ACC310	54
7	ACC312	20
8	BRI202	42
9	BRI202A	34
10	BRI5	42
11	BRI8	36
12	HOH EDI	269
13	HOH1	73
14	HOH2	73
15	HOH506	16
16	HOH706	16
17	JFF LL101	48
18	JFF LL102	48
19	JFF LL103	48
20	JFF LL105	149
21	JFF LL125	101
22	JFF233	60
23	JFF236	60
24	JFF239	48
25	JFF240	48
26	JFF241	48
27	JFF312	20
28	JFF313	20
29	JFF316	48
30	JFF322	48
31	JFF327	36
32	JFF328	36
33	JFF331	36

In [40]:
```
schedule1 = schedule[['First Begin Time', 'First End Time' ,'First Room', 'Reg Co
unt', 'Seats']]

schedule1

#Filtering only by the biggest prime time : 14:00 - 15:50pm

schedule1.dtypes
```

Out[40]:
```
First Begin Time    object
First End Time      object
First Room          object
Reg Count            int64
Seats                int64
dtype: object
```

```
In [65]: import pandas as pd
         schedule=pd.read_excel('Marshall_Course_Enrollment_1516_1617.xlsx')
         cancelled=pd.read_excel('Cancelled_Courses_1516_1617.xlsx')
         master=schedule.append(cancelled)
         capacities=pd.read_excel('Marshall_Room_Capacity_Chart.xlsx')

         master.info()
         master.to_csv('Merged_Enrollment.csv')
         pd.set_option("display.max_columns",100)   # Display all columns so you can see t
         he DataFrame better.
         master.head()



         #Importing packages
         import pandas as pd
         import numpy as np

         #Reading excel files
         master=pd.read_csv('Merged_Enrollment.csv')
         capacities=pd.read_excel('Marshall_Room_Capacity_Chart.xlsx')

         roomSet=set(capacities.Room)      # Create a set which contains the rooms in the
         capacities file.
         ans={}                  # Initialize a dictionary to store the result we want

         df=master
         master.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6131 entries, 0 to 3231
Data columns (total 26 columns):
Course                   3363 non-null object
Course Prefix            6131 non-null object
Course Suffix            6131 non-null object
Department               3363 non-null object
First Begin Time         5629 non-null object
First Days               5553 non-null object
First End Time           5629 non-null object
First Instructor         5664 non-null object
First Instructor UID     5645 non-null float64
First Room               5696 non-null object
Link                     936 non-null object
Max Units                6131 non-null float64
Min Units                6131 non-null float64
Mode                     6131 non-null object
Reg Count                6131 non-null int64
Seats                    6131 non-null int64
Second Begin Time        43 non-null object
Second Days              43 non-null object
Second End Time          43 non-null object
Second Instructor        595 non-null object
Second Instructor UID    595 non-null float64
Second Room              19 non-null object
Section                  6131 non-null int64
Session                  6131 non-null int64
Term                     6131 non-null int64
Title                    6131 non-null object
dtypes: float64(4), int64(5), object(17)
memory usage: 1.3+ MB
```

Out[65]:

	Unnamed: 0	Course	Course Prefix	Course Suffix	Department	First Begin Time	First Days	First End Time	First Instructor	I Instru
0	0	ACCT-370	ACCT	370	ACCT	10:00:00	F	11:50:00	Hopkins, Merle, W	3.783354e
1	1	ACCT-370	ACCT	370	ACCT	08:00:00	MW	09:50:00	Hopkins, Merle, W	3.783354e
2	2	ACCT-370	ACCT	370	ACCT	10:00:00	MW	11:50:00	Hopkins, Merle, W	3.783354e
3	3	ACCT-370	ACCT	370	ACCT	12:00:00	MW	13:50:00	Hopkins, Merle, W	3.783354e
4	4	ACCT-371	ACCT	371	ACCT	10:00:00	F	11:50:00	NaN	NaN

In [222]:
```
# Lets analyze the prime time where begin time = 14:00:00 and end time is 15:50pm
first_primetime = df[(df['First Begin Time'] == '14:00:00') & (df['First End Time
'] == '15:50:00') & (df['First Days'] == 'TH')]

first_primetime = first_primetime[['First Days','First Begin Time', 'First End Ti
me','First Room', 'Reg Count' , 'Seats']]

first_primetime

first_primetime['Utilization'] = first_primetime['Reg Count'] / first_primetime [
'Seats']

print(first_primetime)


first_primetime['Utilization'].mean()
```

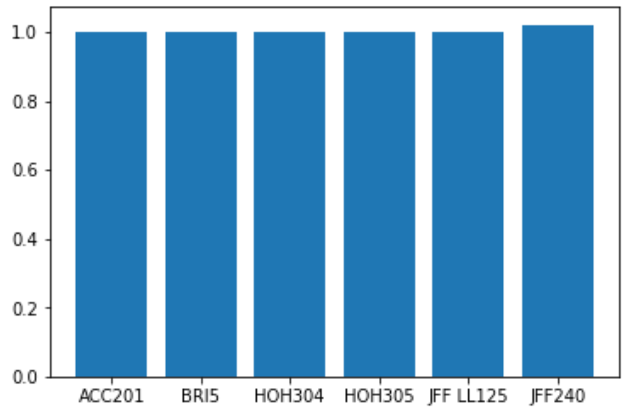
```
     First Days First Begin Time First End Time First Room  Reg Count  Seats  \
41           TH         14:00:00       15:50:00     ACC310         33     40
43           TH         14:00:00       15:50:00     ACC303         23     40
93           TH         14:00:00       15:50:00       HOH1         59     61
137          TH         14:00:00       15:50:00     HOH301         39     40
140          TH         14:00:00       15:50:00     HOH304         46     46
145          TH         14:00:00       15:50:00     ACC303         44     45
166          TH         14:00:00       15:50:00     HOH306         30     32
187          TH         14:00:00       15:50:00     ACC236         33     34
228          TH         14:00:00       15:50:00     HOH422         39     40
335          TH         14:00:00       15:50:00     JKP104         48     48
353          TH         14:00:00       15:50:00     ACC205         15     30
392          TH         14:00:00       15:50:00       HOH2         46     65
400          TH         14:00:00       15:50:00     JKP110         70     75
542          TH         14:00:00       15:50:00     HOH421         32     40
550          TH         14:00:00       15:50:00     HOH305         40     40
573          TH         14:00:00       15:50:00     ACC201         32     32
575          TH         14:00:00       15:50:00       BRI5         40     40
577          TH         14:00:00       15:50:00     HOH303         47     48
662          TH         14:00:00       15:50:00     ACC310         49     50
665          TH         14:00:00       15:50:00     ACC310         44     45
667          TH         14:00:00       15:50:00     HOH305         32     38
668          TH         14:00:00       15:50:00     ACC303         41     42
685          TH         14:00:00       15:50:00     HOH421         42     45
742          TH         14:00:00       15:50:00    SOS B46         56     60
749          TH         14:00:00       15:50:00       HOH1         47     52
750          TH         14:00:00       15:50:00     HOH422         27     29
759          TH         14:00:00       15:50:00     HOH304         36     40
820          TH         14:00:00       15:50:00     ACC205         25     32
852          TH         14:00:00       15:50:00     HOH302         30     31
858          TH         14:00:00       15:50:00     HOH301         38     39
...         ...              ...            ...        ...        ...    ...
5122         TH         14:00:00       15:50:00     THH210         70     70
5132         TH         14:00:00       15:50:00       HOH1         72     72
5141         TH         14:00:00       15:50:00       HOH2         73     73
5150         TH         14:00:00       15:50:00     JKP110         73     75
5168         TH         14:00:00       15:50:00  JFF LL105         75     75
5170         TH         14:00:00       15:50:00  JFF LL125         75     75
5253         TH         14:00:00       15:50:00        NaN          0      1
5438         TH         14:00:00       15:50:00     ACC303         21     46
5490         TH         14:00:00       15:50:00       HOH2         26     68
5516         TH         14:00:00       15:50:00     JFF327         29     32
5523         TH         14:00:00       15:50:00     ACC201         29     48
5525         TH         14:00:00       15:50:00     JFF236         29     60
5529         TH         14:00:00       15:50:00       BRI5         30     31
5546         TH         14:00:00       15:50:00     JFF331         31     32
5657         TH         14:00:00       15:50:00     ACC236         36     39
5664         TH         14:00:00       15:50:00     JFF414         36     46
5667         TH         14:00:00       15:50:00     ACC236         37     37
5683         TH         14:00:00       15:50:00     JFF416         38     48
5694         TH         14:00:00       15:50:00     KAP158         39     45
5725         TH         14:00:00       15:50:00     JFF241         40     48
5734         TH         14:00:00       15:50:00  JFF LL101         41     42
5781         TH         14:00:00       15:50:00     JFF322         46     48
5795         TH         14:00:00       15:50:00     JFF239         47     47
5801         TH         14:00:00       15:50:00  JFF LL103         47     48
5808         TH         14:00:00       15:50:00     JFF240         48     47
5881         TH         14:00:00       15:50:00     ACC310         54     54
5922         TH         14:00:00       15:50:00  JFF LL125         70     70
5926         TH         14:00:00       15:50:00       HOH1         71     77
5942         TH         14:00:00       15:50:00     THH208         74     74
5956         TH         14:00:00       15:50:00     JKP110         76     76
```

```
Out[222]: 0.8701804027846632
```

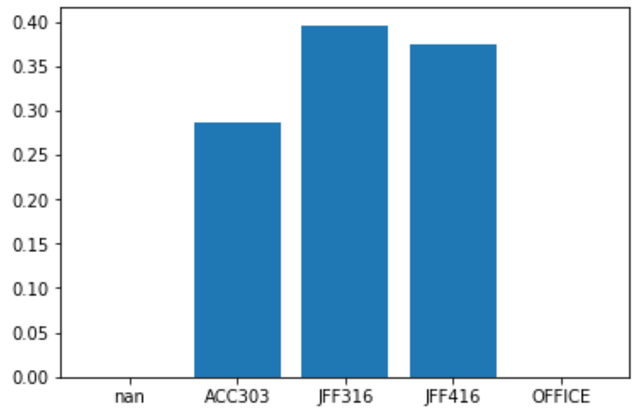
In [224]:

```
#Exclude the one with NA Values in First Room
top6_lowest = first_primetime.sort_values(by = "Utilization", ascending = False)
top6_lowest = top6_lowest[pd.notnull(top6_lowest['First Room'])]


#Excluse office
top6_lowest = top6_lowest[((top6_lowest['First Room'] != 'OFFICE') & (top6_lowest
['First Room'] != 'HOH EDI' ))]

top6_lowest = top6_lowest.head(7)

print(top6_lowest)

plt.bar(top6_lowest['First Room'], top6_lowest['Utilization'])
plt.show()
```

```
     First Days First Begin Time First End Time First Room  Reg Count  Seats  \
5808         TH         14:00:00       15:50:00     JFF240         48     47
2464         TH         14:00:00       15:50:00     JFF240         48     47
3180         TH         14:00:00       15:50:00     ACC201         32     32
1856         TH         14:00:00       15:50:00  JFF LL125         75     75
3408         TH         14:00:00       15:50:00     HOH304         46     46
3358         TH         14:00:00       15:50:00       BRI5         40     40
3356         TH         14:00:00       15:50:00     HOH305         40     40

      Utilization
5808     1.021277
2464     1.021277
3180     1.000000
1856     1.000000
3408     1.000000
3358     1.000000
3356     1.000000
```



In [212]:
```
plt.bar(top6_lowest['First Room'], top6_lowest['Utilization'])
plt.show()
```



In [133]:

```
second_primetime = df[(df['First Begin Time'] == '12:00:00') & (df['First End Tim
e'] == '13:50:00')]

second_primetime = second_primetime[['First Begin Time', 'First End Time','First
Room', 'Reg Count' , 'Seats']]

second_primetime

second_primetime['Utilization'] = second_primetime['Reg Count'] / second_primetim
e [ 'Seats']

print(second_primetime)


second_primetime['Utilization'].mean()
```

```
     First Begin Time First End Time First Room  Reg Count  Seats  Utilization
3            12:00:00       13:50:00     ACC303         42     42     1.000000
6            12:00:00       13:50:00     ACC303         40     42     0.952381
8            12:00:00       13:50:00    HOH EDI        144    269     0.535316
10           12:00:00       13:50:00     ACC310         54     54     1.000000
12           12:00:00       13:50:00    HOH EDI        142    269     0.527881
14           12:00:00       13:50:00     ACC310         49     50     0.980000
18           12:00:00       13:50:00     ACC310         52     52     1.000000
20           12:00:00       13:50:00     ACC310         47     48     0.979167
26           12:00:00       13:50:00     HOH422         40     40     1.000000
34           12:00:00       13:50:00     ACC205         33     34     0.970588
44           12:00:00       13:50:00     ACC201         25     40     0.625000
46           12:00:00       13:50:00     ACC205         36     36     1.000000
89           12:00:00       13:50:00       HOH2         56     60     0.933333
122          12:00:00       13:50:00     HOH422         20     40     0.500000
133          12:00:00       13:50:00     HOH301         38     38     1.000000
134          12:00:00       13:50:00     HOH301         38     39     0.974359
146          12:00:00       13:50:00     ACC201         46     46     1.000000
151          12:00:00       13:50:00     HOH421         44     44     1.000000
154          12:00:00       13:50:00     ACC205         29     36     0.805556
168          12:00:00       13:50:00     HOH306         30     32     0.937500
173          12:00:00       13:50:00     HOH302         31     32     0.968750
180          12:00:00       13:50:00     HOH302         32     32     1.000000
184          12:00:00       13:50:00     HOH421         35     35     1.000000
188          12:00:00       13:50:00       BRI8         20     35     0.571429
203          12:00:00       13:50:00       BRI8         35     35     1.000000
204          12:00:00       13:50:00       BRI8         35     35     1.000000
211          12:00:00       13:50:00       BRI8         36     36     1.000000
226          12:00:00       13:50:00     HOH422         37     40     0.925000
235          12:00:00       13:50:00       HOH2         73     73     1.000000
242          12:00:00       13:50:00       BRI8         36     36     1.000000
...               ...            ...        ...        ...    ...          ...
5791         12:00:00       13:50:00       HOH1         46     70     0.657143
5796         12:00:00       13:50:00     JFF416         47     47     1.000000
5797         12:00:00       13:50:00     ACC310         47     47     1.000000
5798         12:00:00       13:50:00     JFF239         47     47     1.000000
5803         12:00:00       13:50:00  JFF LL101         47     48     0.979167
5809         12:00:00       13:50:00     JFF239         48     47     1.021277
5814         12:00:00       13:50:00  JFF LL103         48     48     1.000000
5817         12:00:00       13:50:00  JFF LL103         48     48     1.000000
5823         12:00:00       13:50:00     JFF322         48     48     1.000000
5826         12:00:00       13:50:00     JFF241         48     48     1.000000
5831         12:00:00       13:50:00  JFF LL103         48     48     1.000000
5833         12:00:00       13:50:00  JFF LL103         48     48     1.000000
5834         12:00:00       13:50:00     KAP156         48     49     0.979592
5849         12:00:00       13:50:00     ZHS163         49     49     1.000000
5871         12:00:00       13:50:00     JFF236         52     53     0.981132
5885         12:00:00       13:50:00     ACC310         55     55     1.000000
5897         12:00:00       13:50:00     JFF414         59     60     0.983333
5904         12:00:00       13:50:00  JFF LL125         64     70     0.914286
5913         12:00:00       13:50:00       HOH2         67     68     0.985294
5920         12:00:00       13:50:00  JFF LL125         69     70     0.985714
5934         12:00:00       13:50:00     JKP110         73     73     1.000000
5962         12:00:00       13:50:00  JFF LL125         77     79     0.974684
5967         12:00:00       13:50:00    WPH B27         79     79     1.000000
5968         12:00:00       13:50:00  JFF LL125         79     80     0.987500
5971         12:00:00       13:50:00  JFF LL105         80     80     1.000000
5985         12:00:00       13:50:00    HOH EDI        119    120     0.991667
5990         12:00:00       13:50:00    HOH EDI        143    196     0.729592
5995         12:00:00       13:50:00    HOH EDI        198    200     0.990000
5996         12:00:00       13:50:00    HOH EDI        199    199     1.000000
6040         12:00:00       13:50:00     KAP163         12     31     0.387097
```

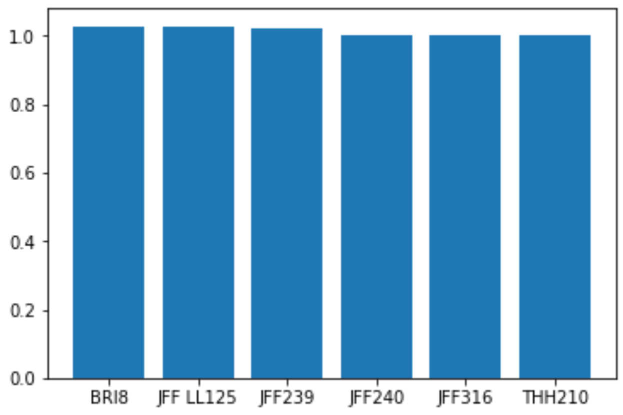
```
Out[133]: 0.9144180077073247
```

```
In [175]: top6_lowest2 = second_primetime.sort_values(by = "Utilization", ascending = False
          )
          top6_lowest2 = top6_lowest2[pd.notnull(top6_lowest2['First Room'])]

          top6_lowest2 = top6_lowest2[(top6_lowest2['First Room'] != 'BRI202A')]

          top6_lowest2 = top6_lowest2.head(11)
          top6_lowest2
```

Out[175]:

	First Begin Time	First End Time	First Room	Reg Count	Seats	Utilization
4029	12:00:00	13:50:00	BRI8	36	35	1.028571
883	12:00:00	13:50:00	BRI8	36	35	1.028571
5164	12:00:00	13:50:00	JFF LL125	75	73	1.027397
1847	12:00:00	13:50:00	JFF LL125	75	73	1.027397
5809	12:00:00	13:50:00	JFF239	48	47	1.021277
2463	12:00:00	13:50:00	JFF239	48	47	1.021277
2024	12:00:00	13:50:00	JFF316	48	48	1.000000
1777	12:00:00	13:50:00	JFF239	48	48	1.000000
4913	12:00:00	13:50:00	JFF240	40	40	1.000000
4903	12:00:00	13:50:00	JFF316	40	40	1.000000
1837	12:00:00	13:50:00	THH210	60	60	1.000000

```
In [177]: plt.bar(top6_lowest2['First Room'], top6_lowest2['Utilization'])
          plt.show()
```



In [225]:
```
df['Utilization'] = df['Reg Count'] / df [ 'Seats']

dfsort = df.groupby( by  = ['First Days' , 'First Begin Time', 'First End Time'])
.size()

dfsort.sort_values(ascending = False)
```

```
Out[225]: First Days  First Begin Time  First End Time
          MW          14:00:00          15:50:00          190
          TH          14:00:00          15:50:00          180
                      10:00:00          11:50:00          179
                      16:00:00          17:50:00          178
                      12:00:00          13:50:00          175
          MW          10:00:00          11:50:00          173
                      16:00:00          17:50:00          165
                      12:00:00          13:50:00          162
          W           18:30:00          21:30:00          112
          M           18:30:00          21:30:00          111
          MW          08:00:00          09:50:00          110
          T           18:30:00          21:30:00          107
          FS          08:50:00          16:30:00          102
          TH          08:00:00          09:50:00           96
          F           10:00:00          11:50:00           90
          H           18:30:00          21:30:00           87
          MW          11:00:00          12:20:00           84
          TH          18:00:00          19:50:00           77
                      12:30:00          13:50:00           77
          MW          12:30:00          13:50:00           71
          F           12:00:00          13:50:00           68
          TH          09:30:00          10:50:00           67
          MW          09:30:00          10:50:00           63
          TH          14:00:00          15:20:00           63
          MW          18:00:00          19:50:00           60
          TH          11:00:00          12:20:00           54
          MW          14:00:00          15:20:00           53
                      17:00:00          18:20:00           52
          TH          17:00:00          18:20:00           52
          MW          15:30:00          16:50:00           52
                                                         ...
          M           19:00:00          20:00:00            1
          F           12:00:00          14:50:00            1
          T           09:00:00          12:00:00            1
          W           08:30:00          11:30:00            1
          MW          11:00:00          12:00:00            1
          T           11:00:00          12:50:00            1
          H           17:00:00          18:50:00            1
                      18:00:00          21:00:00            1
                      16:00:00          18:50:00            1
                      15:30:00          18:50:00            1
          T           13:00:00          16:00:00            1
          H           13:00:00          16:00:00            1
                                        15:50:00            1
                      12:00:00          15:00:00            1
          T           16:40:00          18:00:00            1
                      18:00:00          19:00:00            1
          H           09:00:00          12:00:00            1
          F           12:30:00          15:30:00            1
          H           09:00:00          11:50:00            1
          MW          09:00:00          15:30:00            1
          FS          TBA               TBA                 1
          F           TBA               TBA                 1
          H           17:30:00          20:30:00            1
          TH          18:00:00          21:10:00            1
                                        21:50:00            1
          M           13:30:00          16:00:00            1
          TWH         14:00:00          15:50:00            1
          F           12:30:00          16:50:00            1
                                        16:00:00            1
          T           14:30:00          16:00:00            1
```

In [228]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

def convert(inputTime):
    # Code copy pasted from challenge 1, except first convert input to string bef
ore splitting
    try:
        hh,mm,ss=str(inputTime).split(':')
        ans=int(hh)+int(mm)/60+int(ss)/3600
    except:
        ans=np.nan
    return ans


def loadDataDict(df,roomSet):
    # Code copy pasted from challenge 2, except adding entries to a list instead
of finding beginning and end, and adding empty list for unused classrooms
    ans={}
    # Start with empty lists in all classrooms
    terms=[20153,20161,20162,20163,20171,20172]
    for term in terms:
        for room in roomSet:
            for day in 'MTWHF':
                ans[term,room,day]=[]
    for index,row in df.iterrows():
        term=row['Term']       # Obtain the corresponding column of each row
        room=row['First Room']
        days=row['First Days']
        beg=convert(row['First Begin Time'])   # Convert the begin time strings i
nto decimal numbers using challenge 1
        end=convert(row['First End Time'])     # Convert the begin time strings i
nto decimal numbers using challenge 1
        # Skip rows in which beg and end are np.nan (not a number), and in which
the room is not in the capacity file
        #import pdb; pdb.set_trace()
        if np.isnan(beg) or np.isnan(end) or room not in roomSet:
            continue     # Command to skip this iteration of the loop
        for day in 'MTWHF':   # Iterate through the sequence ['M','T','W','H','F'
]
            if day in days:
                ans[term,room,day].append([beg,end])

    return ans

def computeUsage(inputList, primeStart,primeEnd):
    # Code copy pasted from challenge 3, except sorting the inputList
    sortedList=sorted(inputList)
    usage=0
    prev=0
    for start,end in sortedList:
        if end<primeStart:
            continue
        if start>primeEnd:
            break
        start=max(prev,start)
        end=max(prev,end)
        overlap=max(0,min(primeEnd,end)-max(primeStart,start))
        usage+=overlap
        prev=end
    return usage/(primeEnd-primeStart)
```

Out[228]:

	Term	Room	Day	Utilization
0	20153	JKP110	M	0.750000
1	20153	JKP110	T	0.916667
2	20153	JKP110	W	0.750000
3	20153	JKP110	H	0.916667
4	20153	JKP110	F	0.000000
5	20153	ACC303	M	0.916667
6	20153	ACC303	T	0.916667
7	20153	ACC303	W	0.916667
8	20153	ACC303	H	0.916667
9	20153	ACC303	F	0.583333
10	20153	HOH706	M	0.305556
11	20153	HOH706	T	0.416667
12	20153	HOH706	W	0.083333
13	20153	HOH706	H	0.000000
14	20153	HOH706	F	0.000000
15	20153	JKP102	M	0.666667
16	20153	JKP102	T	0.888889
17	20153	JKP102	W	0.666667
18	20153	JKP102	H	0.888889
19	20153	JKP102	F	0.000000
20	20153	BRI202A	M	0.916667
21	20153	BRI202A	T	0.916667
22	20153	BRI202A	W	0.916667
23	20153	BRI202A	H	0.916667
24	20153	BRI202A	F	0.000000
25	20153	JFF327	M	0.000000
26	20153	JFF327	T	0.000000
27	20153	JFF327	W	0.000000
28	20153	JFF327	H	0.000000
29	20153	JFF327	F	0.000000
...	...	...	...	...
1320	20172	ACC205	M	0.000000
1321	20172	ACC205	T	0.000000
1322	20172	ACC205	W	0.000000

In [242]:
```
average_output = (output.groupby(['Room'], as_index=False).mean()
            .groupby('Room')['Utilization'].mean())

average_output.sort_values(ascending = True).head(10)
```

Out[242]:
```
Room
ACC 306B     0.000000
ACC306B      0.000000
JFF417       0.118519
JFF LL103    0.188889
JFF414       0.203704
JFF LL105    0.218981
HOH706       0.223148
HOH506       0.224074
JFF233       0.238889
JFF LL102    0.240741
Name: Utilization, dtype: float64
```

In [245]:
```
average_output.sort_values(ascending = False).head(10)
```

Out[245]:
```
Room
HOH2       0.730556
HOH1       0.699074
ACC303     0.633333
JKP110     0.631481
HOH EDI    0.552315
JKP210     0.550000
JKP202     0.549074
ACC201     0.548148
JKP204     0.514815
ACC310     0.508333
Name: Utilization, dtype: float64
```