In [14]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [15]:
```
import os
```

In [4]:
```
os.listdir(r"C:\Users\usr\Downloads\uber-project\Uber\Datasets")
```

Out[4]:
```
['other-American_B01362.csv',
 'other-Carmel_B00256.csv',
 'other-Dial7_B00887.csv',
 'other-Diplo_B01196.csv',
 'other-Federal_02216.csv',
 'other-FHV-services_jan-aug-2015.csv',
 'other-Firstclass_B01536.csv',
 'other-Highclass_B01717.csv',
 'other-Lyft_B02510.csv',
 'other-Prestige_B01338.csv',
 'other-Skyline_B00111.csv',
 'Uber-Jan-Feb-FOIL.csv',
 'uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15.csv',
 'uber-raw-data-janjune-15_sample.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [7]:
```
uber_15 = pd.read_csv(r"C:\Users\usr\Downloads\uber-project\Uber\Datasets/uber-raw-data-
```

In [8]:
```
uber_15.shape
```

Out[8]:
```
(100000, 4)
```

In [9]:
```
type(uber_15)
```

Out[9]:
```
pandas.core.frame.DataFrame
```

In [ ]:
```
#check for duplicates and remove them
```

In [11]:
```
uber_15.duplicated().sum()
```

Out[11]:
```
54
```

In [12]:
```
uber_15.drop_duplicates(inplace = True)
```

In [13]:
```
uber_15.duplicated().sum()
```

Out[13]:
```
0
```

In [14]:
```
uber_15.shape
```

Out[14]:
```
(99946, 4)
```

In [15]:
```
uber_15.dtypes
```

Out[15]:
```
Dispatching_base_num    object
```

```
Pickup_date             object
Affiliated_base_num     object
locationID               int64
dtype: object
```

In [16]:
```
uber_15.isnull().sum()
```

Out[16]:
```
Dispatching_base_num       0
Pickup_date                0
Affiliated_base_num     1116
locationID                 0
dtype: int64
```

In [18]:
```
uber_15["Pickup_date"][0]
```

Out[18]:
```
'2015-05-02 21:43:00'
```

In [ ]:
```
#check and convert date format
```

In [21]:
```
type(uber_15["Pickup_date"][0])
```

Out[21]:
```
str
```

In [27]:
```
uber_15['Pickup_date'] = pd.to_datetime(uber_15['Pickup_date'])
```

In [29]:
```
uber_15["Pickup_date"].dtype
```

Out[29]:
```
dtype('<M8[ns]')
```

In [30]:
```
uber_15.dtypes
```

Out[30]:
```
Dispatching_base_num            object
Pickup_date             datetime64[ns]
Affiliated_base_num             object
locationID                       int64
dtype: object
```

In [ ]:
```
#Which month have max Uber pickups in New York city?
```

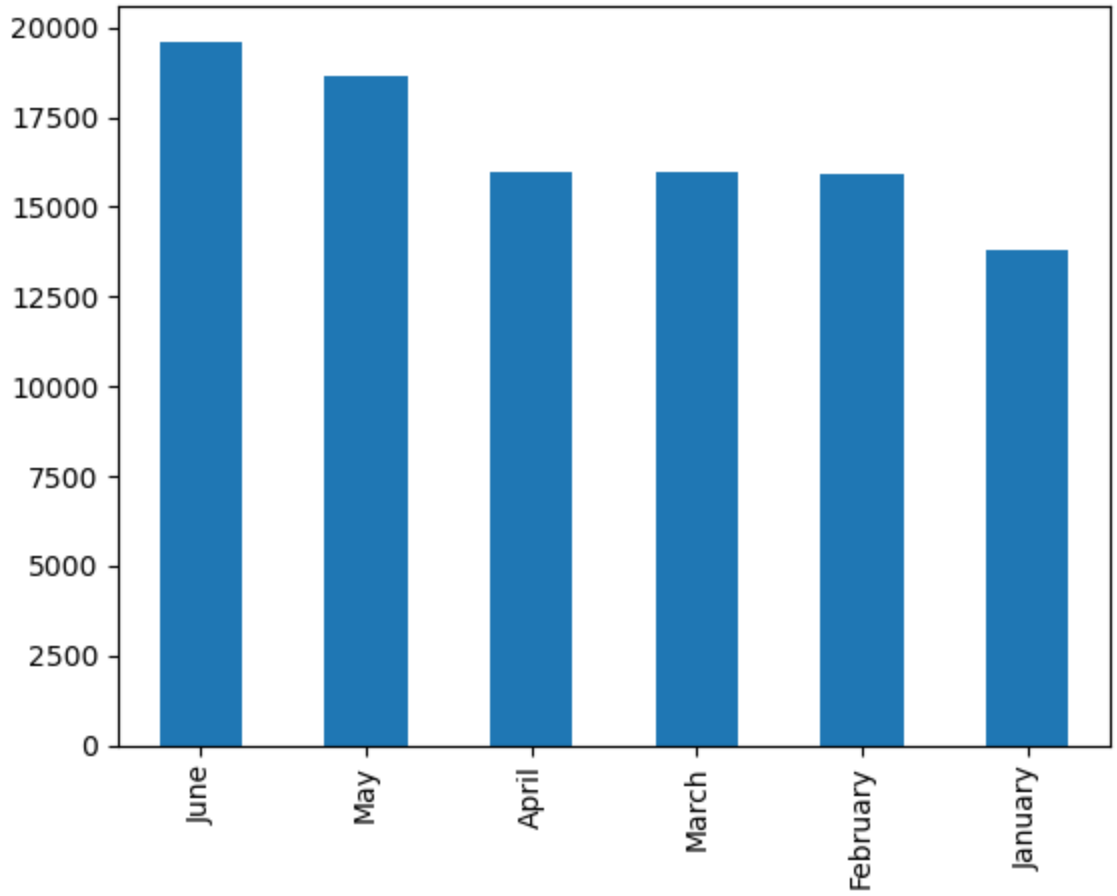
In [33]:
```
uber_15['month'] = uber_15['Pickup_date'].dt.month_name()
```

In [34]:
```
uber_15['month']
```

Out[34]:
```
0            May
1        January
2          March
3          April
4          March
           ...
99995      April
99996      March
99997      March
99998        May
99999       June
Name: month, Length: 99946, dtype: object
```

In [39]:
```
uber_15['month'].value_counts().plot(kind='bar')
```

Out[39]:
```
<Axes: >
```



In [40]:
```
#Now find weekday
```

In [42]:
```
uber_15['weekday'] = uber_15['Pickup_date'].dt.day_name()
uber_15['day'] = uber_15['Pickup_date'].dt.day
uber_15['hour'] = uber_15['Pickup_date'].dt.hour
uber_15['minute'] = uber_15['Pickup_date'].dt.minute
```

In [43]:
```
uber_15.head(4)
```

Out[43]:

	Dispatching_base_num	Pickup_date	Affiliated_base_num	locationID	month	weekday	day	hour	minute
0	B02617	2015-05-02 21:43:00	B02764	237	May	Saturday	2	21	43
1	B02682	2015-01-20 19:52:59	B02682	231	January	Tuesday	20	19	52
2	B02617	2015-03-19 20:26:00	B02617	161	March	Thursday	19	20	26
3	B02764	2015-04-10 17:38:00	B02764	107	April	Friday	10	17	38

In [ ]:
```
#pivot
```

In [44]:
```
pivot = pd.crosstab(index = uber_15['month'], columns = uber_15['weekday'])
```

In [45]:
```
pivot
```

Out[45]:

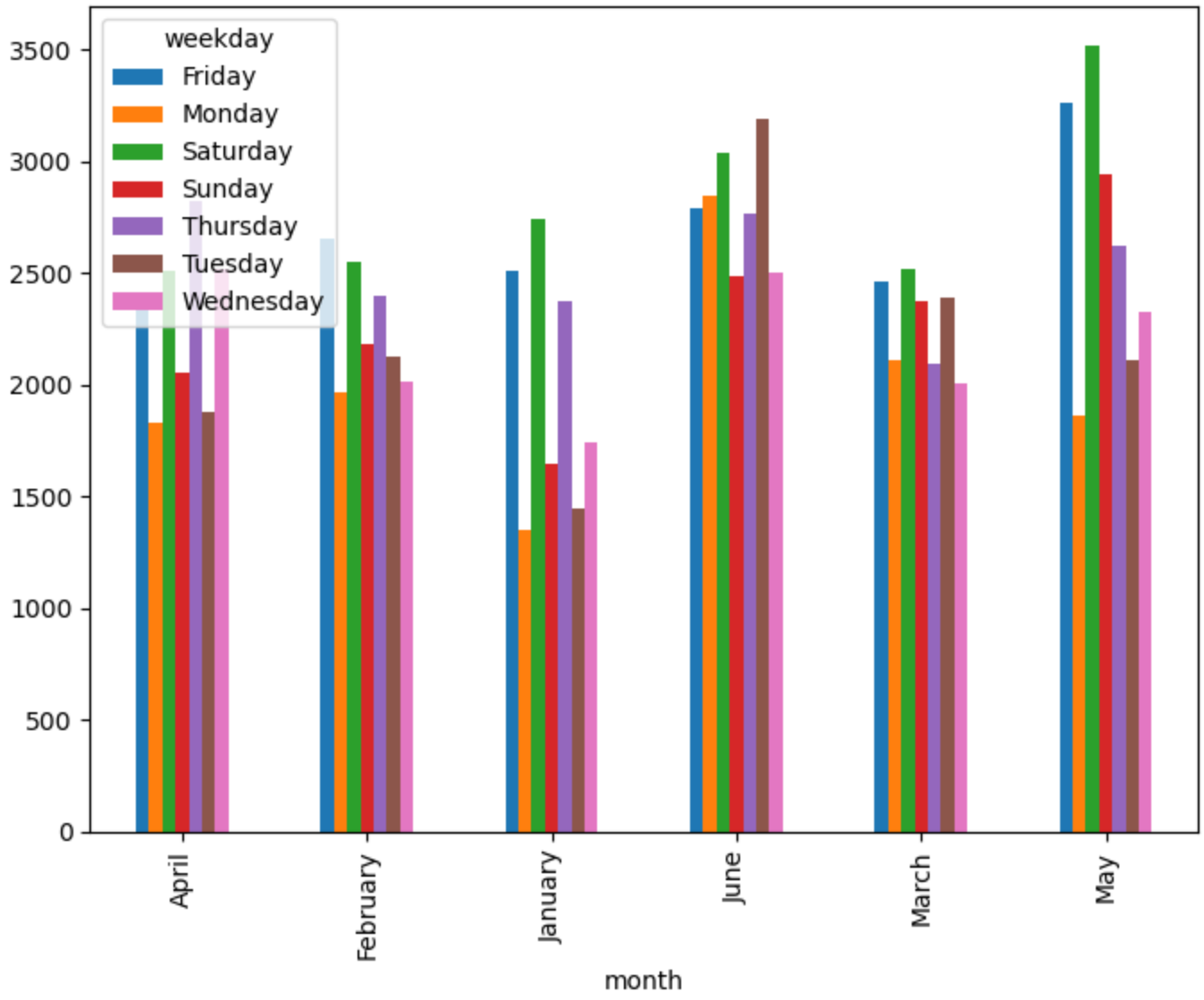
weekday	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday
month							
April	2365	1833	2508	2052	2823	1880	2521

February	2655	1970	2550	2183	2396	2129	2013
January	2508	1353	2745	1651	2378	1444	1740
June	2793	2848	3037	2485	2767	3187	2503
March	2465	2115	2522	2379	2093	2388	2007
May	3262	1865	3519	2944	2627	2115	2328

In [ ]:
```
#Grouped bar chart
```

In [49]:
```
pivot.plot(kind = 'bar', figsize = (8,6))
```

Out[49]:
```
<Axes: xlabel='month'>
```



In [ ]:
```
#Analyzing hourly peak
```

In [52]:
```
summary = uber_15.groupby(['weekday', 'hour'], as_index = False).size()
```

In [53]:
```
summary
```

Out[53]:

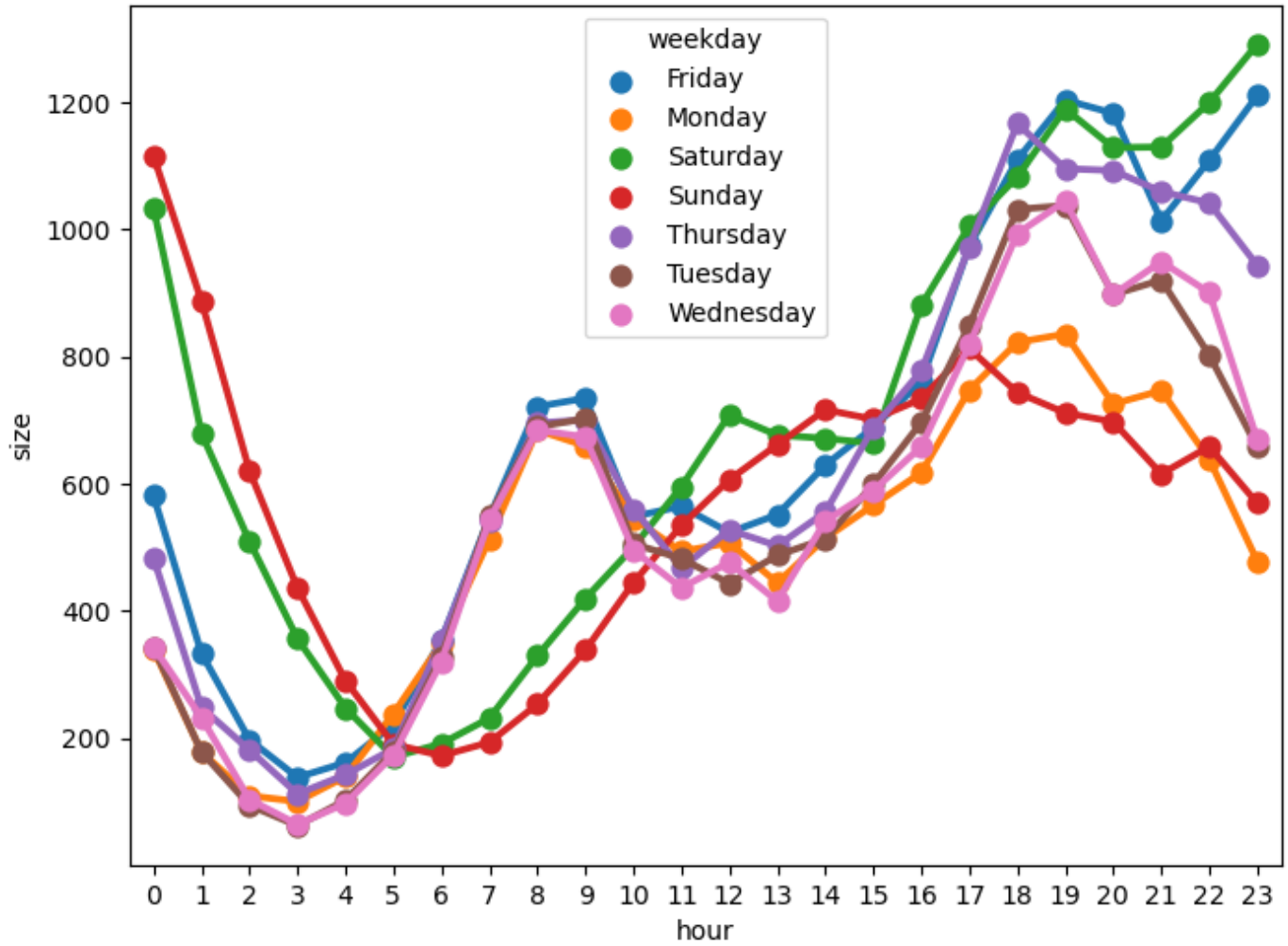
	weekday	hour	size
0	Friday	0	581
1	Friday	1	333

2	Friday	2	197
3	Friday	3	138
4	Friday	4	161
...	...	...	...
163	Wednesday	19	1044
164	Wednesday	20	897
165	Wednesday	21	949
166	Wednesday	22	900
167	Wednesday	23	669

168 rows × 3 columns

In [55]:
```
plt.figure(figsize = (8,6))
sns.pointplot(x= 'hour', y = 'size', hue = 'weekday', data = summary)
```

Out[55]:
```
<Axes: xlabel='hour', ylabel='size'>
```



In [ ]:
```
#find most active uber -based number with active vehicles
```

In [56]:
```
uber_15.columns
```

Out[56]:
```
Index(['Dispatching_base_num', 'Pickup_date', 'Affiliated_base_num',
       'locationID', 'month', 'weekday', 'day', 'hour', 'minute'],
      dtype='object')
```

In [5]:
```
os.listdir(r"C:\Users\usr\Downloads\uber-project\Uber\Datasets")
```

Out[5]:
```
['other-American_B01362.csv',
 'other-Carmel_B00256.csv',
 'other-Dial7_B00887.csv',
 'other-Diplo_B01196.csv',
 'other-Federal_02216.csv',
 'other-FHV-services_jan-aug-2015.csv',
 'other-Firstclass_B01536.csv',
 'other-Highclass_B01717.csv',
 'other-Lyft_B02510.csv',
 'other-Prestige_B01338.csv',
 'other-Skyline_B00111.csv',
 'Uber-Jan-Feb-FOIL.csv',
 'uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15.csv',
 'uber-raw-data-janjune-15_sample.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [60]:
```
uber_foil = pd.read_csv(r"C:\Users\usr\Downloads\uber-project\Uber\Datasets/Uber-Jan-Feb
```

In [62]:
```
uber_foil.shape
```

Out[62]:
```
(354, 4)
```

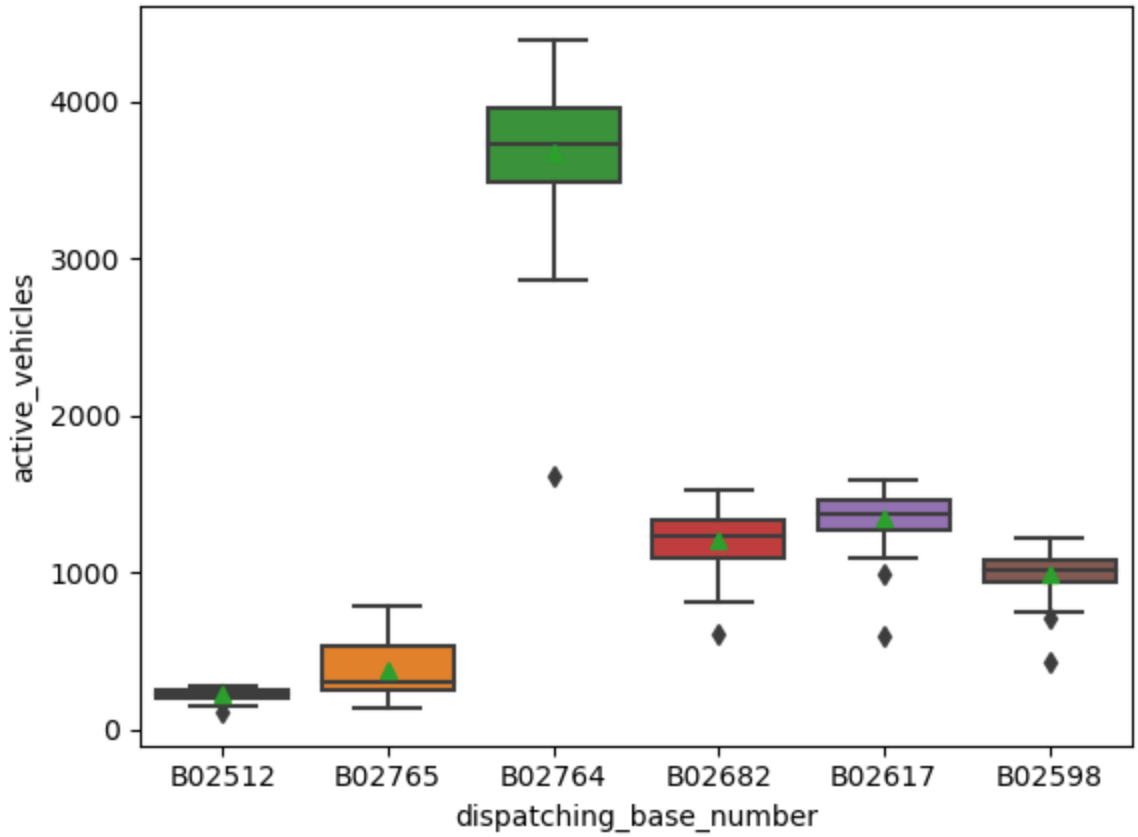
In [63]:
```
uber_foil.head(3)
```

Out[63]:

	dispatching_base_number	date	active_vehicles	trips
0	B02512	1/1/2015	190	1132
1	B02765	1/1/2015	225	1765
2	B02764	1/1/2015	3427	29421

In [83]:
```
sns.boxplot(data = uber_foil, x='dispatching_base_number', y='active_vehicles', showmean
```

Out[83]:
```
<Axes: xlabel='dispatching_base_number', ylabel='active_vehicles'>
```



In [ ]:

In [ ]:

In [ ]:
```
#collect entire data and make it ready for analysis
```

In [6]:
```
files = os.listdir(r"C:\Users\usr\Downloads\uber-project\Uber\Datasets")[-8:]
```

In [7]:
```
files
```

Out[7]:
```
['uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15.csv',
 'uber-raw-data-janjune-15_sample.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [8]:
```
files.remove('uber-raw-data-janjune-15.csv')
```

In [9]:
```
files
```

Out[9]:
```
['uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-janjune-15_sample.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [10]:
```
files.remove('uber-raw-data-janjune-15_sample.csv')
```

In [11]:
```
files
```

Out[11]:
```
['uber-raw-data-apr14.csv',
 'uber-raw-data-aug14.csv',
 'uber-raw-data-jul14.csv',
 'uber-raw-data-jun14.csv',
 'uber-raw-data-may14.csv',
 'uber-raw-data-sep14.csv']
```

In [16]:
```
final = pd.DataFrame()
path = r"C:\Users\usr\Downloads\uber-project\Uber\Datasets"
for file in files:
    current_df = pd.read_csv(path + '/'+ file)
    final = pd.concat([current_df, final])
```

In [17]:
```
final.shape
```

Out[17]:
```
(4534327, 4)
```

In [ ]:
```
#check for duplicates
```

In [19]:
```
final.duplicated().sum()
```

Out[19]:
```
82581
```

In [20]:
```
final.drop_duplicates(inplace=True)
```

In [21]:
```
final.shape
```

Out[21]:
```
(4451746, 4)
```

In [22]:
```
final.head(3)
```

Out[22]:

	Date/Time	Lat	Lon	Base
0	9/1/2014 0:01:00	40.2201	-74.0021	B02512
1	9/1/2014 0:01:00	40.7500	-74.0027	B02512
2	9/1/2014 0:03:00	40.7559	-73.9864	B02512

In [ ]:
```
#New york rush locations - SPACIAL ANALYSIS
```

In [25]:
```
rush_uber = final.groupby(['Lat', 'Lon'], as_index = False).size()
```

In [26]:
```
rush_uber.head(6)
```

Out[26]:

	Lat	Lon	size
0	39.6569	-74.2258	1
1	39.6686	-74.1607	1
2	39.7214	-74.2446	1
3	39.8416	-74.1512	1
4	39.9055	-74.0791	1
5	39.9196	-74.1112	1

In [27]:
```
!pip install folium
```

```
Defaulting to user installation because normal site-packages is not writeable
```

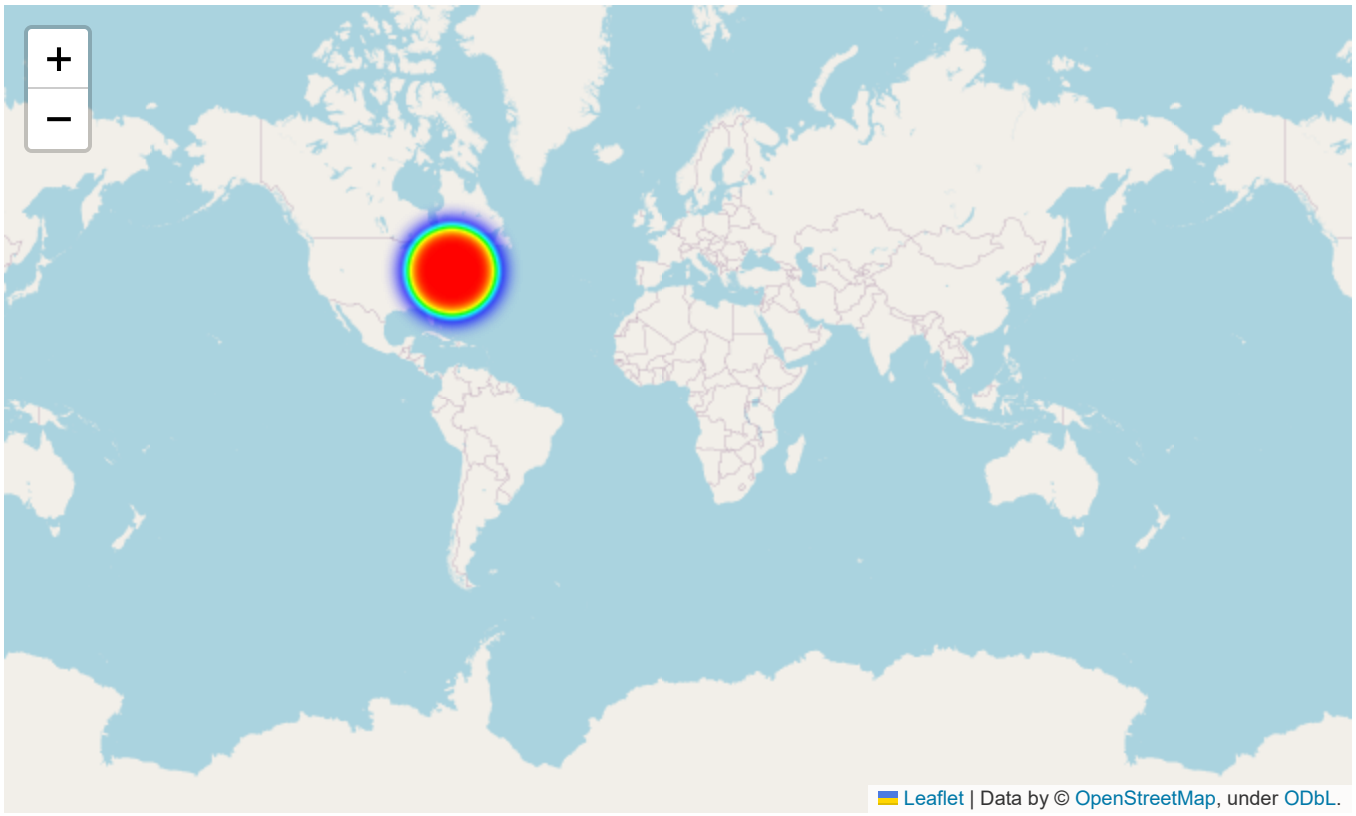
```
Collecting folium
  Downloading folium-0.14.0-py2.py3-none-any.whl (102 kB)
     ------------------------------------ 102.3/102.3 kB 1.2 MB/s eta 0:00:00
Requirement already satisfied: jinja2>=2.9 in c:\programdata\anaconda3\lib\site-packages
(from folium) (3.1.2)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from
folium) (1.23.5)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (f
rom folium) (2.28.1)
Collecting branca>=0.6.0
  Downloading branca-0.6.0-py3-none-any.whl (24 kB)
Requirement already satisfied: MarkupSafe>=2.0 in c:\programdata\anaconda3\lib\site-pack
ages (from jinja2>=2.9->folium) (2.1.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\anaconda3\lib\sit
e-packages (from requests->folium) (1.26.14)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-package
s (from requests->folium) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\programdata\anaconda3\lib
\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-p
ackages (from requests->folium) (2022.12.7)
Installing collected packages: branca, folium
Successfully installed branca-0.6.0 folium-0.14.0
```

In [28]:
```
import folium
```

In [29]:
```
basemap = folium.Map()
```

In [61]:
```
basemap
```

Out[61]:



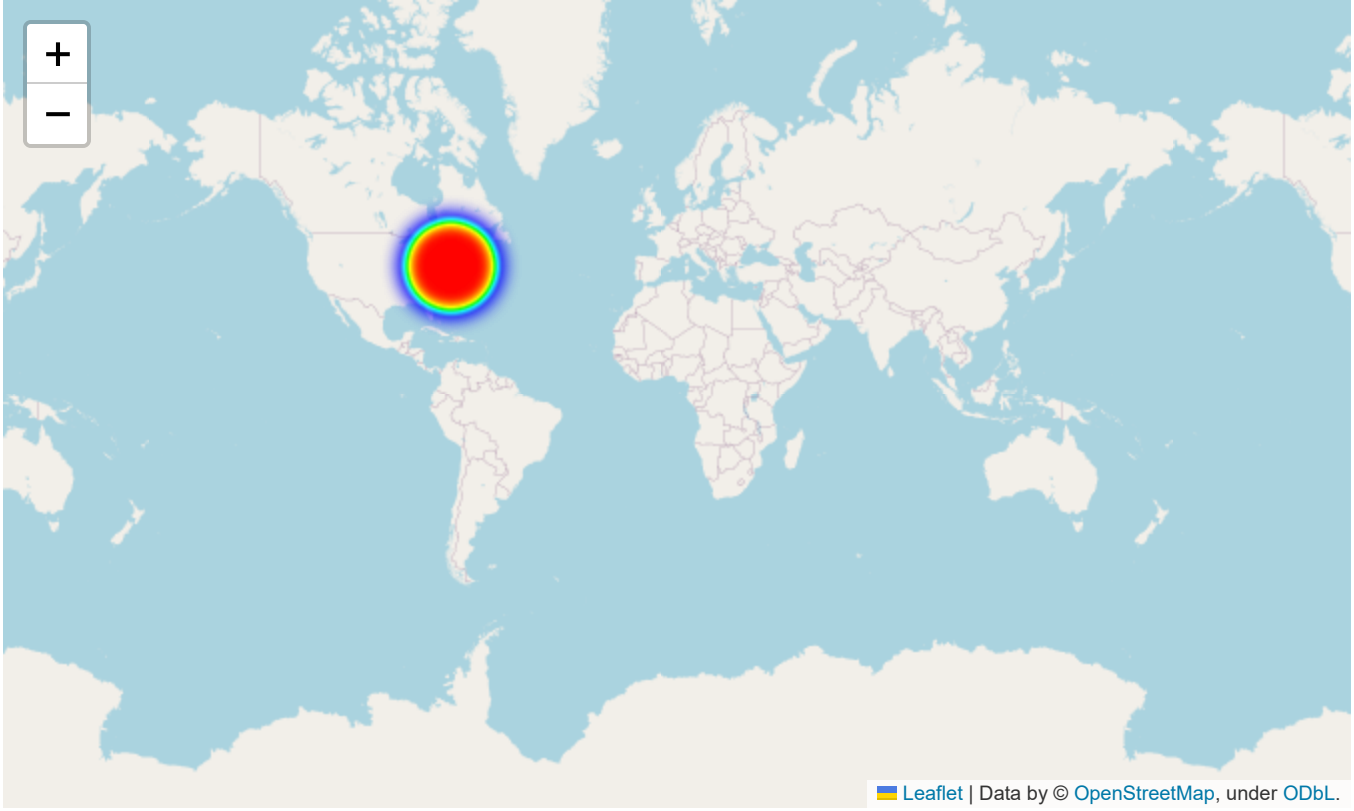
In [33]:
```
from folium.plugins import HeatMap
```

In [34]:
```
HeatMap(rush_uber).add_to(basemap)
```

Out[34]:
```
<folium.plugins.heat_map.HeatMap at 0x1fe0066a500>
```

In [35]:
```
basemap
```

Out[35]:



In [ ]:
```
#PAIR WISE ANALYSIS - Examine rush on Hour and Weekday
```

In [36]:
```
final.columns
```

Out[36]:
```
Index(['Date/Time', 'Lat', 'Lon', 'Base'], dtype='object')
```

In [37]:
```
final.head(3)
```

Out[37]:

	Date/Time	Lat	Lon	Base
0	9/1/2014 0:01:00	40.2201	-74.0021	B02512
1	9/1/2014 0:01:00	40.7500	-74.0027	B02512
2	9/1/2014 0:03:00	40.7559	-73.9864	B02512

In [39]:
```
final.dtypes
```

Out[39]:
```
Date/Time     object
Lat          float64
Lon          float64
Base          object
dtype: object
```

In [44]:
```
final['Date/Time'] = pd.to_datetime(final['Date/Time'], format = "%m/%d/%Y %H:%M:%S")
```

In [46]:
```
final['Date/Time'].dtype
```

Out[46]:
```
dtype('<M8[ns]')
```

In [47]:
```
final['day'] = final['Date/Time'].dt.day
```

In [48]:
```
final['hour'] = final['Date/Time'].dt.hour
```

In [49]:
```
final.head(4)
```

Out[49]:

	Date/Time	Lat	Lon	Base	day	hour
0	2014-09-01 00:01:00	40.2201	-74.0021	B02512	1	0
1	2014-09-01 00:01:00	40.7500	-74.0027	B02512	1	0
2	2014-09-01 00:03:00	40.7559	-73.9864	B02512	1	0
3	2014-09-01 00:06:00	40.7450	-73.9889	B02512	1	0

In [ ]:
```
#create a pivot table
```

In [50]:
```
pivot = final.groupby(['day', 'hour']).size().unstack()
```

In [51]:
```
pivot
```

Out[51]:

hour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19
day																	
1	3178	1944	1256	1308	1429	2126	3664	5380	5292	4617	...	6933	7910	8633	9511	8604	800
2	2435	1569	1087	1414	1876	2812	4920	6544	6310	4712	...	6904	8449	10109	11100	11123	947
3	3354	2142	1407	1467	1550	2387	4241	5663	5386	4657	...	7226	8850	10314	10491	11239	959
4	2897	1688	1199	1424	1696	2581	4592	6029	5704	4744	...	7158	8515	9492	10357	10259	909
5	2733	1541	1030	1253	1617	2900	4814	6261	6469	5530	...	6955	8312	9609	10699	10170	943
6	4537	2864	1864	1555	1551	2162	3642	4766	4942	4401	...	7235	8612	9444	9929	9263	840
7	3645	2296	1507	1597	1763	2422	4102	5575	5376	4639	...	7276	8474	10393	11013	10573	947
8	2830	1646	1123	1483	1889	3224	5431	7361	7357	5703	...	7240	8775	9851	10673	9687	879
9	2657	1724	1222	1480	1871	3168	5802	7592	7519	5895	...	7877	9220	10270	11910	11449	980
10	3296	2126	1464	1434	1591	2594	4664	6046	6158	5072	...	7612	9578	11045	11875	10934	961
11	3036	1665	1095	1424	1842	2520	4954	6876	6871	5396	...	7503	8920	10125	10898	10361	932
12	3227	2147	1393	1362	1757	2710	4576	6250	6231	5177	...	7743	9390	10734	11713	12216	1039
13	5408	3509	2262	1832	1705	2327	4196	5685	6060	5631	...	8200	9264	10534	11826	11450	992
14	3748	2349	1605	1656	1756	2629	4257	5781	5520	4824	...	6963	8192	9511	10115	9553	914
15	2497	1515	1087	1381	1862	2980	5050	6837	6729	5201	...	7633	8505	10285	11959	11728	1103
16	2547	1585	1119	1395	1818	2966	5558	7517	7495	5958	...	7597	9290	10804	11773	10855	1092
17	3155	2048	1500	1488	1897	2741	4562	6315	5882	4934	...	7472	8997	10323	11236	11089	991
18	3390	2135	1332	1626	1892	2959	4688	6618	6451	5377	...	7534	9040	10274	10692	10338	955
19	3217	2188	1604	1675	1810	2639	4733	6159	6014	5006	...	7374	8898	9893	10741	10429	970
20	4475	3190	2100	1858	1618	2143	3584	4900	5083	4765	...	7462	8630	9448	10046	9272	859
21	4294	3194	1972	1727	1926	2615	4185	5727	5529	4707	...	7064	8127	9483	9817	9291	831
22	2787	1637	1175	1468	1934	3151	5204	6872	6850	5198	...	7337	9148	10574	10962	9884	898
23	2546	1580	1136	1429	1957	3132	5204	6890	6436	5177	...	7575	9309	9980	10341	10823	1134
24	3200	2055	1438	1493	1798	2754	4484	6013	5913	5146	...	7083	8706	10366	10786	9772	908
25	2405	1499	1072	1439	1943	2973	5356	7627	7078	5994	...	7298	8732	9922	10504	10673	904
26	3810	3065	2046	1806	1730	2337	3776	5172	5071	4808	...	7269	8815	9885	10697	10867	1012

27	5196	3635	2352	2055	1723	2336	3539	4937	5053	4771	...	7519	8803	9793	9838	9228	8267
28	4123	2646	1843	1802	1883	2793	4290	5715	5671	5206	...	7341	8584	9671	9975	9132	8255
29	2678	1827	1409	1678	1948	3056	5213	6852	6695	5481	...	7630	9249	10105	11113	10411	9301
30	2401	1510	1112	1403	1841	3216	5757	7596	7611	6064	...	8396	10243	11554	12126	12561	11024
31	2174	1394	1087	919	773	997	1561	2169	2410	2525	...	4104	5099	5386	5308	5350	4898

31 rows × 24 columns

In [52]:
```
pivot.style.background_gradient()
```

Out[52]:

hour	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
day																	
1	3178	1944	1256	1308	1429	2126	3664	5380	5292	4617	4607	4729	4930	5794	6933	7910	8633
2	2435	1569	1087	1414	1876	2812	4920	6544	6310	4712	4797	4975	5188	5695	6904	8449	10109
3	3354	2142	1407	1467	1550	2387	4241	5663	5386	4657	4788	5065	5384	6093	7226	8850	10314
4	2897	1688	1199	1424	1696	2581	4592	6029	5704	4744	4743	4975	5193	6175	7158	8515	9492
5	2733	1541	1030	1253	1617	2900	4814	6261	6469	5530	5141	5011	5047	5690	6955	8312	9609
6	4537	2864	1864	1555	1551	2162	3642	4766	4942	4401	4801	5174	5426	6258	7235	8612	9444
7	3645	2296	1507	1597	1763	2422	4102	5575	5376	4639	4905	5166	5364	6214	7276	8474	10393
8	2830	1646	1123	1483	1889	3224	5431	7361	7357	5703	5288	5350	5483	6318	7240	8775	9851
9	2657	1724	1222	1480	1871	3168	5802	7592	7519	5895	5406	5443	5496	6419	7877	9220	10270
10	3296	2126	1464	1434	1591	2594	4664	6046	6158	5072	4976	5415	5506	6527	7612	9578	11045
11	3036	1665	1095	1424	1842	2520	4954	6876	6871	5396	5215	5423	5513	6486	7503	8920	10125
12	3227	2147	1393	1362	1757	2710	4576	6250	6231	5177	5157	5319	5570	6448	7743	9390	10734
13	5408	3509	2262	1832	1705	2327	4196	5685	6060	5631	5442	5720	5914	6678	8200	9264	10534
14	3748	2349	1605	1656	1756	2629	4257	5781	5520	4824	4911	5118	5153	5747	6963	8192	9511
15	2497	1515	1087	1381	1862	2980	5050	6837	6729	5201	5347	5517	5503	6997	7633	8505	10285
16	2547	1585	1119	1395	1818	2966	5558	7517	7495	5958	5626	5480	5525	6198	7597	9290	10804
17	3155	2048	1500	1488	1897	2741	4562	6315	5882	4934	5004	5306	5634	6507	7472	8997	10323
18	3390	2135	1332	1626	1892	2959	4688	6618	6451	5377	5150	5487	5490	6383	7534	9040	10274
19	3217	2188	1604	1675	1810	2639	4733	6159	6014	5006	5092	5240	5590	6367	7374	8898	9893
20	4475	3190	2100	1858	1618	2143	3584	4900	5083	4765	5135	5650	5745	6656	7462	8630	9448
21	4294	3194	1972	1727	1926	2615	4185	5727	5529	4707	4911	5212	5465	6085	7064	8127	9483
22	2787	1637	1175	1468	1934	3151	5204	6872	6850	5198	5277	5352	5512	6342	7337	9148	10574
23	2546	1580	1136	1429	1957	3132	5204	6890	6436	5177	5066	5304	5504	6232	7575	9309	9980
24	3200	2055	1438	1493	1798	2754	4484	6013	5913	5146	4947	5311	5229	5974	7083	8706	10366
25	2405	1499	1072	1439	1943	2973	5356	7627	7078	5994	5432	5504	5694	6204	7298	8732	9922
26	3810	3065	2046	1806	1730	2337	3776	5172	5071	4808	5061	5179	5381	6166	7269	8815	9885
27	5196	3635	2352	2055	1723	2336	3539	4937	5053	4771	5198	5732	5839	6820	7519	8803	9793

28	4123	2646	1843	1802	1883	2793	4290	5715	5671	5206	5247	5500	5486	6120	7341	8584	9671
29	2678	1827	1409	1678	1948	3056	5213	6852	6695	5481	5234	5163	5220	6305	7630	9249	10105
30	2401	1510	1112	1403	1841	3216	5757	7596	7611	6064	5987	6090	6423	7249	8396	10243	11554
31	2174	1394	1087	919	773	997	1561	2169	2410	2525	2564	2777	2954	3280	4104	5099	5386

In [59]:
```
def gen_pivot_table(df , col1 , col2):
    pivot = final.groupby([col1 , col2]).size().unstack()
    return pivot.style.background_gradient()
```

In [54]:
```
final.columns
```

Out[54]:
```
Index(['Date/Time', 'Lat', 'Lon', 'Base', 'day', 'hour'], dtype='object')
```

In [60]:
```
gen_pivot_table(final , "day" , "hour")
```

Out[60]:

hour	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
day																	
1	3178	1944	1256	1308	1429	2126	3664	5380	5292	4617	4607	4729	4930	5794	6933	7910	8633
2	2435	1569	1087	1414	1876	2812	4920	6544	6310	4712	4797	4975	5188	5695	6904	8449	10109
3	3354	2142	1407	1467	1550	2387	4241	5663	5386	4657	4788	5065	5384	6093	7226	8850	10314
4	2897	1688	1199	1424	1696	2581	4592	6029	5704	4744	4743	4975	5193	6175	7158	8515	9492
5	2733	1541	1030	1253	1617	2900	4814	6261	6469	5530	5141	5011	5047	5690	6955	8312	9609
6	4537	2864	1864	1555	1551	2162	3642	4766	4942	4401	4801	5174	5426	6258	7235	8612	9444
7	3645	2296	1507	1597	1763	2422	4102	5575	5376	4639	4905	5166	5364	6214	7276	8474	10393
8	2830	1646	1123	1483	1889	3224	5431	7361	7357	5703	5288	5350	5483	6318	7240	8775	9851
9	2657	1724	1222	1480	1871	3168	5802	7592	7519	5895	5406	5443	5496	6419	7877	9220	10270
10	3296	2126	1464	1434	1591	2594	4664	6046	6158	5072	4976	5415	5506	6527	7612	9578	11045
11	3036	1665	1095	1424	1842	2520	4954	6876	6871	5396	5215	5423	5513	6486	7503	8920	10125
12	3227	2147	1393	1362	1757	2710	4576	6250	6231	5177	5157	5319	5570	6448	7743	9390	10734
13	5408	3509	2262	1832	1705	2327	4196	5685	6060	5631	5442	5720	5914	6678	8200	9264	10534
14	3748	2349	1605	1656	1756	2629	4257	5781	5520	4824	4911	5118	5153	5747	6963	8192	9511
15	2497	1515	1087	1381	1862	2980	5050	6837	6729	5201	5347	5517	5503	6997	7633	8505	10285
16	2547	1585	1119	1395	1818	2966	5558	7517	7495	5958	5626	5480	5525	6198	7597	9290	10804
17	3155	2048	1500	1488	1897	2741	4562	6315	5882	4934	5004	5306	5634	6507	7472	8997	10323
18	3390	2135	1332	1626	1892	2959	4688	6618	6451	5377	5150	5487	5490	6383	7534	9040	10274
19	3217	2188	1604	1675	1810	2639	4733	6159	6014	5006	5092	5240	5590	6367	7374	8898	9893
20	4475	3190	2100	1858	1618	2143	3584	4900	5083	4765	5135	5650	5745	6656	7462	8630	9448
21	4294	3194	1972	1727	1926	2615	4185	5727	5529	4707	4911	5212	5465	6085	7064	8127	9483
22	2787	1637	1175	1468	1934	3151	5204	6872	6850	5198	5277	5352	5512	6342	7337	9148	10574
23	2546	1580	1136	1429	1957	3132	5204	6890	6436	5177	5066	5304	5504	6232	7575	9309	9980
24	3200	2055	1438	1493	1798	2754	4484	6013	5913	5146	4947	5311	5229	5974	7083	8706	10366

25	2405	1499	1072	1439	1943	2973	5356	7627	7078	5994	5432	5504	5694	6204	7298	8732	9922
26	3810	3065	2046	1806	1730	2337	3776	5172	5071	4808	5061	5179	5381	6166	7269	8815	9885
27	5196	3635	2352	2055	1723	2336	3539	4937	5053	4771	5198	5732	5839	6820	7519	8803	9793
28	4123	2646	1843	1802	1883	2793	4290	5715	5671	5206	5247	5500	5486	6120	7341	8584	9671
29	2678	1827	1409	1678	1948	3056	5213	6852	6695	5481	5234	5163	5220	6305	7630	9249	10105
30	2401	1510	1112	1403	1841	3216	5757	7596	7611	6064	5987	6090	6423	7249	8396	10243	11554
31	2174	1394	1087	919	773	997	1561	2169	2410	2525	2564	2777	2954	3280	4104	5099	5386

In [ ]: