Uber data Analysis

Analysis by Stanley Bankesie

stanleyetornam@gmail.com (mailto:stanleyetornam@gmail.com)

Uber is a ride sharing with their Headquaters in the United States of America.

Problem Statements

Dataset used for this analysis were obtain from uber from january 2016 to december 2016. dataset can be downloaded from from Kaggles using this link (http://www.kaggle.com/zusmani/uberdrives)

Questions that this analysis seek to solve are

1. Check how long do people travel with uber?
2. What hour do most people take uber to their destinations?
3. The purpose of trips
4. What day has the highest number of trips?
5. What are thew number of trips per day in a month?
6. The number of trips per month in a year?
7. The location with the highest number of start trips

First we import the necessary libraries that will be used in the Analysis

In [81]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
matplotlib.style.use("ggplot")
import seaborn as sns
import datetime
import calendar
```

Then we import our dataset

In [4]:

```
data=pd.read_csv("uber_2016.csv")
data
```

Out[4]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site
1155	Totals	NaN	NaN	NaN	NaN	12204.7	NaN

1156 rows × 7 columns

Checking for Missing Values in the dataset

```
In [5]: data.isnull()
```

Out[5]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	True
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
1151	False	False	False	False	False	False	False
1152	False	False	False	False	False	False	False
1153	False	False	False	False	False	False	False
1154	False	False	False	False	False	False	False
1155	False	True	True	True	True	False	True

1156 rows × 7 columns

Missing values were identified so the next step was to check which columns had the missing values

```
In [6]: data.isnull().any()
```

```
Out[6]: START_DATE*    False
        END_DATE*       True
        CATEGORY*       True
        START*          True
        STOP*           True
        MILES*         False
        PURPOSE*        True
        dtype: bool
```

Now checking the number of missing values in each column

```
In [7]: data.isnull().sum()
```

```
Out[7]: START_DATE*      0
        END_DATE*        1
        CATEGORY*        1
        START*           1
        STOP*            1
        MILES*           0
        PURPOSE*       503
        dtype: int64
```

Removing null/missing values from the dataset

Removing null/missing values from the dataset

In [11]:
```
data=data.dropna()
data
```

Out[11]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
5	1/6/2016 17:15	1/6/2016 17:19	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain
...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site

653 rows × 7 columns

Confirming the removal of missing values

In [12]:
```
data.isnull().sum()
```

Out[12]:
```
START_DATE*    0
END_DATE*      0
CATEGORY*      0
START*         0
STOP*          0
MILES*         0
PURPOSE*       0
dtype: int64
```

Checking the data type of each column in the dataset

In [13]:
```
data.dtypes
```

Out[13]:
```
START_DATE*     object
END_DATE*       object
CATEGORY*       object
START*          object
STOP*           object
MILES*         float64
PURPOSE*        object
dtype: object
```

Obtaining further Information of the dataset

In [15]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 653 entries, 0 to 1154
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   START_DATE*  653 non-null    object 
 1   END_DATE*    653 non-null    object 
 2   CATEGORY*    653 non-null    object 
 3   START*       653 non-null    object 
 4   STOP*        653 non-null    object 
 5   MILES*       653 non-null    float64
 6   PURPOSE*     653 non-null    object 
dtypes: float64(1), object(6)
memory usage: 40.8+ KB
```

START_DATE and END_DATE are date and are supposed to be in the datetime format but are in object format so these columns has to be converted to datetime format

In [19]:
```
data["START_DATE*"]=pd.to_datetime(data["START_DATE*"],format="%m/%d/%Y %H:%M")
data["END_DATE*"]=pd.to_datetime(data["END_DATE*"],format="%m/%d/%Y %H:%M")
```

Confirmation of datatype format convert

In [20]:
```
data.dtypes
```

Out[20]:
```
START_DATE*    datetime64[ns]
END_DATE*      datetime64[ns]
CATEGORY*              object
START*                 object
STOP*                  object
MILES*                float64
PURPOSE*               object
dtype: object
```

In [21]:
```
data.head()
```

Out[21]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
5	2016-01-06 17:15:00	2016-01-06 17:19:00	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain

Date and time are in one column so the format will be changed to seperate the date and time into different column

In [27]:
```
hour=[]
day=[]
dayofweek=[]
month=[]
weekday=[]

for x in data["START_DATE*"]:
    hour.append(x.hour)
    day.append(x.day)
    dayofweek.append(x.dayofweek)
    month.append(x.month)
    weekday.append(calendar.day_name[dayofweek[-1]])

data["HOUR"] = hour
data["DAY"] = day
data["DAY_OF_WEEK"] = dayofweek
data["MONTH"] = month
data["WEEKDAY"] = weekday
```

Confirmation of splitting date and time into seperate columns

In [28]:
```
data.head()
```

Out[28]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	HOUR	DAY
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	21	1
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	20	2
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting	17	5
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	14	6
5	2016-01-06 17:15:00	2016-01-06 17:19:00	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain	17	6

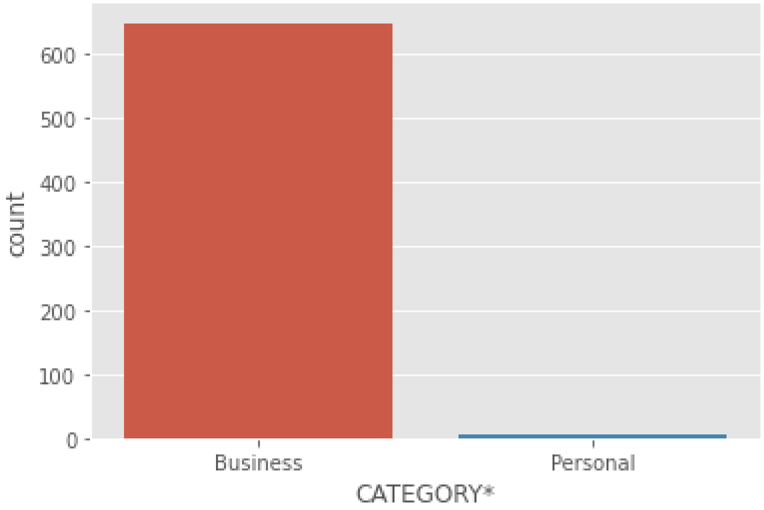
Category of trips

In [31]:
```
data["CATEGORY*"].value_counts()
```

Out[31]:
```
Business    647
Personal      6
Name: CATEGORY*, dtype: int64
```

In [76]:
```
sns.countplot(x="CATEGORY*",data=data)
```

Out[76]:
```
<AxesSubplot:xlabel='CATEGORY*', ylabel='count'>
```

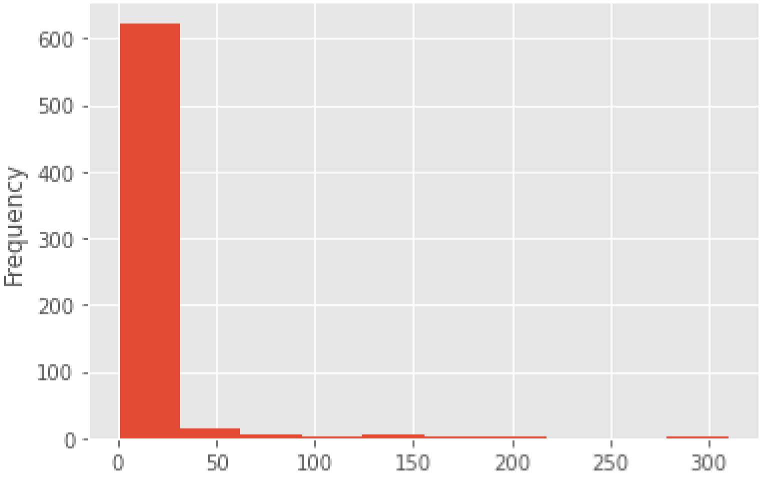


Distance (Miles) Being covered

People prefer using uber for shorter trips between 1 to 50 miles

```
In [77]: data["MILES*"].plot.hist()
```

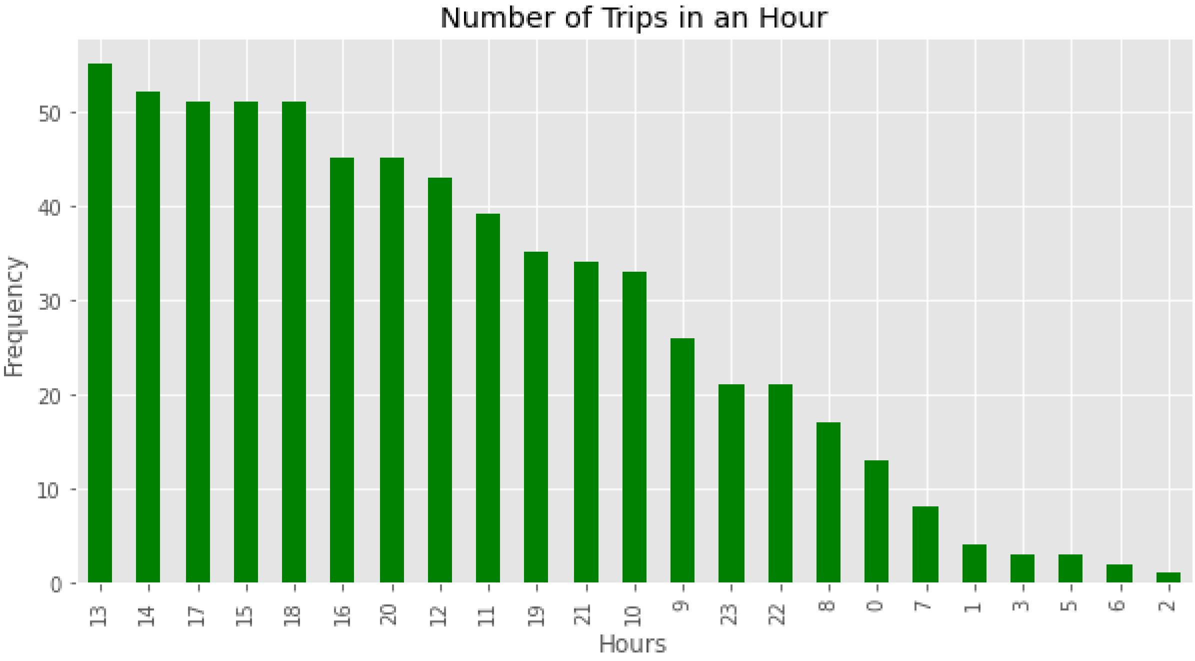
```
Out[77]: <AxesSubplot:ylabel='Frequency'>
```



Particular Hours people use uber the most

```
In [78]: hours = data["START_DATE*"].dt.hour.value_counts()
         hours.plot(kind ="bar",xlabel = "Hours",ylabel="Frequency",title = "Number of Tri
         #plt.xlabel("Hours")
         #plt.ylabel("Frequency")
         #plt.title("Number of Trips in an Hour")
```

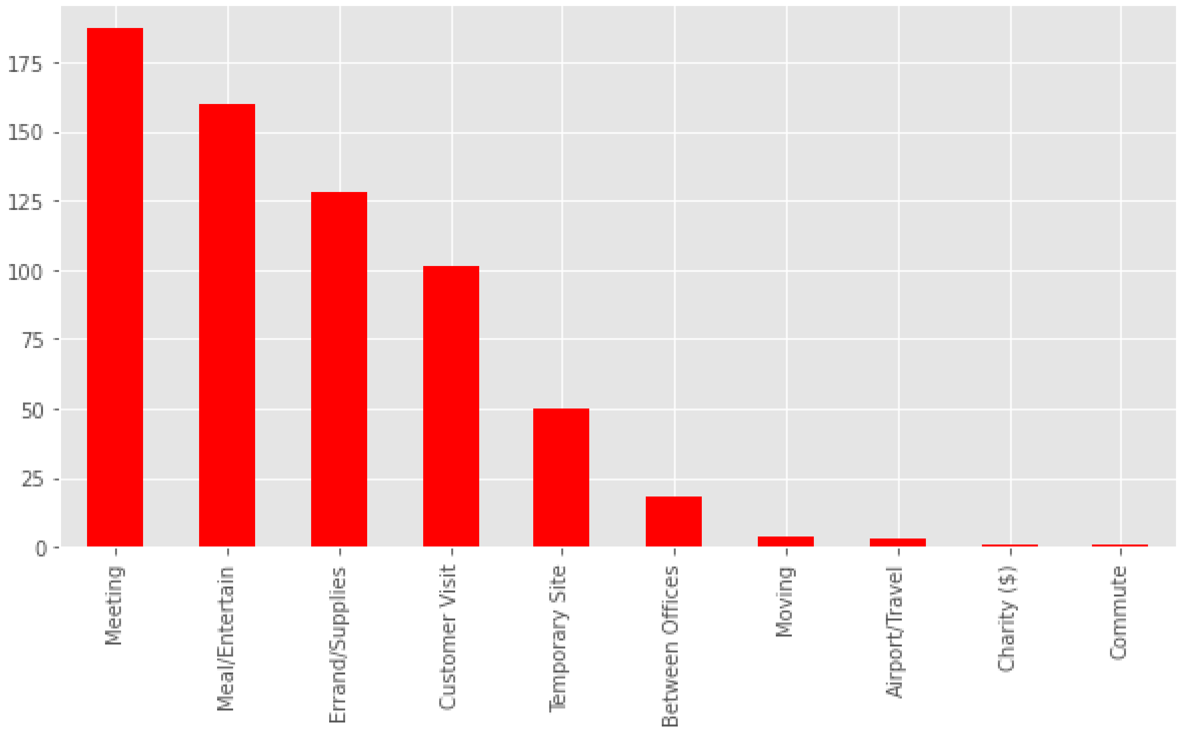
```
Out[78]: <AxesSubplot:title={'center':'Number of Trips in an Hour'}, xlabel='Hours', yla
         bel='Frequency'>
```



Purposes of Rides

```
In [79]: data["PURPOSE*"].value_counts().plot(kind="bar",figsize = (10,5), color = "red",)
```

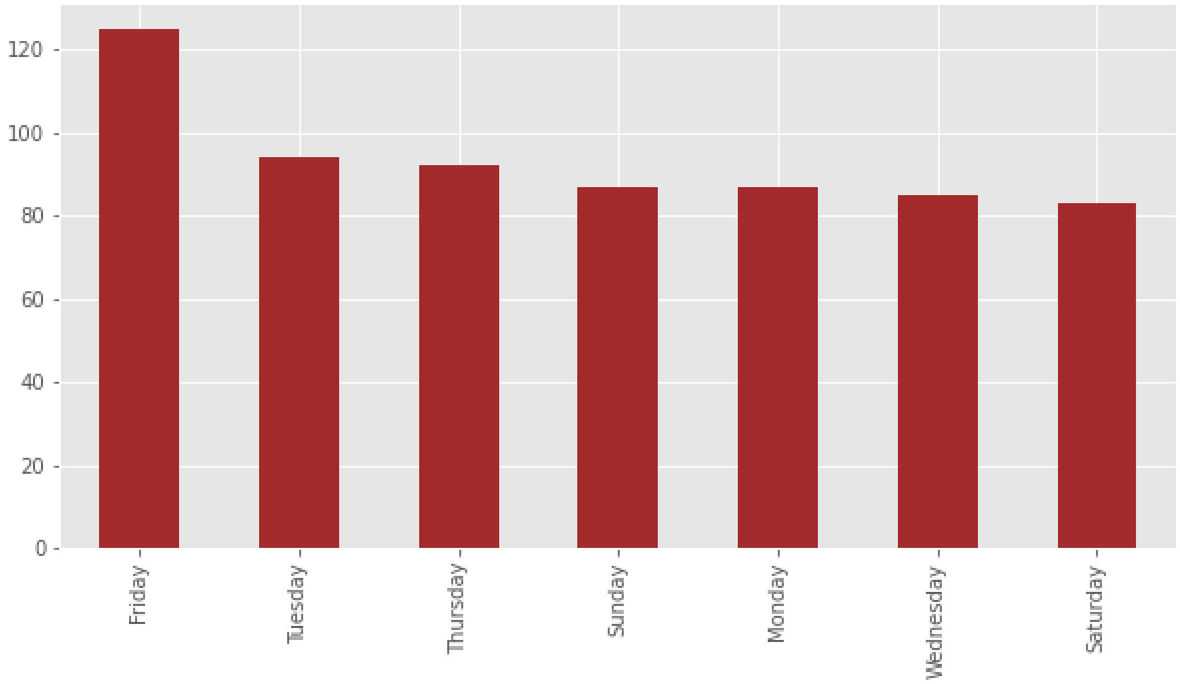
```
Out[79]: <AxesSubplot:>
```



#### days of the week with the highest number of trips

```
In [82]: data["WEEKDAY"].value_counts().plot(kind = "bar", figsize = (10,5), color= "brown
```

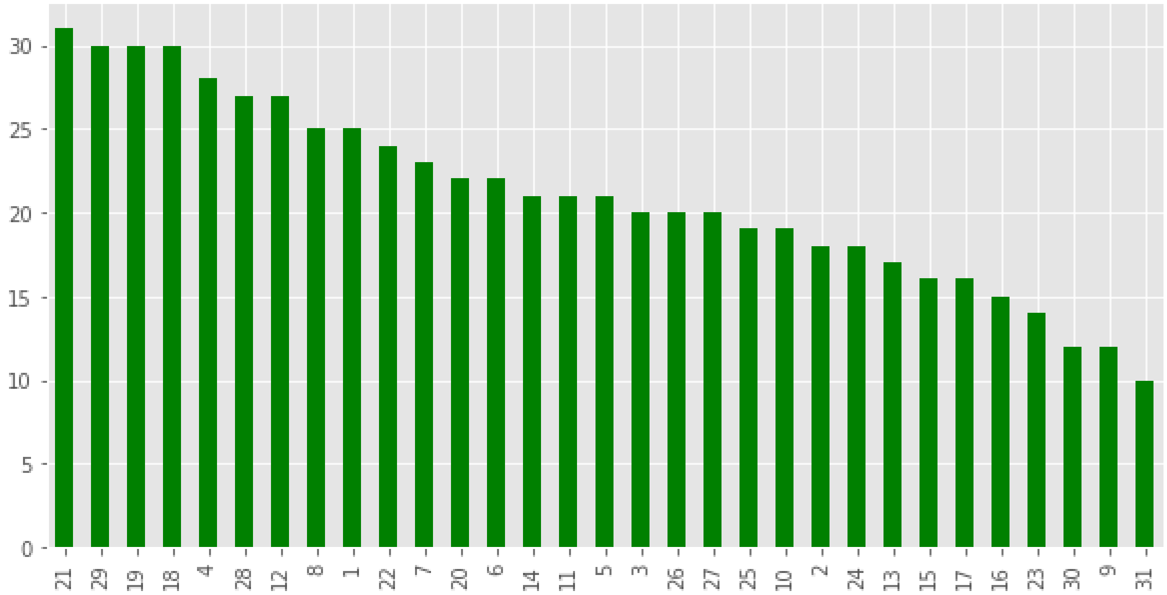
```
Out[82]: <AxesSubplot:>
```



Day with the highest number of trips per month

```
In [83]: data['DAY'].value_counts().plot(kind="bar",color="green",figsize=(10,5))
```

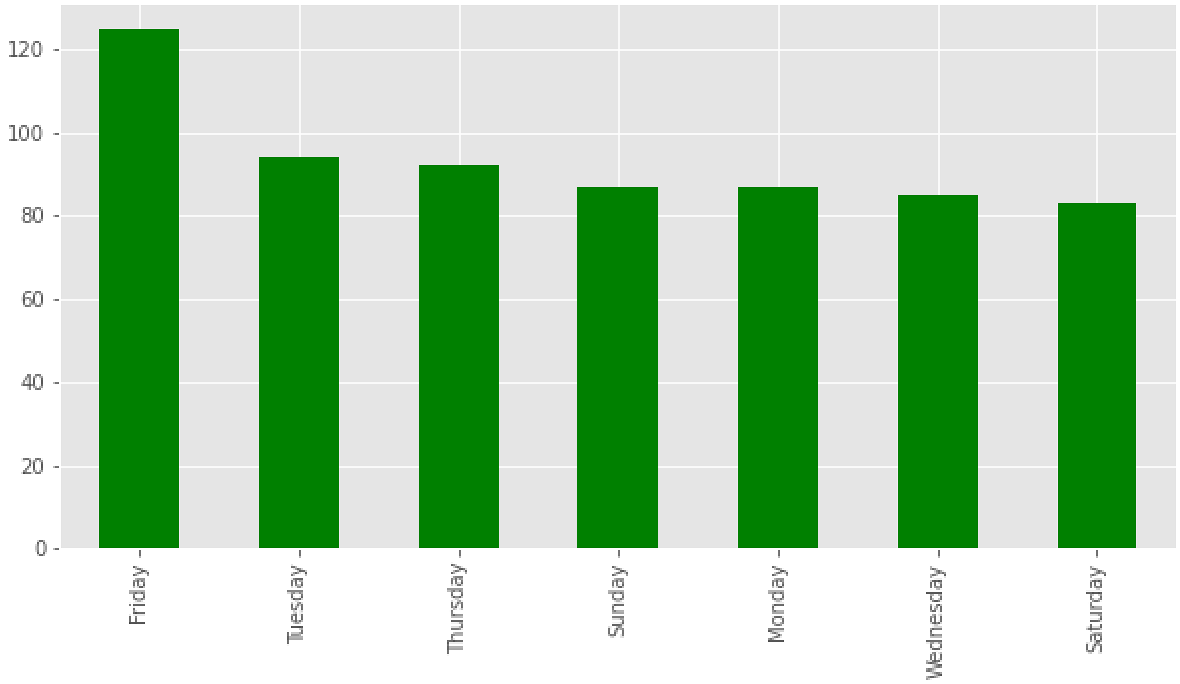
```
Out[83]: <AxesSubplot:>
```



```
In [ ]:
```

```
In [84]: data["WEEKDAY"].value_counts().plot(kind="bar",color="green",figsize=(10,5))
```

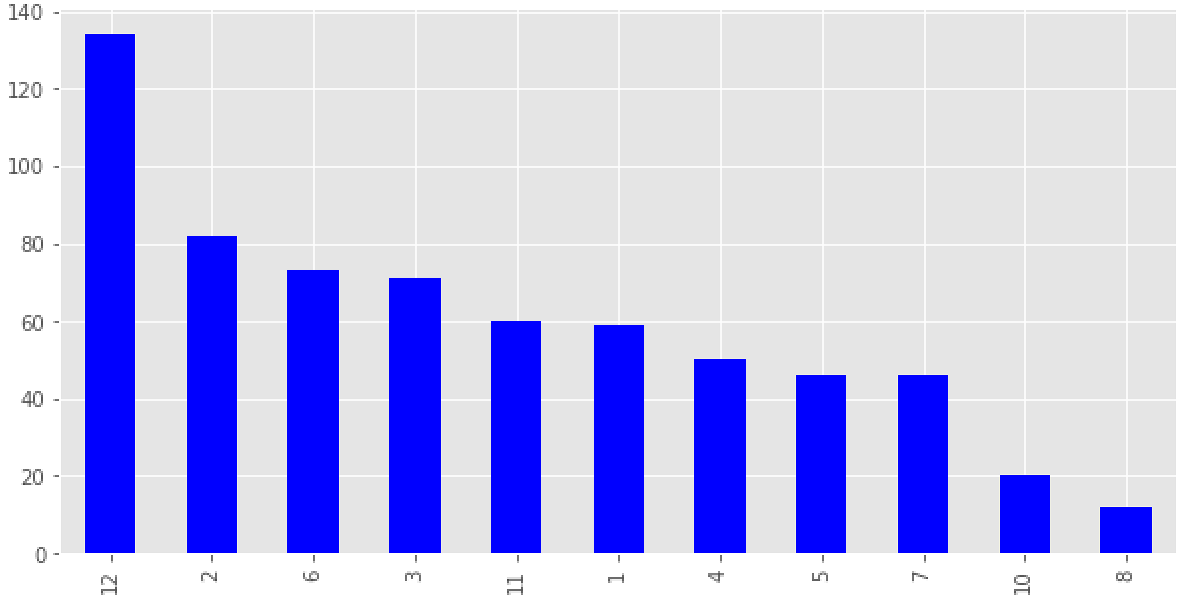
```
Out[84]: <AxesSubplot:>
```



Month with the highest number of trips in a year

```
In [85]: data["MONTH"].value_counts().plot(kind="bar", figsize=(10,5), color = "blue")
```

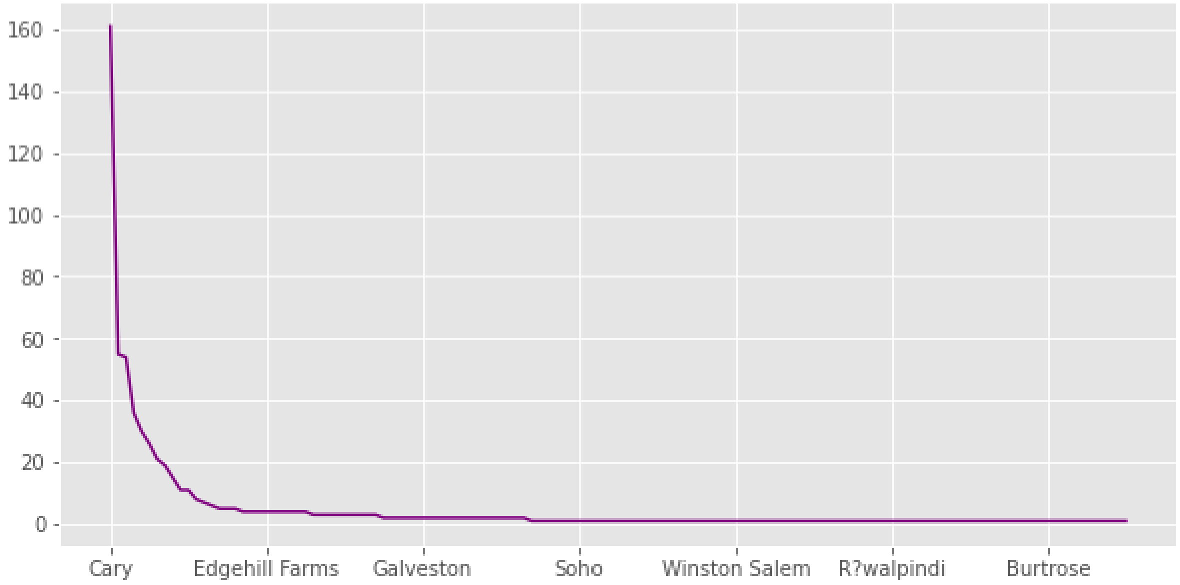
```
Out[85]: <AxesSubplot:>
```



location with the highest patronage

```
In [89]: data["START*"].value_counts().plot(kind="line",figsize=(10,5),color="purple")

Out[89]: <AxesSubplot:>
```



Conclusion

Insight derived from the Analysis above showed that

1. Riders prefer using uber for Business purposes than using it for Personal use
2. Riders prefer patronising for short distances (1 to 50Miles) as compared to distances further than the 50mile range.
3. High patronage of uber rides awere identified in the afternoon and in late afternoon between the hours of 13:00 and 20:00 while low patronage was identified midnights to dawn between the hours of 1:00 and 6:00.
4. Majority of riders use uber for their meetings followed by those who use uber Enternainment purposes while very few riders use uber rides for charity for commuting purposes.
5. uber rides are patronised largely on fridays than any other days of the week
6. December had the highest patronage per ride as compared to other months of the year.
7. Most trips were started in cary

Recommendation

1. Advertisement about uber rides for Personal purposes must be emphasized.
2. Short distance rides must be available for easy accessibilty for it has high potential of patronage in the future.
3. promotions and discouts should be rewarded to riders who patronises the midnight and dawn rides to increase patronage at that time of the day.
4. Enough vehicles must be available awaiting higher patronage on fridays
5. Promotions and discouts should the rewarded to riders who patronises uber in other months of the year apart from the december festive season and the new year
6. Advertidements should be done in other catchment areas as well