In [1]:

```
#importing pandas
import pandas as pd
#importing numpy
import numpy as np
#importing matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
import seaborn as sb
```

In [3]:

```
# importing city data using pandas
city = pd.read_csv('city.csv')
# importing global data using pandas
global_data = pd.read_csv('global.csv')
```

In [4]:

```
# printing first 5 rows of city dataframe
city.head()
```

Out[4]:

```
   year     city        country  avg_temp
0  1849  Abidjan  Côte D'Ivoire     25.58
1  1850  Abidjan  Côte D'Ivoire     25.52
2  1851  Abidjan  Côte D'Ivoire     25.67
3  1852  Abidjan  Côte D'Ivoire       NaN
4  1853  Abidjan  Côte D'Ivoire       NaN
```

In [5]:

```
# printing first 5 rows of global dataframe
global_data.head()
```

Out[5]:

```
   year  avg_temp
0  1750      8.72
1  1751      7.98
2  1752      5.78
3  1753      8.39
4  1754      8.47
```

In [6]:

```
# describing global_data
global_data.describe()
```

Out[6]:

	year	avg_temp
count	266.000000	266.000000
mean	1882.500000	8.369474
std	76.931788	0.584747
min	1750.000000	5.780000
25%	1816.250000	8.082500
50%	1882.500000	8.375000
75%	1948.750000	8.707500
max	2015.000000	9.830000

In [7]:

```
# checking columns and rows in city dataframe before cleaning dataframe
city.shape
```

Out[7]:

```
(70792, 4)
```

In [8]:

```
# dropping empty values cleaning data is highly important
city = city.dropna()
```

In [67]:

```
# checking columns and rows in city dataframe after cleaning
city.shape
```

Out[67]:

```
(68245, 4)
```

In [74]:

```
a =[ None, None, None, None, None, None]
for i in range(6, global_data.shape[0]):
    b = 0
    for j in range(7):
        b = b + global_data.iloc[i-j,1:2][0]
    a.append(b / 7)
a = np.array(a)
# print(a)
global_data['moving average'] = a
```

```
[None None None None None None 8.078571428571427 8.12142857142857
 7.944285714285714 8.26 8.088571428571429 8.131428571428573
 8.167142857142858 7.974285714285715 7.885714285714286 8.10142857142
8572
 8.161428571428571 8.308571428571428 8.024285714285714 7.89285714285
7143
 7.920000000000001 7.841428571428572 7.832857142857143 7.80571428571
4286
 7.884285714285714 8.227142857142857 8.314285714285713 8.39571428571
4286
 8.494285714285713 8.607142857142858 8.780000000000001 8.68428571428
5714
 8.501428571428571 8.412857142857144 8.355714285714287 8.18714285714
2858
 8.084285714285715 7.884285714285714 7.934285714285713 7.99571428571
42856
 8.038571428571428 8.09142857142857 8.195714285714285 8.191428571428
572
 8.262857142857143 8.248571428571429 8.24 8.315714285714288
 8.37857142857143 8.438571428571427 8.474285714285715 8.482857142857
144
 8.515714285714285 8.548571428571428 8.595714285714287 8.58
 8.56857142857143 8.540000000000001 8.402857142857142 8.188571428571
43
 7.962857142857142 7.68 7.464285714285714 7.365714285714286
 7.267142857142857 7.211428571428572 7.191428571428571 7.2
 7.338571428571428 7.384285714285715 7.367142857142858 7.43857142857
1429
 7.574285714285714 7.685714285714285 7.909999999999998 7.98999999999
9999
 8.13142857142857 8.301428571428572 8.312857142857142 8.277142857142
858
 8.391428571428573 8.261428571428572 8.127142857142857 8.07714285714
2858
 7.982857142857142 7.871428571428571 7.837142857142857 7.67428571428
5714
 7.655714285714287 7.681428571428571 7.65142857142857 7.585714285714
287
 7.675714285714286 7.742857142857143 7.781428571428571 7.83
 7.96142857142857 8.002857142857142 8.044285714285715 8.038571428571
428
 8.0 8.075714285714286 8.111428571428572 8.03857142857143
 8.055714285714286 8.074285714285713 8.077142857142857 8.05714285714
2857
 8.045714285714286 8.067142857142857 8.055714285714286 8.00428571428
5713
 7.925714285714285 7.941428571428571 7.972857142857143 7.98428571428
5714
 7.990000000000001 8.05857142857143 8.115714285714287 8.239999999999
998
 8.252857142857142 8.272857142857143 8.274285714285714 8.28285714285
7143
 8.28142857142857 8.225714285714284 8.175714285714283 8.224285714285
713
 8.325714285714286 8.322857142857142 8.29 8.267142857142856
 8.305714285714284 8.291428571428572 8.181428571428572 8.05142857142
8572
 8.02 7.990000000000001 7.964285714285715 7.991428571428573
 7.990000000000001 8.025714285714287 8.047142857142857 8.06285714285
7142
 8.098571428571429 8.107142857142858 8.09142857142857 8.137142857142
859
 8.16 8.207142857142857 8.27 8.324285714285713 8.345714285714285
 8.347142857142858 8.31857142857143 8.325714285714286 8.322857142857
142
 8.244285714285715 8.194285714285714 8.177142857142856 8.17714285714
2858
 8.19 8.181428571428572 8.17 8.26142857142857 8.318571428571428
 8.325714285714286 8.297142857142857 8.290000000000001 8.32000000000
0002
 8.328571428571431 8.325714285714287 8.3 8.327142857142858
 8.397142857142857 8.454285714285716 8.504285714285713 8.52714285714
2858
 8.535714285714286 8.511428571428572 8.541428571428572 8.57142857142
8573
 8.597142857142858 8.541428571428572 8.557142857142859 8.54142857142
8572
 8.585714285714287 8.595714285714285 8.615714285714285 8.62285714285
7143
 8.682857142857141 8.702857142857141 8.732857142857142 8.76285714285
7143
 8.784285714285714 8.744285714285713 8.73285714285714 8.738571428571
43
 8.735714285714286 8.715714285714286 8.66 8.628571428571428
 8.637142857142857 8.664285714285713 8.629999999999999 8.61285714285
7143
 8.568571428571428 8.620000000000001 8.64 8.652857142857142
 8.611428571428572 8.645714285714286 8.662857142857144 8.74571428571
4286
 8.7 8.665714285714285 8.647142857142857 8.664285714285713
 8.624285714285715 8.602857142857143 8.58 8.607142857142858
 8.602857142857143 8.652857142857144 8.620000000000001 8.65142857142
8571
 8.615714285714287 8.637142857142857 8.65 8.682857142857143
 8.687142857142858 8.787142857142857 8.772857142857143 8.87000000000
0001
 8.847142857142858 8.842857142857143 8.857142857142858 8.85857142857
1429
 8.862857142857141 8.902857142857142 8.93142857142857 9.001428571428
573
 9.027142857142858 9.032857142857145 9.040000000000001 9.06142857142
857
 9.078571428571427 9.074285714285713 9.122857142857143 9.18714285714
2858
 9.234285714285713 9.287142857142857 9.318571428571428 9.38857142857
1428
 9.405714285714286 9.431428571428572 9.465714285714286 9.54142857142
8572
 9.544285714285715 9.535714285714283 9.560000000000002 9.58857142857
1429
 9.561428571428573 9.572857142857142 9.549999999999999 9.60714285714
2858]
```

In [75]:

```
global_data.head()
```

Out[75]:

	year	avg_temp	moving average
0	1750	8.72	None
1	1751	7.98	None
2	1752	5.78	None
3	1753	8.39	None
4	1754	8.47	None

In [60]:

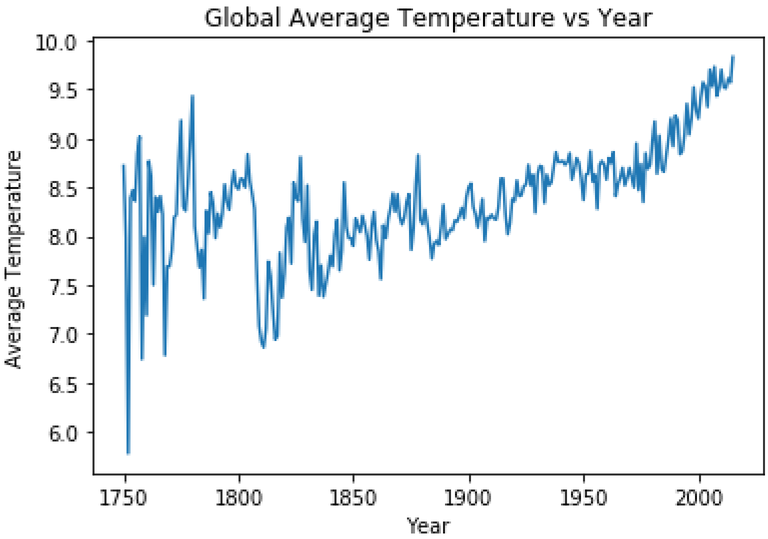
```
global_data.iloc[1,1:2][0]
```

Out[60]:

```
7.98
```

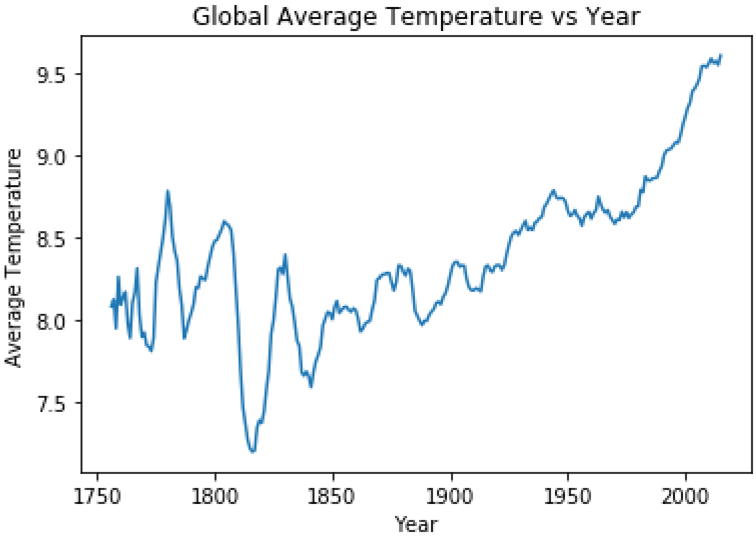
In [76]:

```
# line plot of global avg temp vs year
plt.plot(global_data['year'],global_data['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Global Average Temperature vs Year')
plt.plot();
```



In [80]:

```
# line plot of global moving avg temp vs year
plt.plot(global_data['year'],global_data['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Global Average Temperature vs Year')
plt.plot();
```

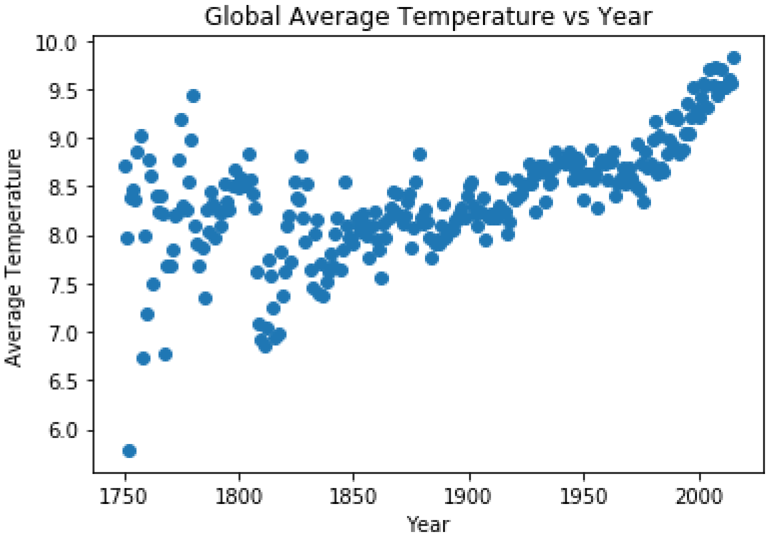


Moving average makes the graph more consistent

Average Temperature increases over the years and it is maybe a reason due to global warming

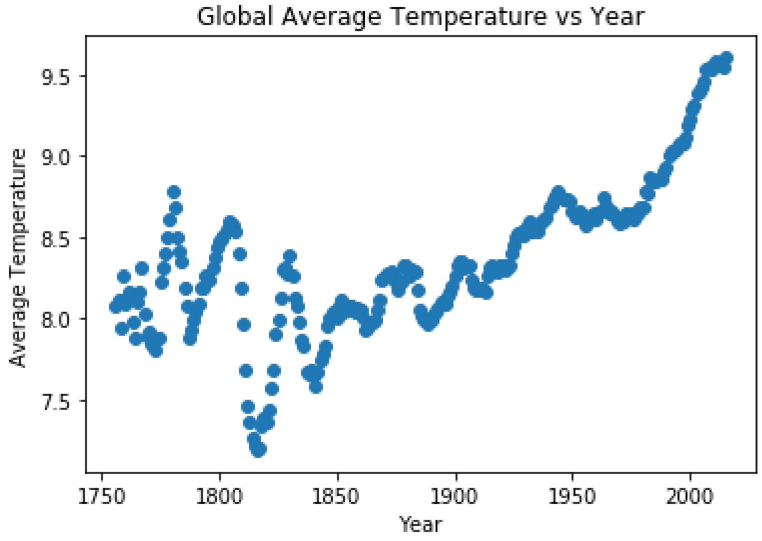
In [78]:

```
#Scatter plot of global avg temp vs year
plt.scatter(global_data['year'],global_data['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Global Average Temperature vs Year')
plt.plot();
```



In [79]:

```
#Scatter plot of global avg temp vs year
plt.scatter(global_data['year'],global_data['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Global Average Temperature vs Year')
plt.plot();
```



The average temperature becomes consistent over the years

In [12]:

```
# displaying some rows of city dataframe
city.head()
```

Out[12]:

	year	city	country	avg_temp
0	1849	Abidjan	Côte D'Ivoire	25.58
1	1850	Abidjan	Côte D'Ivoire	25.52
2	1851	Abidjan	Côte D'Ivoire	25.67
7	1856	Abidjan	Côte D'Ivoire	26.28
8	1857	Abidjan	Côte D'Ivoire	25.17

In [13]:

```
# searching for my city
city['city'].unique()
```

Out[13]:

```
array(['Abidjan', 'Abu Dhabi', 'Abuja', 'Accra', 'Adana', 'Adelaid
e',
       'Agra', 'Ahmadabad', 'Albuquerque', 'Alexandria', 'Algiers',
       'Allahabad', 'Almaty', 'Amritsar', 'Amsterdam', 'Ankara', 'An
shan',
       'Antananarivo', 'Arlington', 'Asmara', 'Astana', 'Athens',
       'Atlanta', 'Austin', 'Baghdad', 'Baku', 'Baltimore', 'Bamak
o',
       'Bandung', 'Bangalore', 'Bangkok', 'Bangui', 'Barcelona',
       'Barquisimeto', 'Barranquilla', 'Beirut', 'Belfast', 'Belgrad
e',
       'Belo Horizonte', 'Benghazi', 'Berlin', 'Bern', 'Bhopal',
       'Birmingham', 'Bissau', 'Boston', 'Bratislava', 'Brazzavill
e',
       'Brisbane', 'Brussels', 'Bucharest', 'Budapest', 'Bujumbura',
       'Bursa', 'Cairo', 'Cali', 'Campinas', 'Canberra', 'Caracas',
       'Cardiff', 'Casablanca', 'Changchun', 'Changzhou', 'Charlott
e',
       'Chelyabinsk', 'Chengdu', 'Chicago', 'Chisinau', 'Colombo',
       'Colorado Springs', 'Columbus', 'Conakry', 'Copenhagen', 'Cor
doba',
       'Curitiba', 'Dakar', 'Dalian', 'Dallas', 'Damascus',
       'Dar Es Salaam', 'Datong', 'Delhi', 'Denver', 'Detroit', 'Dha
ka',
       'Doha', 'Douala', 'Dublin', 'Durban', 'Dushanbe', 'Ecatepec',
       'Edinburgh', 'El Paso', 'Faisalabad', 'Fort Worth', 'Fortalez
a',
       'Foshan', 'Freetown', 'Fresno', 'Fuzhou', 'Gaborone', 'George
town',
       'Guadalajara', 'Guangzhou', 'Guarulhos', 'Guatemala City',
       'Guayaquil', 'Guiyang', 'Gujranwala', 'Hamburg', 'Handan',
       'Hangzhou', 'Hanoi', 'Haora', 'Harare', 'Harbin', 'Hefei',
       'Helsinki', 'Hiroshima', 'Ho Chi Minh City', 'Houston',
       'Hyderabad', 'Ibadan', 'Indianapolis', 'Indore', 'Irbil',
       'Islamabad', 'Istanbul', 'Izmir', 'Jacksonville', 'Jaipur',
       'Jakarta', 'Jilin', 'Jinan', 'Johannesburg', 'Juba', 'Kabul',
       'Kaduna', 'Kampala', 'Kano', 'Kanpur', 'Kansas City', 'Karach
i',
       'Kathmandu', 'Kawasaki', 'Kazan', 'Khartoum', 'Khulna', 'Kie
v',
       'Kigali', 'Kingston', 'Kinshasa', 'Kitakyushu', 'Kobe',
       'Kuala Lumpur', 'Kunming', 'La Paz', 'Lagos', 'Lahore', 'Lanz
hou',
       'Las Vegas', 'Libreville', 'Lilongwe', 'Lima', 'Lisbon',
       'Ljubljana', 'London', 'Long Beach', 'Los Angeles', 'Louisvil
le',
       'Luanda', 'Lubumbashi', 'Ludhiana', 'Luoyang', 'Lusaka', 'Mad
rid',
       'Maiduguri', 'Malabo', 'Managua', 'Manama', 'Manaus', 'Manil
a',
       'Maputo', 'Maracaibo', 'Maseru', 'Mashhad', 'Mecca', 'Medan',
       'Melbourne', 'Memphis', 'Mesa', 'Mexicali', 'Miami', 'Milan',
       'Milwaukee', 'Minneapolis', 'Minsk', 'Mogadishu', 'Monrovia',
       'Monterrey', 'Montevideo', 'Montreal', 'Moscow', 'Multan',
       'Munich', 'Nagoya', 'Nagpur', 'Nairobi', 'Nanchang', 'Nanjin
g',
       'Nanning', 'Nashville', 'Nassau', 'New Delhi', 'New Orleans',
       'New York', 'Niamey', 'Nouakchott', 'Novosibirsk', 'Oakland',
       'Oklahoma City', 'Omaha', 'Omsk', 'Oslo', 'Ottawa', 'Ouagadou
gou',
       'Palembang', 'Paramaribo', 'Paris', 'Patna', 'Perm', 'Perth',
       'Peshawar', 'Philadelphia', 'Phoenix', 'Podgorica',
       'Port Au Prince', 'Port Harcourt', 'Port Louis', 'Port Moresb
y',
       'Portland', 'Porto Alegre', 'Prague', 'Pretoria', 'Pristina',
       'Puebla', 'Pune', 'Qingdao', 'Qiqihar', 'Quito', 'Rabat', 'Ra
jkot',
       'Raleigh', 'Ranchi', 'Rawalpindi', 'Recife', 'Riga',
       'Rio De Janeiro', 'Riyadh', 'Rome', 'Rosario', 'Sacramento',
       'Salvador', 'Samara', 'San Antonio', 'San Diego', 'San Franci
sco',
       'San Jose', 'San Salvador', 'Santa Cruz', 'Santiago',
       'Santo Domingo', 'Sarajevo', 'Seattle', 'Semarang', 'Seoul',
       'Shanghai', 'Shenyang', 'Shenzhen', 'Shiraz', 'Singapore',
       'Skopje', 'Sofia', 'Soweto', 'Stockholm', 'Surabaya', 'Sura
t',
       'Suzhou', 'Sydney', 'Tabriz', 'Taiyuan', 'Tallinn', 'Tangsha
n',
       'Tashkent', 'Tbilisi', 'Tegucigalpa', 'Tianjin', 'Tijuana',
       'Tirana', 'Tokyo', 'Toronto', 'Tripoli', 'Tucson', 'Tulsa',
       'Tunis', 'Ufa', 'Ulaanbaatar', 'Vadodara', 'Valencia', 'Varan
asi',
       'Victoria', 'Vienna', 'Vientiane', 'Vilnius', 'Virginia Beac
h',
       'Volgograd', 'Warsaw', 'Washington', 'Wellington', 'Wichita',
       'Windhoek', 'Wuhan', 'Wuxi', 'Xian', 'Xuzhou', 'Yamoussoukr
o',
       'Yerevan', 'Zagreb', 'Zapopan'], dtype=object)
```

In [14]:

```
# taking my city rows from the main dataframe
# Delhi is where i live
my_city = city[city['city'] == 'Delhi']
```

In [83]:

```
# mycity dataframe
my_city
```

Out[83]:

	year	city	country	avg_temp	moving average
18444	1796	Delhi	India	25.03	None
18445	1797	Delhi	India	26.71	None
18446	1798	Delhi	India	24.29	None
18447	1799	Delhi	India	25.28	None
18448	1800	Delhi	India	25.21	None
...	...	...	...	...	...
18657	2009	Delhi	India	26.55	26.0614
18658	2010	Delhi	India	26.52	26.1757
18659	2011	Delhi	India	25.63	26.0886
18660	2012	Delhi	India	25.89	26.1129
18661	2013	Delhi	India	26.71	26.1614

201 rows × 5 columns

In [82]:

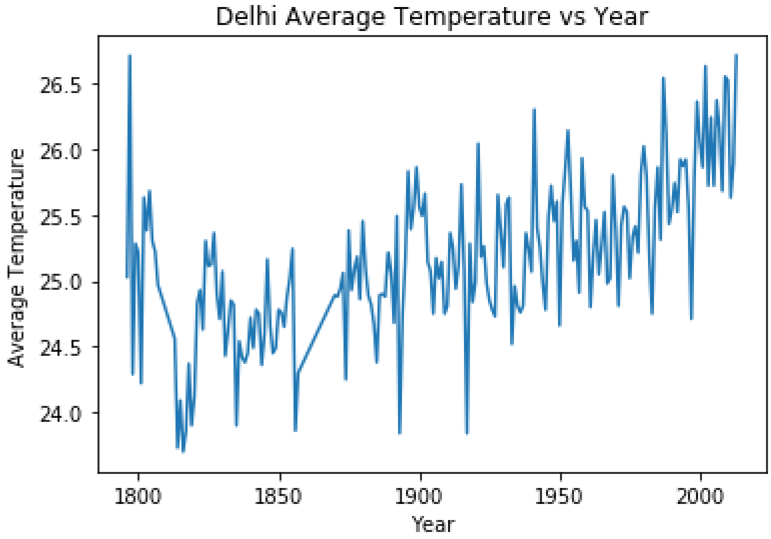
```
a =[ None, None, None, None, None, None]
for i in range(6, my_city.shape[0]):
    b = 0
    for j in range(7):
        b = b + my_city.iloc[i-j,2:3][0]
    a.append(b / 7)
a = np.array(a)
my_city['moving average'] = a
```

```
/Users/mrigankanand/opt/anaconda3/lib/python3.7/site-packages/ipyker
nel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  if __name__ == '__main__':
```

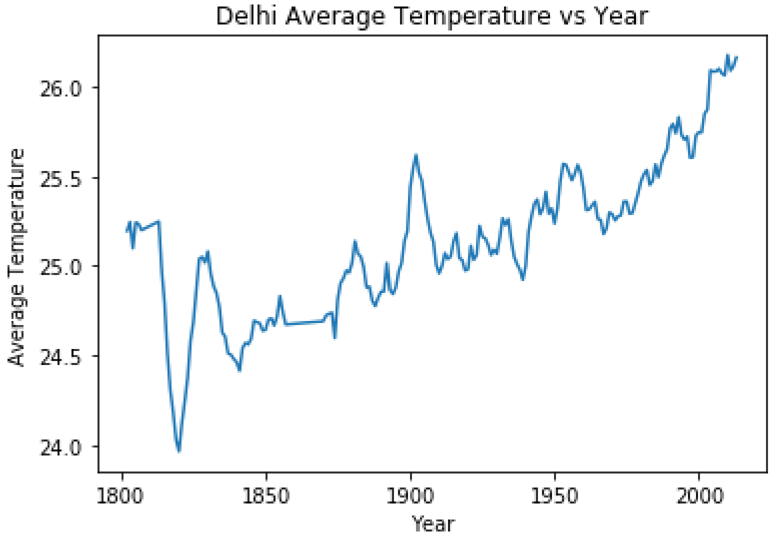
In [84]:

```
#line plot of delhi average temperature vs year
plt.plot(my_city['year'], my_city['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Delhi Average Temperature vs Year')
plt.plot();
```



In [86]:

```
#line plot of delhi moving average temperature vs year
plt.plot(my_city['year'], my_city['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Delhi Average Temperature vs Year')
plt.plot();
```

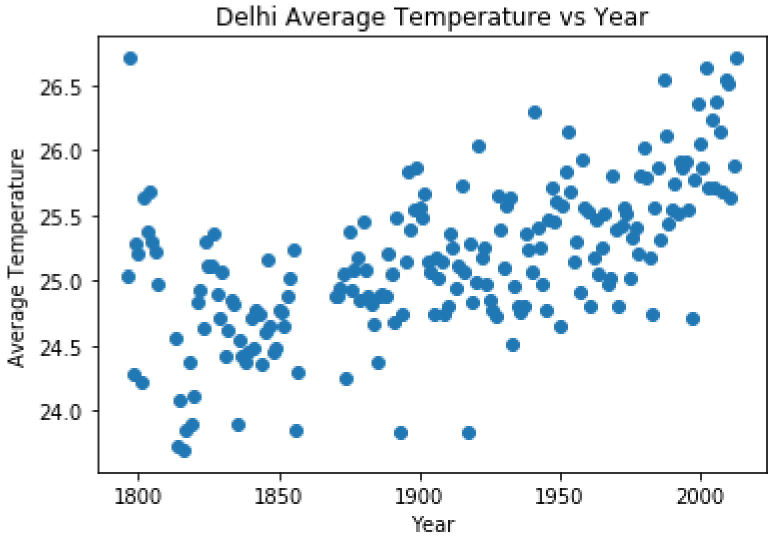


Moving average makes the graph consistent

The average Temperature of my city is highly inconsistent and the line moves up and down over years

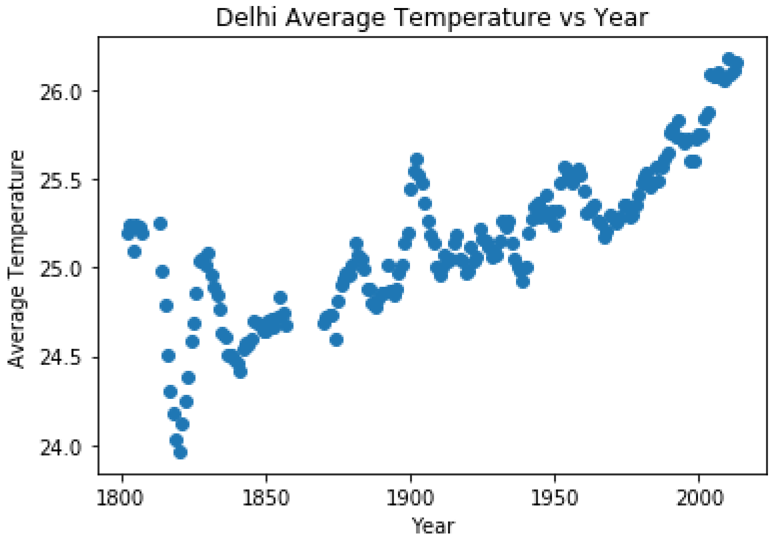
In [87]:

```
#scatter plot of delhi average temperature vs year
plt.scatter(my_city['year'], my_city['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Delhi Average Temperature vs Year')
plt.plot();
```



In [88]:

```
#scatter plot of delhi moving average temperature vs year
plt.scatter(my_city['year'], my_city['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Delhi Average Temperature vs Year')
plt.plot();
```



The scatter plot shows that the avg temperature is changing over the years and is not consistent

Inference

Delhi temperature is very much higher than average global temperature

Delhi temperature is more scattered than global temperature

Delhi is a hot place

Global temperature and Delhi's temperature is increasing over the year

The reason of increasing temperature is global warming

Global temperature is becoming consistent but delhi's temperature is scattered over the years consistently

In [18]:

```
global_data['avg_temp'].mean()
```

Out[18]:

```
8.36947368421053
```

In [19]:

```
my_city['avg_temp'].mean()
```

Out[19]:

```
25.16626865671644
```

Global Mean temperature is 8.36947 celcius and Delhi mean temperature is 25.1662 celcius

In [20]:

```
global_data['avg_temp'].std()
```

Out[20]:

```
0.5847474097994195
```

In [21]:

```
my_city['avg_temp'].std()
```

Out[21]:

```
0.5940029524023316
```

Global Standard deviation of temperature is 0.5847 celcius and Delhi Standard deviation of temperature is 0.5940 celcius

In [22]:

```
# I will be analysing my favourite city
fav_city = city[city['city'] == 'Berlin']
```

In [93]:

```
# printing some rows of Berlin temperature
fav_city.head()
```

Out[93]:

	year	city	country	avg_temp
8990	1743	Berlin	Germany	6.33
8991	1744	Berlin	Germany	10.36
8992	1745	Berlin	Germany	1.43
8997	1750	Berlin	Germany	9.83
8998	1751	Berlin	Germany	9.75

In [94]:

```
fav_city.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 267 entries, 8990 to 9260
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   year      267 non-null    int64
 1   city      267 non-null    object
 2   country   267 non-null    object
 3   avg_temp  267 non-null    float64
dtypes: float64(1), int64(1), object(2)
memory usage: 20.4+ KB
```

In [92]:

```
# printing shape of Berlin's Dataframe
fav_city.shape
```

Out[92]:

```
(267, 4)
```

In [96]:

```
a =[ None, None, None, None, None, None]
for i in range(6, fav_city.shape[0]):
    b = 0
    for j in range(7):
        b = b + fav_city.iloc[i-j,3:4][0]
    a.append(b / 7)
a = np.array(a)
fav_city['moving average'] = a
```

```
/Users/mrigankanand/opt/anaconda3/lib/python3.7/site-packages/ipyker
nel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
```

In [98]:

```
# printing shape of Berlin's Dataframe
fav_city.shape
```

Out[98]:

```
(267, 5)
```

In [102]:

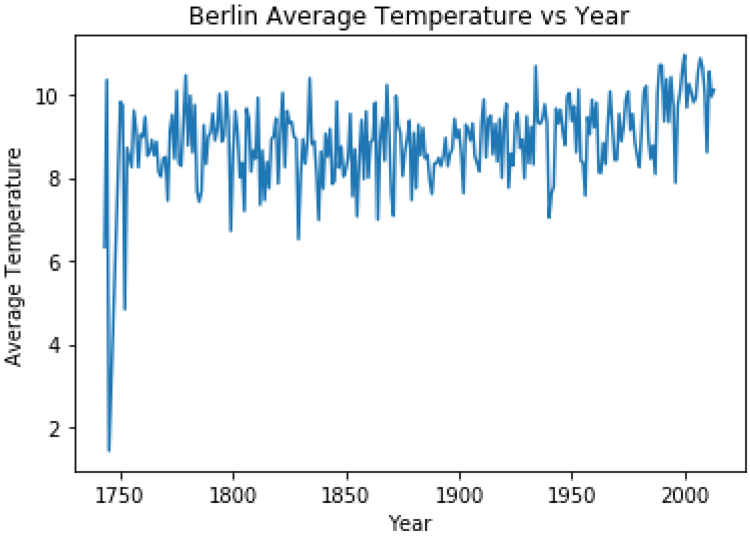
```
fav_city.head(10)
```

Out[102]:

	year	city	country	avg_temp	moving average
8990	1743	Berlin	Germany	6.33	None
8991	1744	Berlin	Germany	10.36	None
8992	1745	Berlin	Germany	1.43	None
8997	1750	Berlin	Germany	9.83	None
8998	1751	Berlin	Germany	9.75	None
8999	1752	Berlin	Germany	4.84	None
9000	1753	Berlin	Germany	8.72	7.32286
9001	1754	Berlin	Germany	8.49	7.63143
9002	1755	Berlin	Germany	8.26	7.33143
9003	1756	Berlin	Germany	9.62	8.50143

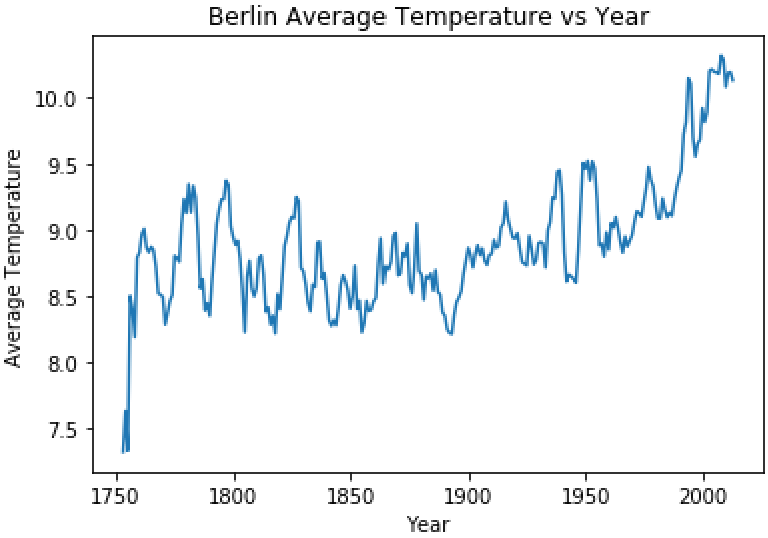
In [103]:

```
# line plot of average temperature vs year
plt.plot(fav_city['year'], fav_city['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Berlin Average Temperature vs Year')
plt.plot();
```



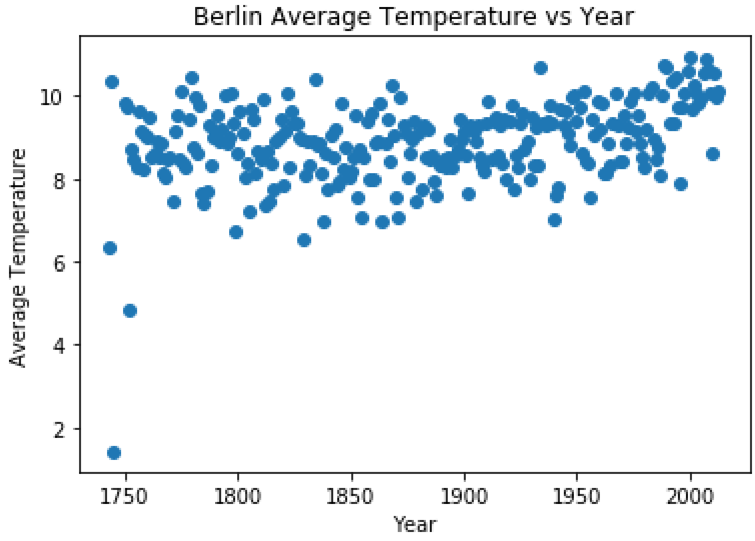
In [104]:

```
# line plot of moving average temperature vs year
plt.plot(fav_city['year'], fav_city['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Berlin Average Temperature vs Year')
plt.plot();
```



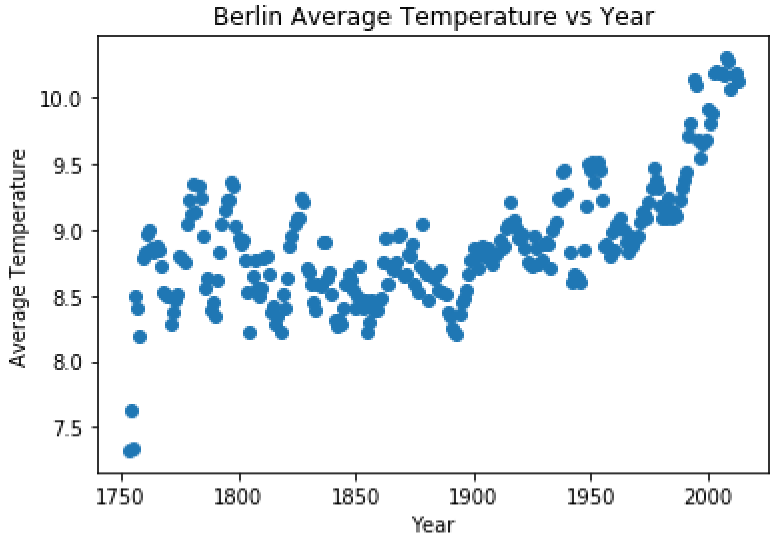
In [105]:

```
# scatter plot of average temperature vs year
plt.scatter(fav_city['year'], fav_city['avg_temp'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Berlin Average Temperature vs Year')
plt.plot();
```



In [106]:

```
# scatter plot of moving average temperature vs year
plt.scatter(fav_city['year'], fav_city['moving average'])
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.title('Berlin Average Temperature vs Year')
plt.plot();
```



Berlin is quite cold city and its average temperature is near global temperature

Berlin temperature is nearly consistent over the year's and does not have high up and down

Berlin temperature is nearly equal to average global temperature

Berlin temperature is more consistent than global temperature

Berlin is a cold place

In [27]:

```
fav_city['avg_temp'].mean()
```

Out[27]:

```
8.885393258426967
```

In [28]:

```
global_data['avg_temp'].mean()
```

Out[28]:

```
8.36947368421053
```

Global Mean temperature is 8.36947 celcius and Berlin mean temperature is 8.3694 celcius

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```