Load the Data of city temperature.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')

df =pd.read_csv(r'D:\MTECH\city_temperature\city_temperature.csv')
df.head()
```

Out[1]:

	Region	Country	State	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	NaN	Algiers	1	1	1995	64.2
1	Africa	Algeria	NaN	Algiers	1	2	1995	49.4
2	Africa	Algeria	NaN	Algiers	1	3	1995	48.8
3	Africa	Algeria	NaN	Algiers	1	4	1995	46.4
4	Africa	Algeria	NaN	Algiers	1	5	1995	47.9

In [2]:
```
df['Region'].value_counts()
```

Out[2]:
```
North America                       1556681
Europe                               381990
Asia                                 316663
Africa                               251118
South/Central America & Carribean    219530
Middle East                          124749
Australia/South Pacific               55596
Name: Region, dtype: int64
```

Data cleaning

In [3]:
```
df = df.drop_duplicates()
df.head()
```

Out[3]:

	Region	Country	State	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	NaN	Algiers	1	1	1995	64.2
1	Africa	Algeria	NaN	Algiers	1	2	1995	49.4
2	Africa	Algeria	NaN	Algiers	1	3	1995	48.8
3	Africa	Algeria	NaN	Algiers	1	4	1995	46.4
4	Africa	Algeria	NaN	Algiers	1	5	1995	47.9

In [4]:
```
df.describe()
```

Out[4]:

	Month	Day	Year	AvgTemperature
count	2.885612e+06	2.885612e+06	2.885612e+06	2.885612e+06
mean	6.469037e+00	1.571679e+01	2.006707e+03	5.601560e+01
std	3.456445e+00	8.800516e+00	1.965710e+01	3.215210e+01
min	1.000000e+00	0.000000e+00	2.000000e+02	-9.900000e+01
25%	3.000000e+00	8.000000e+00	2.001000e+03	4.580000e+01
50%	6.000000e+00	1.600000e+01	2.007000e+03	6.250000e+01
75%	9.000000e+00	2.300000e+01	2.013000e+03	7.560000e+01
max	1.200000e+01	3.100000e+01	2.020000e+03	1.100000e+02

In [5]:
```
cardinality = {}

for col in df:
    cardinality[col] = df[col].nunique() #all the different column values.

cardinality
```

Out[5]:
```
{'Region': 7,
 'Country': 125,
 'State': 52,
 'City': 321,
 'Month': 12,
 'Day': 32,
 'Year': 28,
 'AvgTemperature': 1517}
```

In [6]:
```
df.shape[0]
```

Out[6]:
```
2885612
```

In [7]:
```
missing_values = df.isna().sum() #is there any missing values
missing_values
```

Out[7]:
```
Region                 0
Country                0
State            1448805
City                   0
Month                  0
Day                    0
Year                   0
AvgTemperature         0
dtype: int64
```

In [8]:
```
df = df.drop(df.columns[[2]], axis=1)
df.head()
```

Out[8]:

	Region	Country	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	Algiers	1	1	1995	64.2
1	Africa	Algeria	Algiers	1	2	1995	49.4
2	Africa	Algeria	Algiers	1	3	1995	48.8
3	Africa	Algeria	Algiers	1	4	1995	46.4
4	Africa	Algeria	Algiers	1	5	1995	47.9

In [9]:
```
Region_order = df.Region.unique()
Region_order
```

Out[9]:
```
array(['Africa', 'Asia', 'Australia/South Pacific', 'Europe',
       'Middle East', 'North America',
       'South/Central America & Carribean'], dtype=object)
```

In [10]:
```
df.Day.unique()
```

Out[10]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31,  0],
      dtype=int64)
```

In [11]:
```
day_zero = df[df.Day == 0]
day_zero
```

Out[11]:

	Region	Country	City	Month	Day	Year	AvgTemperature
82774	Africa	Guinea	Conakry	3	0	2008	-99.0
85697	Africa	Guinea	Conakry	3	0	2016	-99.0
92041	Africa	Guinea-Bissau	Bissau	3	0	2008	-99.0
146077	Africa	Malawi	Lilongwe	3	0	2012	-99.0
177862	Africa	Nigeria	Lagos	3	0	2008	-99.0
241159	Africa	Uganda	Kampala	3	0	2012	-99.0
1209901	North America	Mexico	Guadalajara	3	0	2012	-99.0
1333910	South/Central America & Carribean	Cuba	Havana	3	0	2008	-99.0

In [12]:
```
data = df[df.Day != 0] #make a new data frame without having any 0 values in day
data.head()
```

Out[12]:

	Region	Country	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	Algiers	1	1	1995	64.2
1	Africa	Algeria	Algiers	1	2	1995	49.4
2	Africa	Algeria	Algiers	1	3	1995	48.8
3	Africa	Algeria	Algiers	1	4	1995	46.4
4	Africa	Algeria	Algiers	1	5	1995	47.9

In [13]:
```
month_order = df.Month.unique()
month_order
```

Out[13]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12], dtype=int64)
```

In [14]:
```
year_order = df.Year.unique()

year_order
```

Out[14]:
```
array([1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,
       2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016,
       2017, 2018, 2019, 2020,  201,  200], dtype=int64)
```

In [15]:
```
data =data [(data.Year!=200) & (data.Year!=201)]
data.head()
```

Out[15]:

	Region	Country	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	Algiers	1	1	1995	64.2
1	Africa	Algeria	Algiers	1	2	1995	49.4
2	Africa	Algeria	Algiers	1	3	1995	48.8
3	Africa	Algeria	Algiers	1	4	1995	46.4
4	Africa	Algeria	Algiers	1	5	1995	47.9

In [16]:
```
data = data[data['AvgTemperature'] != -99] #as per data -99 is the data when t
he value is not recorded officially
data
```

Out[16]:

	Region	Country	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	Algiers	1	1	1995	64.2
1	Africa	Algeria	Algiers	1	2	1995	49.4
2	Africa	Algeria	Algiers	1	3	1995	48.8
3	Africa	Algeria	Algiers	1	4	1995	46.4
4	Africa	Algeria	Algiers	1	5	1995	47.9
...	...	...	...	...	...	...	...
2906322	North America	US	San Juan Puerto Rico	7	27	2013	82.4
2906323	North America	US	San Juan Puerto Rico	7	28	2013	81.6
2906324	North America	US	San Juan Puerto Rico	7	29	2013	84.2
2906325	North America	US	San Juan Puerto Rico	7	30	2013	83.8
2906326	North America	US	San Juan Puerto Rico	7	31	2013	83.6

2806369 rows × 7 columns

In [17]:
```
datetime_series = pd.to_datetime(data[['Year', 'Month', 'Day']])
data['Date'] = datetime_series
df = data.set_index('Date')
df = df.drop(['Year', 'Month', 'Day'],axis = 1)
df.head()
```

Out[17]:

Date	Region	Country	City	AvgTemperature
1995-01-01	Africa	Algeria	Algiers	64.2
1995-01-02	Africa	Algeria	Algiers	49.4
1995-01-03	Africa	Algeria	Algiers	48.8
1995-01-04	Africa	Algeria	Algiers	46.4
1995-01-05	Africa	Algeria	Algiers	47.9

Data analysis and Visualization

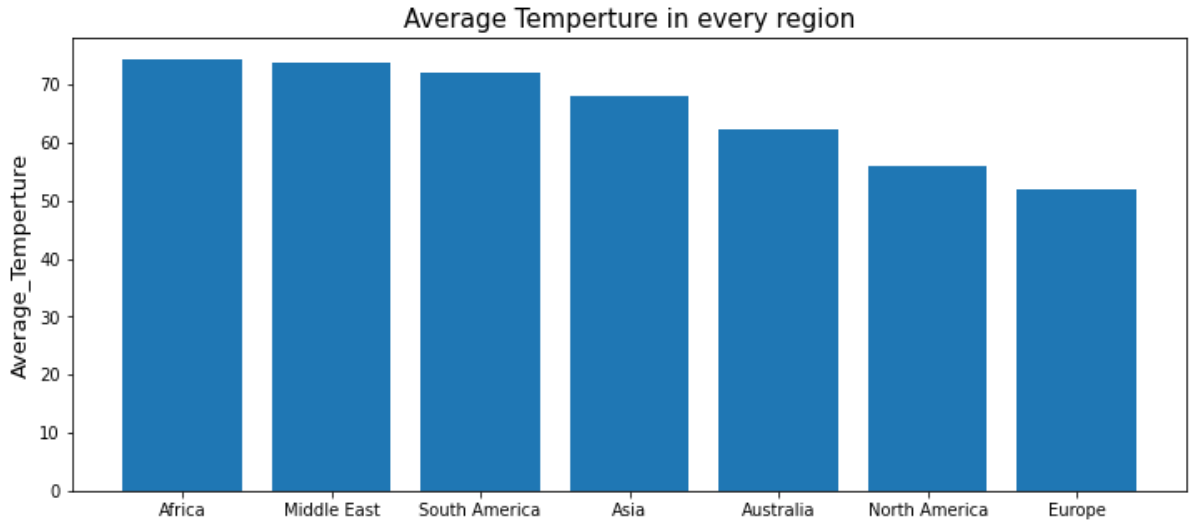
In [18]:
```
Average_Temperture_in_every_region = df.groupby("Region")["AvgTemperature"].mean().sort_values()[-1::-1] #avg temp. of all the region
Average_Temperture_in_every_region
```

Out[18]:
```
Region
Africa                                74.402602
Middle East                           73.840683
South/Central America & Carribean     72.202024
Asia                                  68.109723
Australia/South Pacific               62.303693
North America                         56.121249
Europe                                51.958137
Name: AvgTemperature, dtype: float64
```

In [19]:
```
Average_Temperture_in_every_region = Average_Temperture_in_every_region.rename({"South/Central America & Carribean":"South America","Australia/South Pacific":"Australia"})
Average_Temperture_in_every_region
```

Out[19]:
```
Region
Africa           74.402602
Middle East      73.840683
South America    72.202024
Asia             68.109723
Australia        62.303693
North America    56.121249
Europe           51.958137
Name: AvgTemperature, dtype: float64
```

In [20]:
```
plt.figure(figsize = (12,5))
plt.bar(Average_Temperture_in_every_region.index,Average_Temperture_in_every_region.values)
plt.ylabel("Average_Temperture",size = 13)
plt.title("Average Temperture in every region",size = 15)
plt.show()
```



In above graph we can clearly see that Africa, Middle East and South America have high temp. than rest of the continents. however, Europe has the lowest temp than any other region

In [21]:
```
data['AvgTemperature'].mean()
```

Out[21]:
```
60.39274746122129
```

In [22]:
```
data[data['AvgTemperature'] == max(data['AvgTemperature'])] #hottest region
```

Out[22]:

	Region	Country	City	Month	Day	Year	AvgTemperature	Date
1034962	Middle East	Kuwait	Kuwait	8	1	2012	110.0	2012-08-01

In [23]:
```
data[data['AvgTemperature'] == min(data['AvgTemperature'])] #coolest region
```

Out[23]:

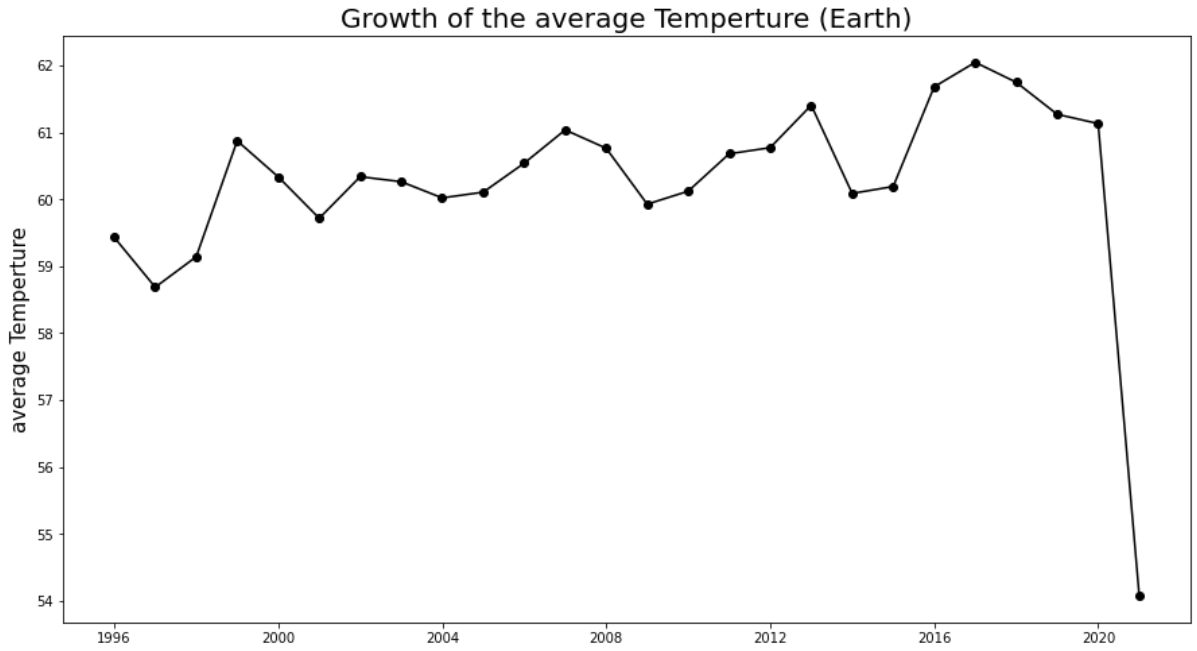
	Region	Country	City	Month	Day	Year	AvgTemperature	Date
1499140	North America	US	Fairbanks	12	31	1999	-50.0	1999-12-31

In [24]:
```
df_earth = df.groupby([pd.Grouper(freq = "Y")]).mean()
df_earth.head()
```

Out[24]:

Date	AvgTemperature
1995-12-31	59.433860
1996-12-31	58.687758
1997-12-31	59.142613
1998-12-31	60.875079
1999-12-31	60.337522

In [25]:
```
plt.figure(figsize = (15,8))
plt.plot(df_earth.index,df_earth.values,marker ="o",color = 'black')
plt.ylabel("average Temperture",size = 15)
plt.title("Growth of the average Temperture (Earth)",size =20)
plt.show()
```



In above graph the graph is fluctuating between 60-70 F in the year 2002 to 2014. In the year 2020, avg temp of earth is gradually decreasing from 61 to 54 we can co-relate it with COVID-19 that,One of the few positives of the pandemic, has been lockdown's effect on the environment.

```
# Top 20 hottest cities of ASIA
```

In [26]:
```
data_region_Asia = df[df['Region']=='Asia']
data_region_Asia
```

Out[26]:

Date	Region	Country	City	AvgTemperature
1995-01-01	Asia	Bangladesh	Dhaka	64.8
1995-01-02	Asia	Bangladesh	Dhaka	58.4
1995-01-04	Asia	Bangladesh	Dhaka	59.6
1995-01-05	Asia	Bangladesh	Dhaka	64.9
1995-01-07	Asia	Bangladesh	Dhaka	71.9
...	...	...	...	...
2020-05-09	Asia	Vietnam	Hanoi	84.1
2020-05-10	Asia	Vietnam	Hanoi	87.0
2020-05-11	Asia	Vietnam	Hanoi	84.5
2020-05-12	Asia	Vietnam	Hanoi	80.9
2020-05-13	Asia	Vietnam	Hanoi	82.4

306163 rows × 4 columns

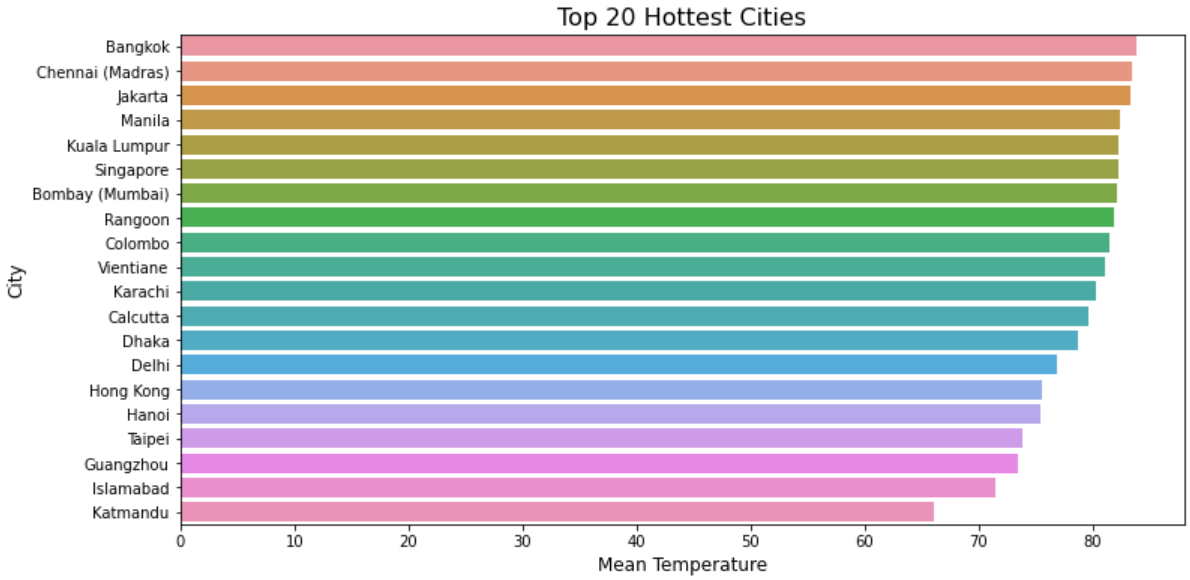
In [27]:
```
top_20_hotest_Cities_in_Asia = data_region_Asia.groupby("City").mean().sort_values(by = "AvgTemperature")[-1:14:-1]
top_20_hotest_Cities_in_Asia
```

Out[27]:

City	AvgTemperature
Bangkok	83.918827
Chennai (Madras)	83.417939
Jakarta	83.366229
Manila	82.364495
Kuala Lumpur	82.298922
Singapore	82.241197
Bombay (Mumbai)	82.118125
Rangoon	81.897779
Colombo	81.446830
Vientiane	81.055646
Karachi	80.278237
Calcutta	79.662352
Dhaka	78.720325
Delhi	76.938229
Hong Kong	75.628162
Hanoi	75.414951
Taipei	73.886466
Guangzhou	73.406714
Islamabad	71.492958
Katmandu	66.040979

In [28]:
```
import seaborn as sns
plt.figure(figsize=(12,6))
Asia = data[data['Region']=='Asia']
city_stats = Asia.groupby('City')['AvgTemperature'].agg(mean_temp='mean',std_t
emp = 'std',min_temp = 'min',max_temp = 'max',median_temp = 'median').reset_in
dex().sort_values('mean_temp',ascending=False).head(20)
sns.barplot(x='mean_temp',y='City',data=city_stats)
plt.xlabel("Mean Temperature",fontsize=12)
plt.ylabel('City',fontsize=12)
plt.title("Top 20 Hottest Cities",fontsize=16)
```

Out[28]: Text(0.5, 1.0, 'Top 20 Hottest Cities')



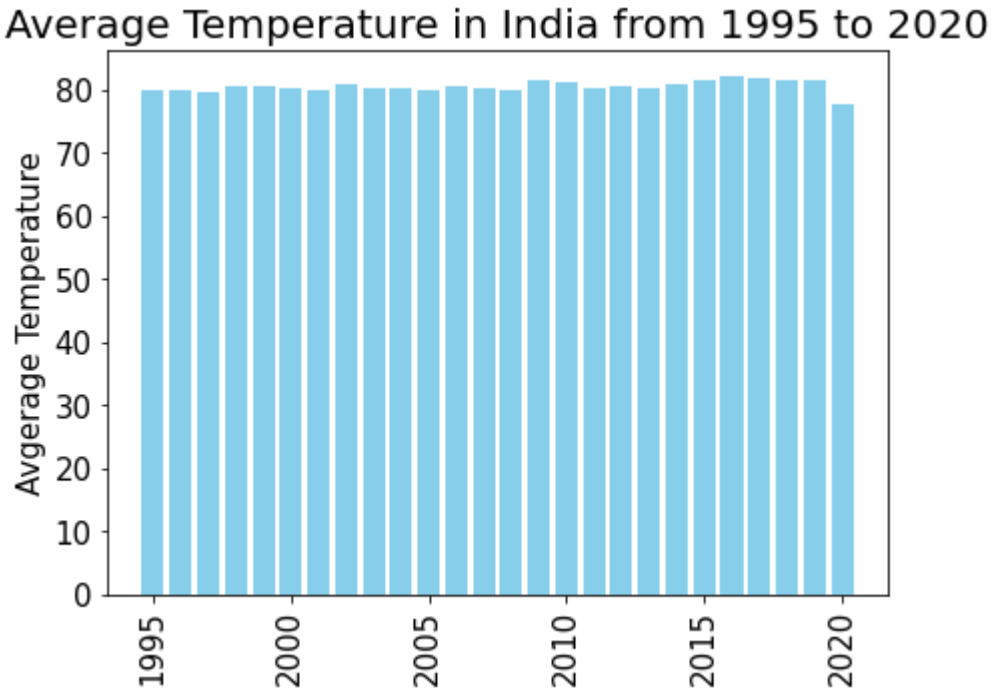
Temperature of major cities of India

In [29]:
```
Temperature_IN_year = df.loc[df["Country"] == "India"].groupby(pd.Grouper(freq
= "Y")).mean()
Temperature_IN_year.head()
```

Out[29]:

Date	AvgTemperature
1995-12-31	79.979147
1996-12-31	79.841358
1997-12-31	79.589247
1998-12-31	80.547675
1999-12-31	80.438446

In [30]:
```
plt.figure(figsize = (7,5))
plt.bar(Temperature_IN_year.index.year,Temperature_IN_year.AvgTemperature, col
or='skyblue')
plt.yticks(size = 15)
plt.xticks(size = 15,rotation = 90)
plt.xlabel(None)
plt.ylabel("Avgerage Temperature",size = 15)
plt.title("Average Temperature in India from 1995 to 2020",size = 20)
plt.show()
```



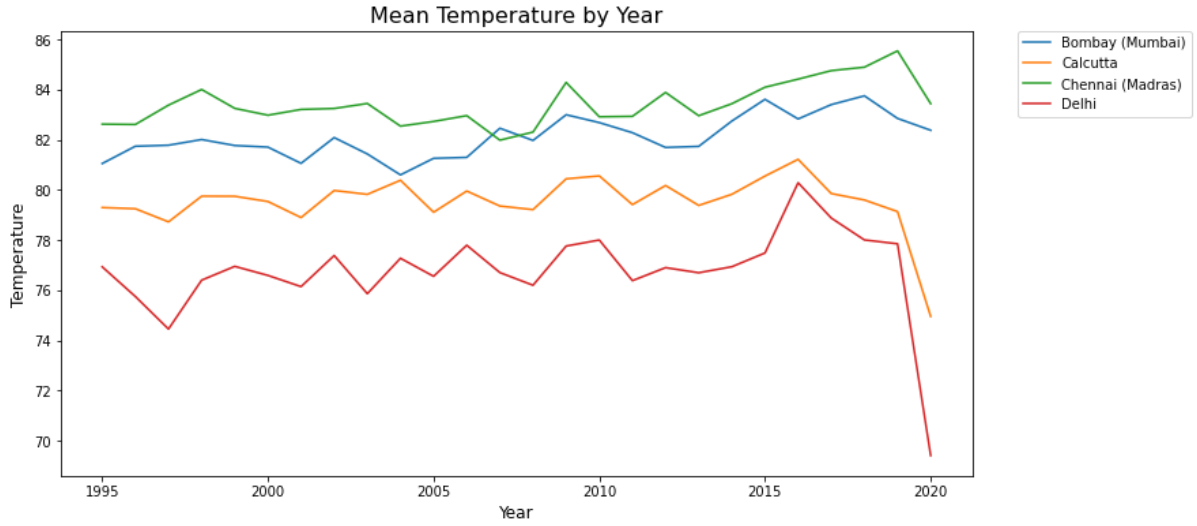
In [31]:
```
data_Con_india = data[data['Country']=='India']
data_Con_india.head()
```

Out[31]:

	Region	Country	City	Month	Day	Year	AvgTemperature	Date
312523	Asia	India	Bombay (Mumbai)	1	1	1995	71.8	1995-01-01
312524	Asia	India	Bombay (Mumbai)	1	2	1995	72.0	1995-01-02
312525	Asia	India	Bombay (Mumbai)	1	3	1995	70.3	1995-01-03
312526	Asia	India	Bombay (Mumbai)	1	4	1995	69.7	1995-01-04
312527	Asia	India	Bombay (Mumbai)	1	5	1995	71.3	1995-01-05

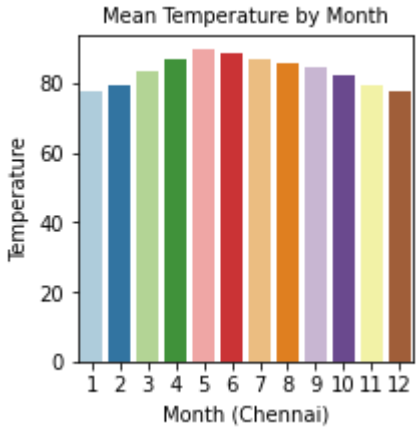
In [32]:
```
plt.figure(figsize=(12,6))
data_Con_india.head()
year_stats = data_Con_india.groupby(['Year','City'])['AvgTemperature'].agg(mea
n_temp='mean').reset_index()
year_stats.head()
sns.lineplot(x='Year',y='mean_temp',hue='City',data=year_stats)
plt.xlabel("Year",fontsize=12)
plt.ylabel('Temperature',fontsize=12)
plt.title("Mean Temperature by Year",fontsize=16)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
```

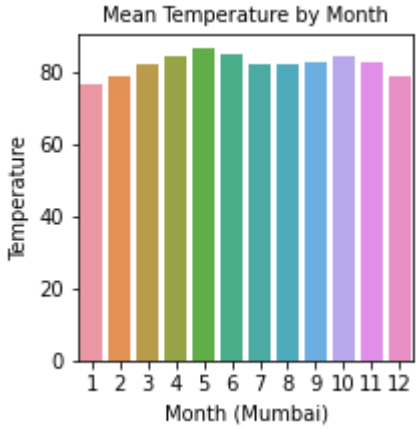
Out[32]: <matplotlib.legend.Legend at 0x23b14e8f850>

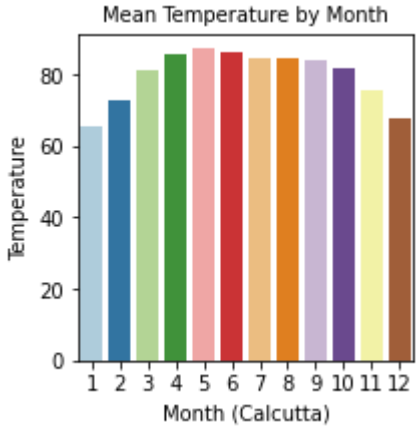


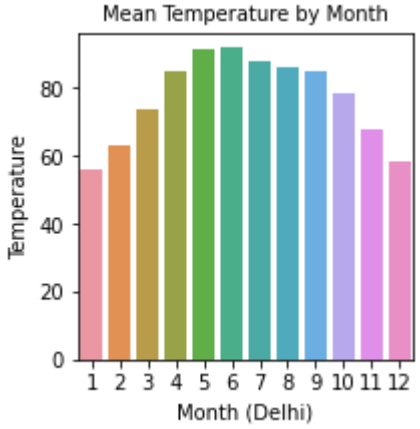
In [33]:
```
chennai = data[data['City']=='Chennai (Madras)']
plt.figure(figsize=(3,3))
India_stats = chennai.groupby('Month')['AvgTemperature'].agg(mean_temp='mean')
.reset_index()
sns.barplot(x='Month',y='mean_temp',data=India_stats , palette='Paired')
plt.xlabel("Month (Chennai)",fontsize=10)
plt.ylabel('Temperature',fontsize=10)
plt.title("Mean Temperature by Month",fontsize=10)

bombay = data[data['City']=='Bombay (Mumbai)']
plt.figure(figsize=(3,3))
India_stats = bombay.groupby('Month')['AvgTemperature'].agg(mean_temp='mean').
reset_index()
sns.barplot(x='Month',y='mean_temp',data=India_stats)
plt.xlabel("Month (Mumbai)",fontsize=10)
plt.ylabel('Temperature',fontsize=10)
plt.title("Mean Temperature by Month",fontsize=10)

calcutta = data[data['City']=='Calcutta']
plt.figure(figsize=(3,3))
India_stats = calcutta.groupby('Month')['AvgTemperature'].agg(mean_temp='mean'
).reset_index()
sns.barplot(x='Month',y='mean_temp',data=India_stats, palette='Paired')
plt.xlabel("Month (Calcutta)",fontsize=10)
plt.ylabel('Temperature',fontsize=10)
plt.title("Mean Temperature by Month",fontsize=10)

delhi = data[data['City']=='Delhi']
plt.figure(figsize=(3,3))
India_stats = delhi.groupby('Month')['AvgTemperature'].agg(mean_temp='mean').r
eset_index()
sns.barplot(x='Month',y='mean_temp',data=India_stats)
plt.xlabel("Month (Delhi)",fontsize=10)
plt.ylabel('Temperature',fontsize=10)
plt.title("Mean Temperature by Month",fontsize=10)
```

Out[33]: Text(0.5, 1.0, 'Mean Temperature by Month')



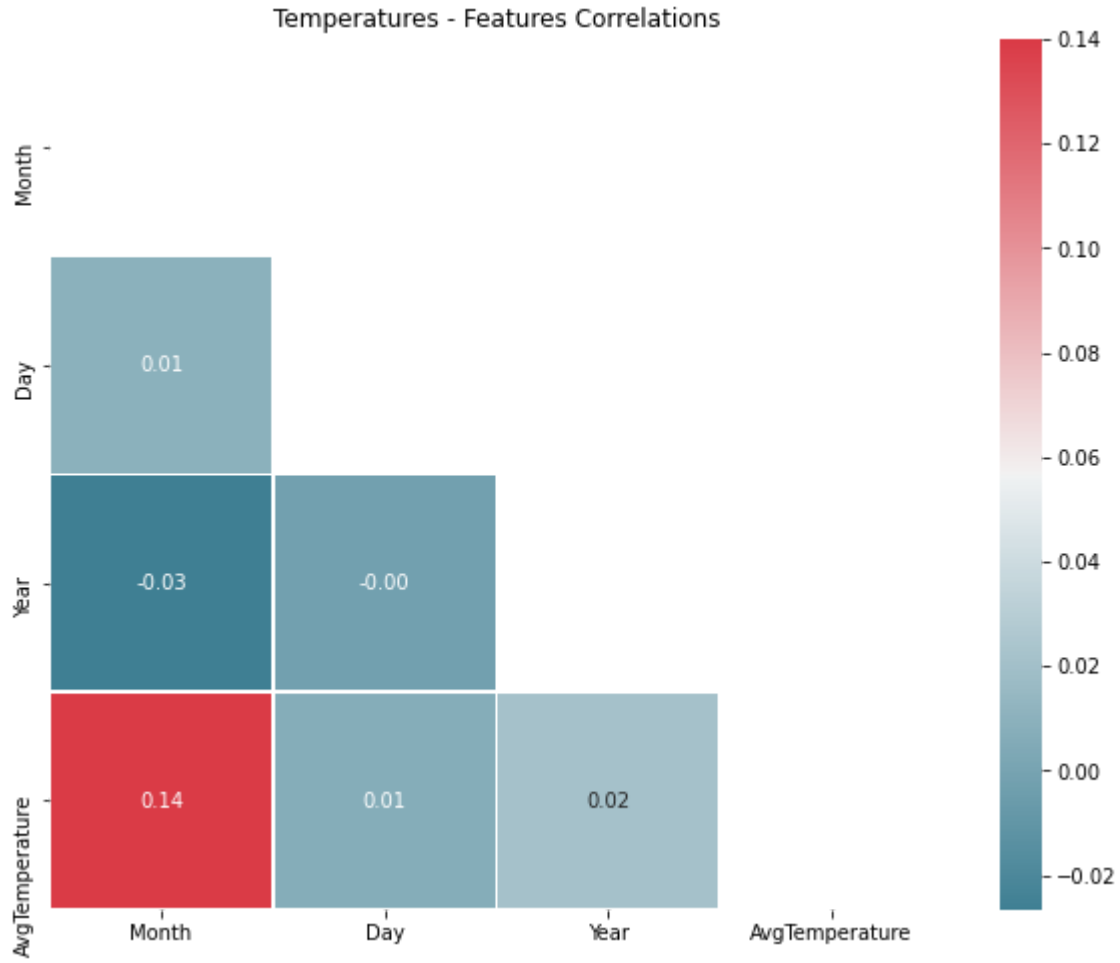






```
# Correlations between data
```

In [34]:
```
corr = data.corr()
fig, ax = plt.subplots(figsize=(10, 8))
colormap = sns.diverging_palette(220, 10, as_cmap=True)
dropSelf = np.zeros_like(corr)
dropSelf[np.triu_indices_from(dropSelf)] = True
colormap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(corr, cmap=colormap, linewidths=.5, annot=True, fmt=".2f", mask=dr
opSelf)
plt.title('Temperatures - Features Correlations')
plt.show()
```



Student’s t-test

In [35]:
```
df_delhi = data[(data.City=='Delhi')]

df_delhi_2013 = df_delhi[(df_delhi.Year==2013)]

df_delhi_2013.head()
```

Out[35]:

	Region	Country	City	Month	Day	Year	AvgTemperature	Date
346896	Asia	India	Delhi	1	1	2013	48.4	2013-01-01
346897	Asia	India	Delhi	1	2	2013	45.4	2013-01-02
346898	Asia	India	Delhi	1	3	2013	46.0	2013-01-03
346899	Asia	India	Delhi	1	4	2013	45.7	2013-01-04
346900	Asia	India	Delhi	1	5	2013	44.5	2013-01-05

In [36]:
```
df_delhi_1995 = df_delhi[(df_delhi.Year==1995)]
df_delhi_1995.head()
```

Out[36]:

	Region	Country	City	Month	Day	Year	AvgTemperature	Date
340321	Asia	India	Delhi	1	1	1995	50.7	1995-01-01
340322	Asia	India	Delhi	1	2	1995	52.1	1995-01-02
340323	Asia	India	Delhi	1	3	1995	53.8	1995-01-03
340324	Asia	India	Delhi	1	4	1995	53.7	1995-01-04
340325	Asia	India	Delhi	1	5	1995	54.5	1995-01-05

In [37]:
```
df_delhi_2013['AvgTemperature'].mean()
```

Out[37]: 76.70137362637362

In [38]:
```
df_delhi_1995['AvgTemperature'].mean()
```

Out[38]: 76.9379120879121

We can see that the average temperature of both years are very close to each other. Now, let's apply the test.

In [39]:
```
import scipy
from scipy import stats
from scipy.stats import ttest_rel

stat, p = ttest_rel(df_delhi_1995['AvgTemperature'],df_delhi_2013['AvgTemperat
ure'])
print('stat=%.3f, p=%.3f' % (stat, p))
if p > 0.05:
    print('Probably the same distribution')
else:
    print('Probably different distributions')
```

```
stat=0.850, p=0.396
Probably the same distribution
```

In [40]:
```
df_Calcutta = data[(data.City=='Calcutta')]

df_Calcutta_2013 = df_Calcutta[(df_Calcutta.Year==2013)]
```

In [41]:
```
df_Calcutta_2013['AvgTemperature'].mean()
```

Out[41]:
```
79.38547945205478
```

In [42]:
```
len(df_delhi_2013)
```

Out[42]:
```
364
```

In [43]:
```
len(df_Calcutta_2013)
```

Out[43]:
```
365
```

In [44]:
```
df2013 = df_delhi_2013["AvgTemperature"].mean()
```

In [45]:
```
df_delhi_2013.loc[16]=["Region","Country","City","Month","Day","Year",df2013,
"Date"]
```

In [46]:
```
df_delhi_2013
```

Out[46]:

	Region	Country	City	Month	Day	Year	AvgTemperature	Date
346896	Asia	India	Delhi	1	1	2013	48.400000	2013-01-01 00:00:00
346897	Asia	India	Delhi	1	2	2013	45.400000	2013-01-02 00:00:00
346898	Asia	India	Delhi	1	3	2013	46.000000	2013-01-03 00:00:00
346899	Asia	India	Delhi	1	4	2013	45.700000	2013-01-04 00:00:00
346900	Asia	India	Delhi	1	5	2013	44.500000	2013-01-05 00:00:00
...	...	...	...	...	...	...	...	...
347257	Asia	India	Delhi	12	28	2013	51.700000	2013-12-28 00:00:00
347258	Asia	India	Delhi	12	29	2013	51.300000	2013-12-29 00:00:00
347259	Asia	India	Delhi	12	30	2013	54.100000	2013-12-30 00:00:00
347260	Asia	India	Delhi	12	31	2013	57.700000	2013-12-31 00:00:00
16	Region	Country	City	Month	Day	Year	76.701374	Date

365 rows × 8 columns

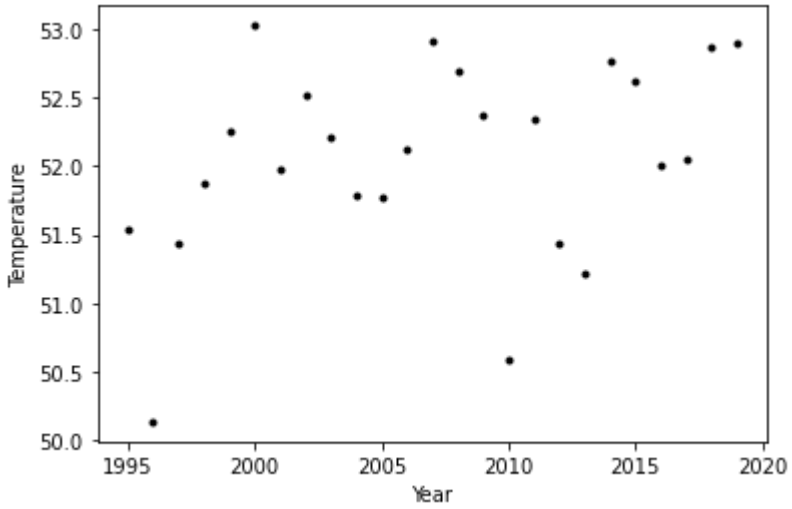
In [47]:
```
stat, p = ttest_rel(df_Calcutta_2013['AvgTemperature'],df_delhi_2013['AvgTempe
rature'])
print('stat=%.3f, p=%.3f' % (stat, p))
if p > 0.05:
    print('Probably the same distribution')
else:
    print('Probably different distributions')
```

```
stat=6.843, p=0.000
Probably different distributions
```

Linear Regression

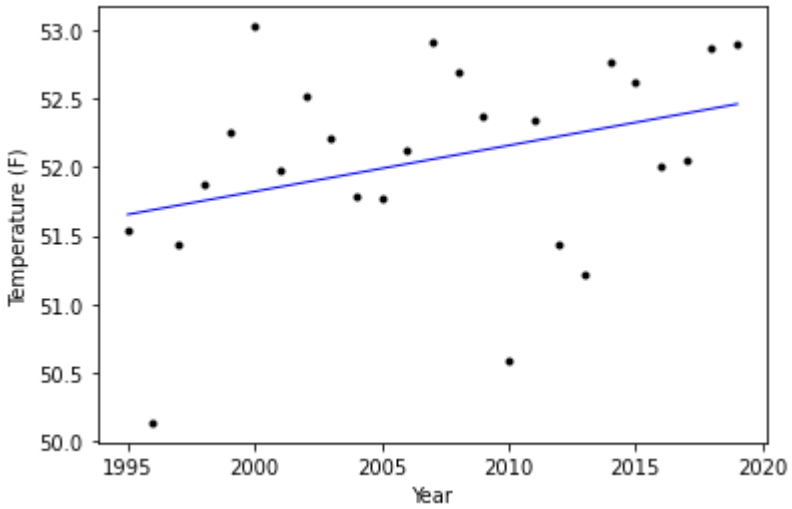
In [48]:
```
average_temperatures = data[data['Region'] == 'Europe'].groupby(data.Year).mea
n()
#removing not meaningfull rows
average_temperatures = average_temperatures[(average_temperatures.index != 202
0)]
```

In [49]:
```
plt.xlabel('Year')
plt.ylabel('Temperature')
plt.errorbar(average_temperatures.index, average_temperatures.AvgTemperature,
color='black', fmt='.')
plt.show()
```



In [50]:
```
from sklearn import linear_model
x = list(map(lambda val: [val], average_temperatures.index))
y = list(average_temperatures.AvgTemperature)

reg = linear_model.Ridge(alpha=1)
reg.fit(x, y)
prediction = reg.predict(x)
```

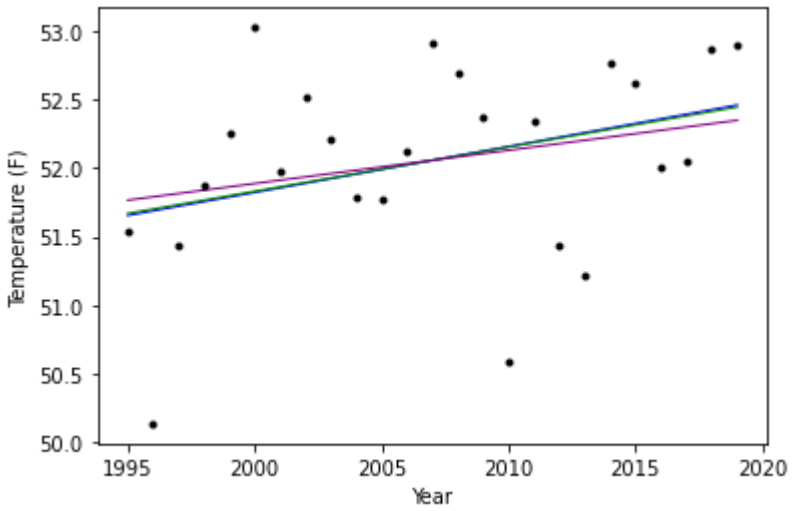
In [51]:
```
from sklearn.linear_model import LinearRegression
plt.xlabel('Year')
plt.ylabel('Temperature (F)')

plt.errorbar(average_temperatures.index, average_temperatures.AvgTemperature,
fmt='.', color='black')
plt.plot(x, prediction, color='blue', linewidth='1')
plt.show()
```



In [52]:
```
from sklearn.linear_model import LinearRegression

x = list(map(lambda val: [val], average_temperatures.index))
y = list(average_temperatures.AvgTemperature)

reg = linear_model.Ridge(alpha=1)
reg.fit(x, y)
prediction = reg.predict(x)
```

In [53]:
```
reg2 = linear_model.Ridge(alpha=50)
reg2.fit(x, y)
prediction2 = reg2.predict(x)

reg3 = linear_model.Ridge(alpha=500)
reg3.fit(x, y)
prediction3 = reg3.predict(x)
```

In [54]:
```
plt.xlabel('Year')
plt.ylabel('Temperature (F)')

plt.errorbar(average_temperatures.index, average_temperatures.AvgTemperature,
fmt='.', color='black')
plt.plot(x, prediction, color='blue', linewidth='1')
plt.plot(x, prediction2, color='green', linewidth='1')
plt.plot(x, prediction3, color='purple', linewidth='1')
plt.show()
```



In [55]:
```
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import PolynomialFeatures
x = list(map(lambda val: [val], average_temperatures.index))
y = list(average_temperatures.AvgTemperature)

# alpha 1
polymodel = make_pipeline(PolynomialFeatures(2), linear_model.Ridge(alpha=0))
polymodel.fit(x, y)
predpolymodel = polymodel.predict(x)
#alpha 2
polymodel2 = make_pipeline(PolynomialFeatures(2), linear_model.Ridge(alpha=0.0
1))
polymodel2.fit(x, y)
predpolymodel2 = polymodel2.predict(x)
#alpha 3
polymodel3 = make_pipeline(PolynomialFeatures(2), linear_model.Ridge(alpha=0.0
01))
polymodel3.fit(x, y)
predpolymodel3 = polymodel3.predict(x)
```

In [56]:
```
plt.xlabel('Year')
plt.ylabel('Temperature (F)')

plt.errorbar(average_temperatures.index, average_temperatures.AvgTemperature,
fmt='.', color='black')
p1 = plt.plot(x, predpolymodel, color='orange')
p2 = plt.plot(x, predpolymodel2, color='pink')
p3 = plt.plot(x, predpolymodel3, color='brown')

plt.legend(['alpha = 0', 'alpha = 0.01', 'alpha = 0.001'], numpoints=1)
plt.show()
```