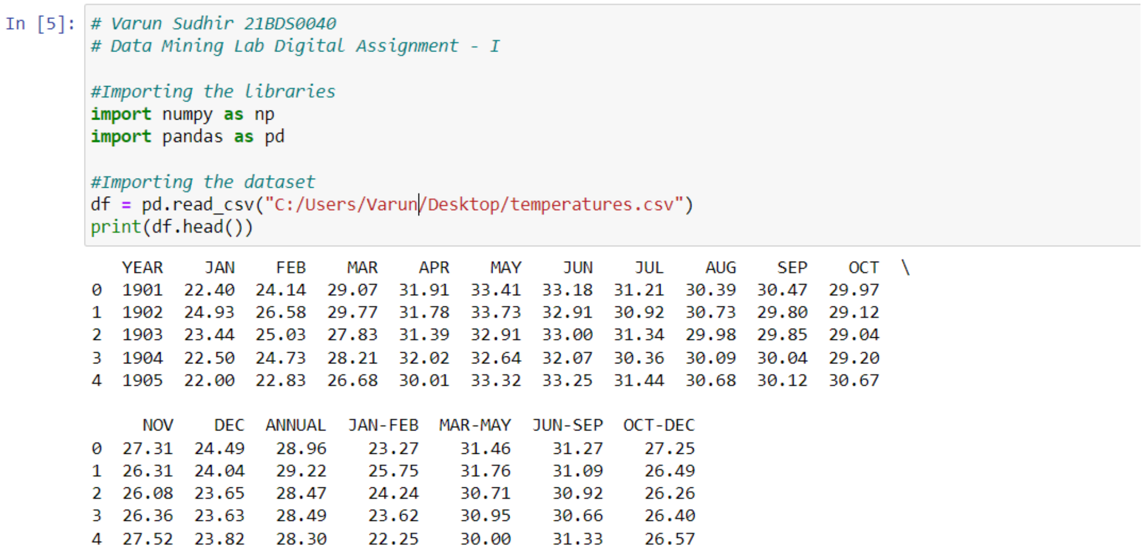
Name: Varun Sudhir

Reg No : 21BDS0040

Data Mining Digital Assignment – I

Before proceeding with the questions, we will import the dataset and display it



```
In [5]: # Varun Sudhir 21BDS0040
        # Data Mining Lab Digital Assignment - I

        #Importing the libraries
        import numpy as np
        import pandas as pd

        #Importing the dataset
        df = pd.read_csv("C:/Users/Varun/Desktop/temperatures.csv")
        print(df.head())
```

```
   YEAR    JAN    FEB    MAR    APR    MAY    JUN    JUL    AUG    SEP    OCT  \
0  1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47  29.97
1  1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80  29.12
2  1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85  29.04
3  1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04  29.20
4  1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12  30.67

     NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0  27.31  24.49   28.96    23.27    31.46    31.27    27.25
1  26.31  24.04   29.22    25.75    31.76    31.09    26.49
2  26.08  23.65   28.47    24.24    30.71    30.92    26.26
3  26.36  23.63   28.49    23.62    30.95    30.66    26.40
4  27.52  23.82   28.30    22.25    30.00    31.33    26.57
```

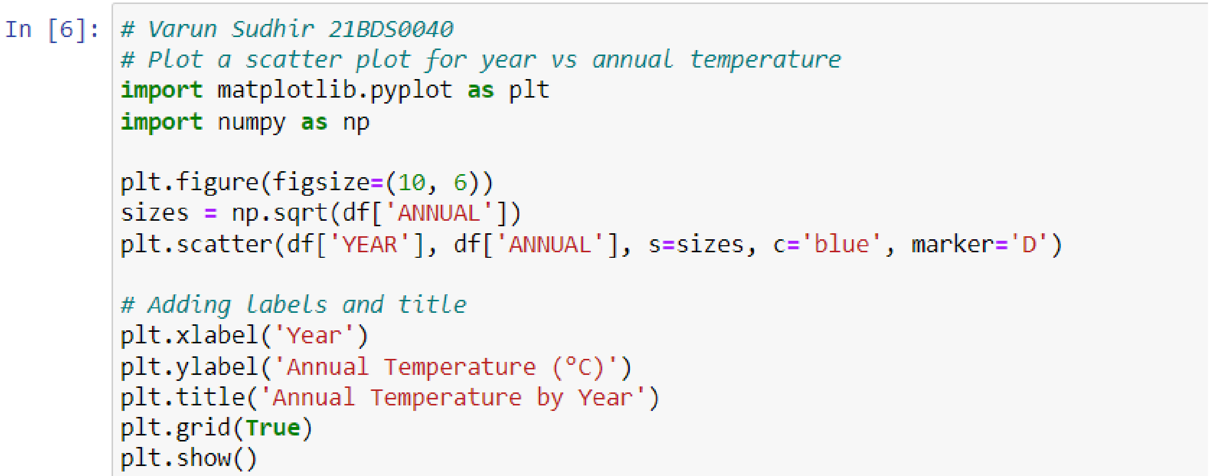
a) Plot a scatter plot for each year where (with x axis and y axis labels)

X-axis should be the YEAR

Y-axis should be the ANNUAL temperature

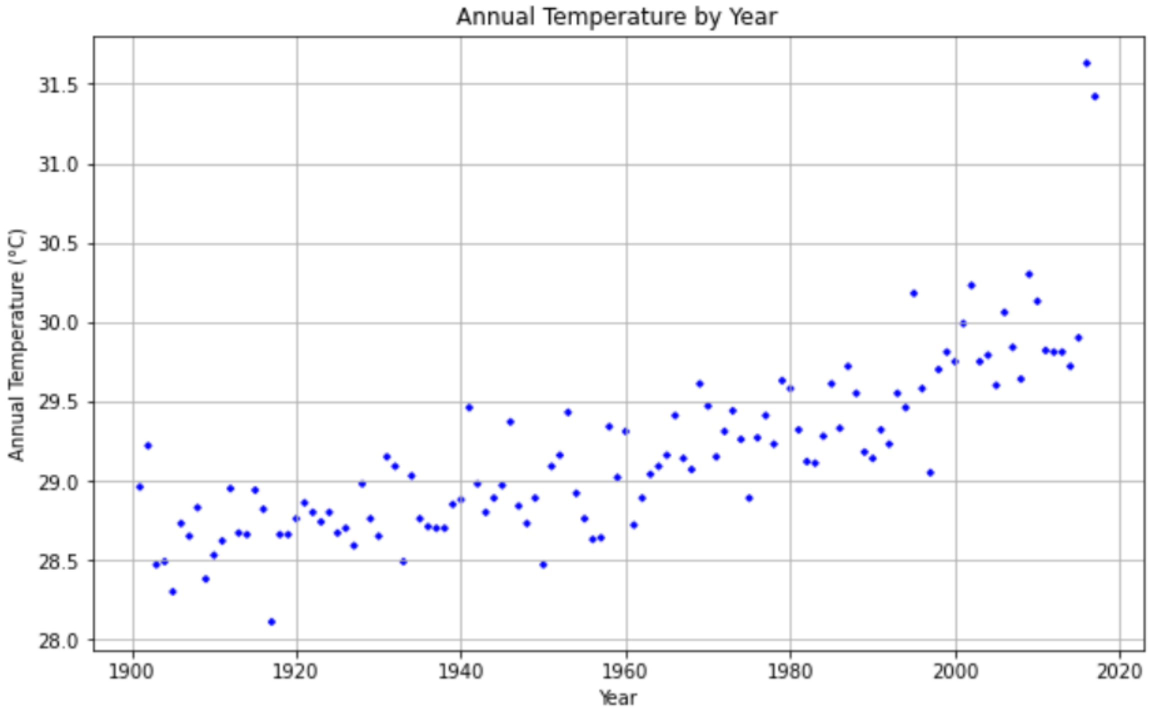
Change the marker to a diamond and size as the square root of the ANNUAL temperature.

Code:



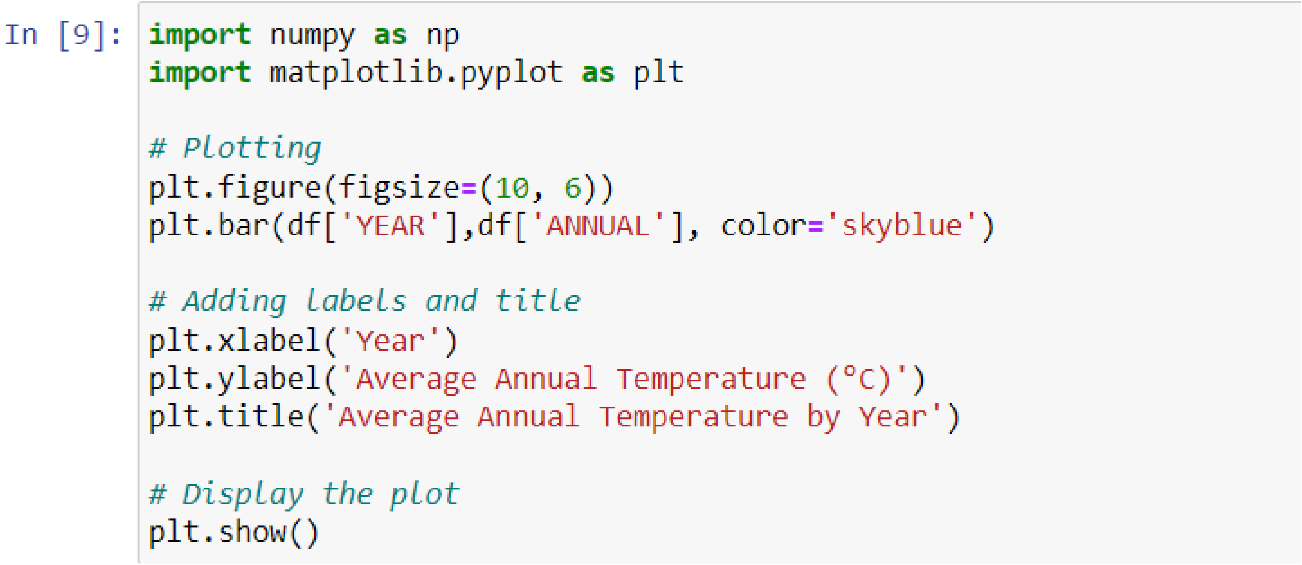
```
In [6]: # Varun Sudhir 21BDS0040
        # Plot a scatter plot for year vs annual temperature
        import matplotlib.pyplot as plt
        import numpy as np

        plt.figure(figsize=(10, 6))
        sizes = np.sqrt(df['ANNUAL'])
        plt.scatter(df['YEAR'], df['ANNUAL'], s=sizes, c='blue', marker='D')

        # Adding labels and title
        plt.xlabel('Year')
        plt.ylabel('Annual Temperature (°C)')
        plt.title('Annual Temperature by Year')
        plt.grid(True)
        plt.show()
```

Output:



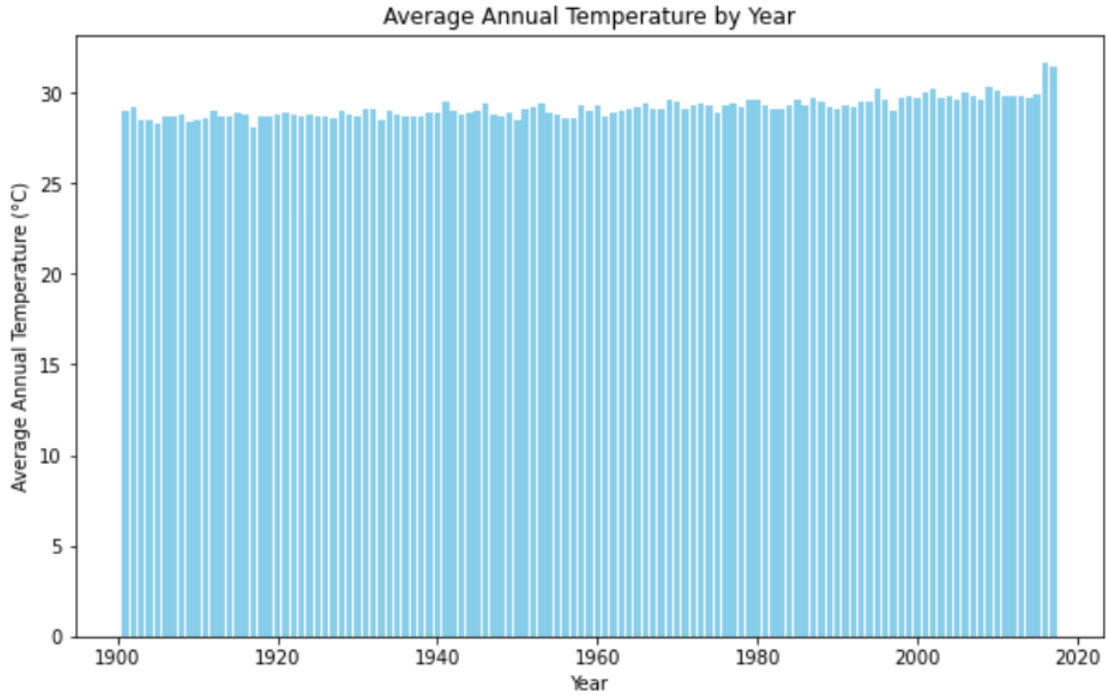
b) Group the data on the column year and display a bar chart depicting the average ANNUAL temperature for each year.

Code:



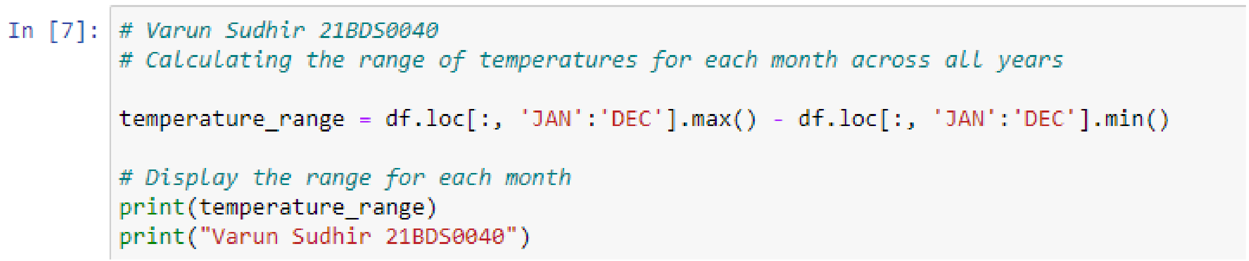
```
In [9]: import numpy as np
        import matplotlib.pyplot as plt

        # Plotting
        plt.figure(figsize=(10, 6))
        plt.bar(df['YEAR'],df['ANNUAL'], color='skyblue')

        # Adding labels and title
        plt.xlabel('Year')
        plt.ylabel('Average Annual Temperature (°C)')
        plt.title('Average Annual Temperature by Year')

        # Display the plot
        plt.show()
```

Output:



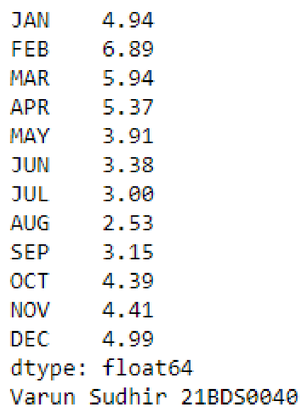
c) Calculate the range (maximum - minimum) of temperatures for each month across all years

Code:



```
In [7]: # Varun Sudhir 21BDS0040
        # Calculating the range of temperatures for each month across all years

        temperature_range = df.loc[:, 'JAN':'DEC'].max() - df.loc[:, 'JAN':'DEC'].min()

        # Display the range for each month
        print(temperature_range)
        print("Varun Sudhir 21BDS0040")
```

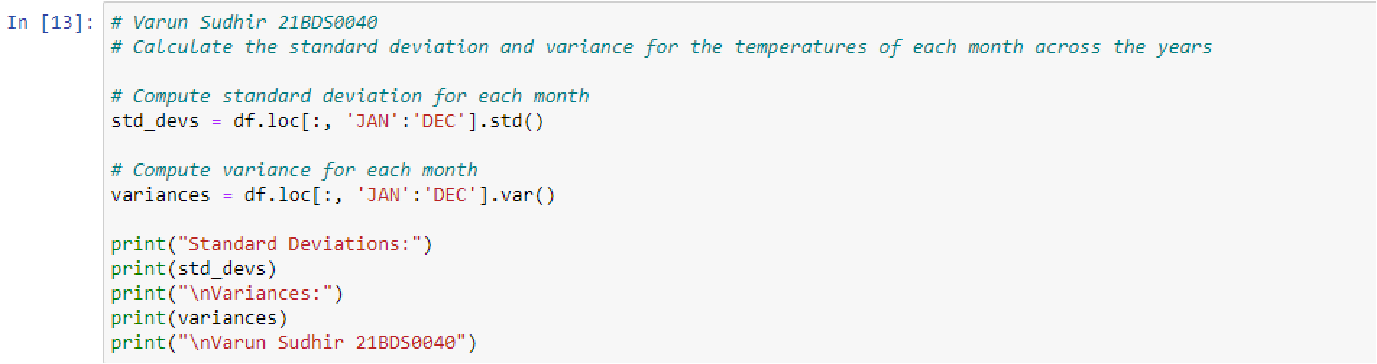
Output:



```
JAN    4.94
FEB    6.89
MAR    5.94
APR    5.37
MAY    3.91
JUN    3.38
JUL    3.00
AUG    2.53
SEP    3.15
OCT    4.39
NOV    4.41
DEC    4.99
dtype: float64
Varun Sudhir 21BDS0040
```

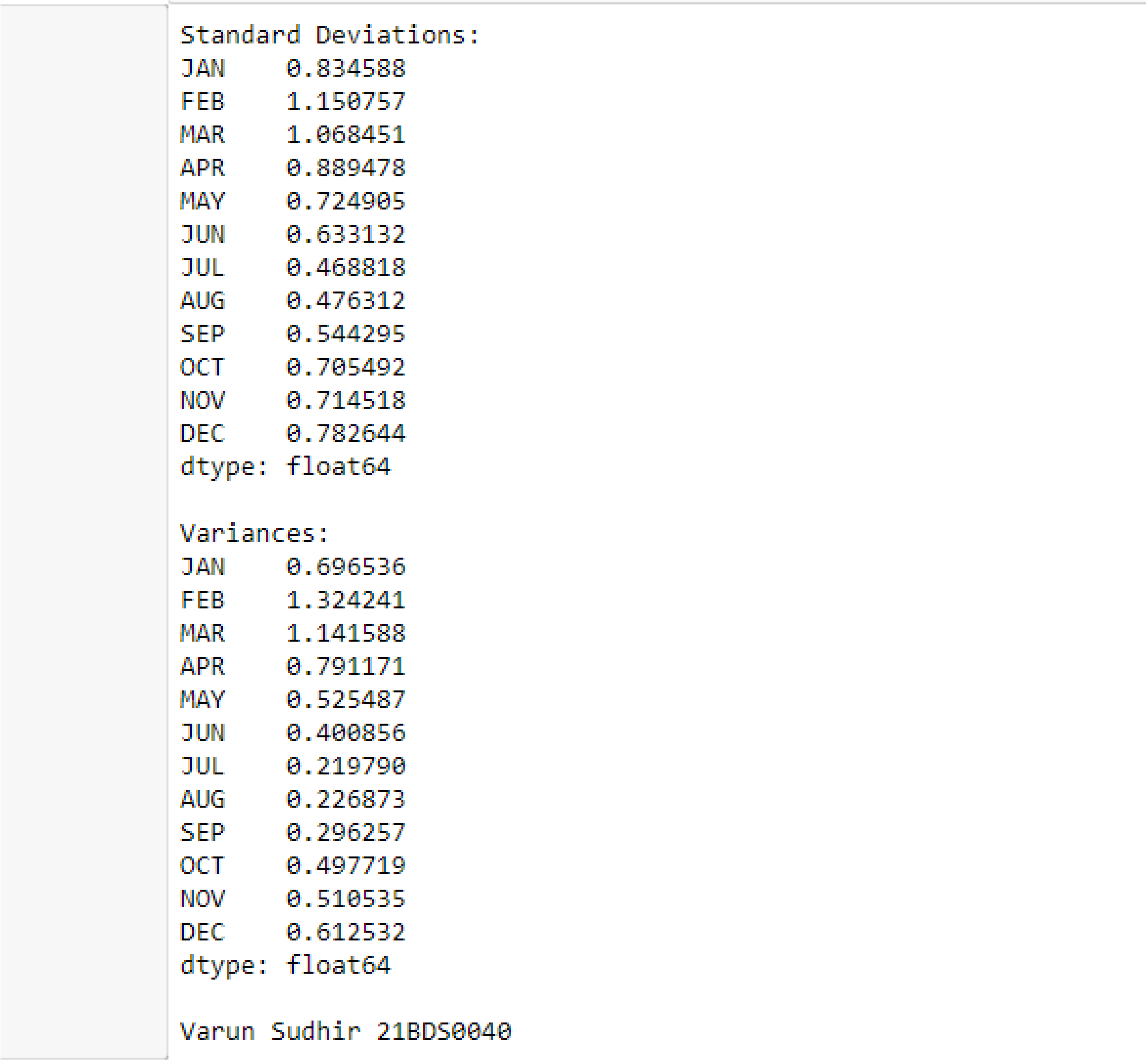
d) Compute the standard deviation and variance for the temperatures of each month across all years.

Code:



```
In [13]: # Varun Sudhir 21BDS0040
         # Calculate the standard deviation and variance for the temperatures of each month across the years

         # Compute standard deviation for each month
         std_devs = df.loc[:, 'JAN':'DEC'].std()

         # Compute variance for each month
         variances = df.loc[:, 'JAN':'DEC'].var()

         print("Standard Deviations:")
         print(std_devs)
         print("\nVariances:")
         print(variances)
         print("\nVarun Sudhir 21BDS0040")
```

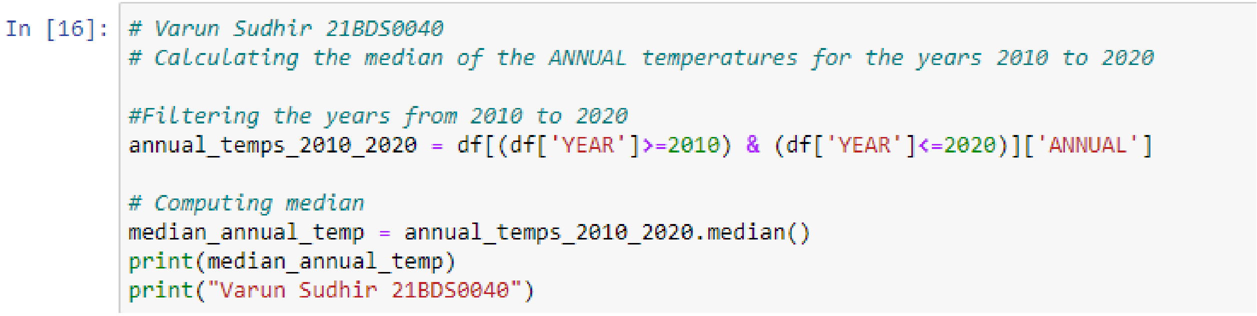
Output:



```
Standard Deviations:
JAN    0.834588
FEB    1.150757
MAR    1.068451
APR    0.889478
MAY    0.724905
JUN    0.633132
JUL    0.468818
AUG    0.476312
SEP    0.544295
OCT    0.705492
NOV    0.714518
DEC    0.782644
dtype: float64

Variances:
JAN    0.696536
FEB    1.324241
MAR    1.141588
APR    0.791171
MAY    0.525487
JUN    0.400856
JUL    0.219790
AUG    0.226873
SEP    0.296257
OCT    0.497719
NOV    0.510535
DEC    0.612532
dtype: float64

Varun Sudhir 21BDS0040
```

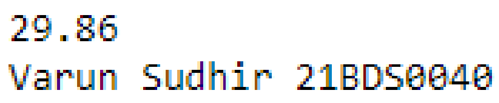
e) Display the median of the ANNUAL temperatures for the years 2010 to 2020

Code:



```
In [16]: # Varun Sudhir 21BDS0040
         # Calculating the median of the ANNUAL temperatures for the years 2010 to 2020

         #Filtering the years from 2010 to 2020
         annual_temps_2010_2020 = df[(df['YEAR']>=2010) & (df['YEAR']<=2020)]['ANNUAL']

         # Computing median
         median_annual_temp = annual_temps_2010_2020.median()
         print(median_annual_temp)
         print("Varun Sudhir 21BDS0040")
```

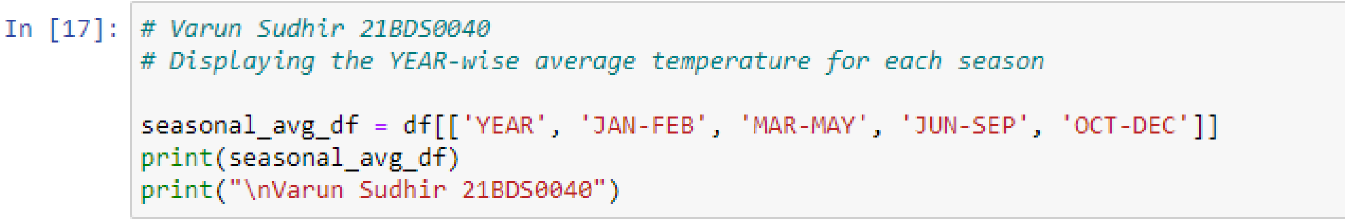
Output:



```
29.86
Varun Sudhir 21BDS0040
```

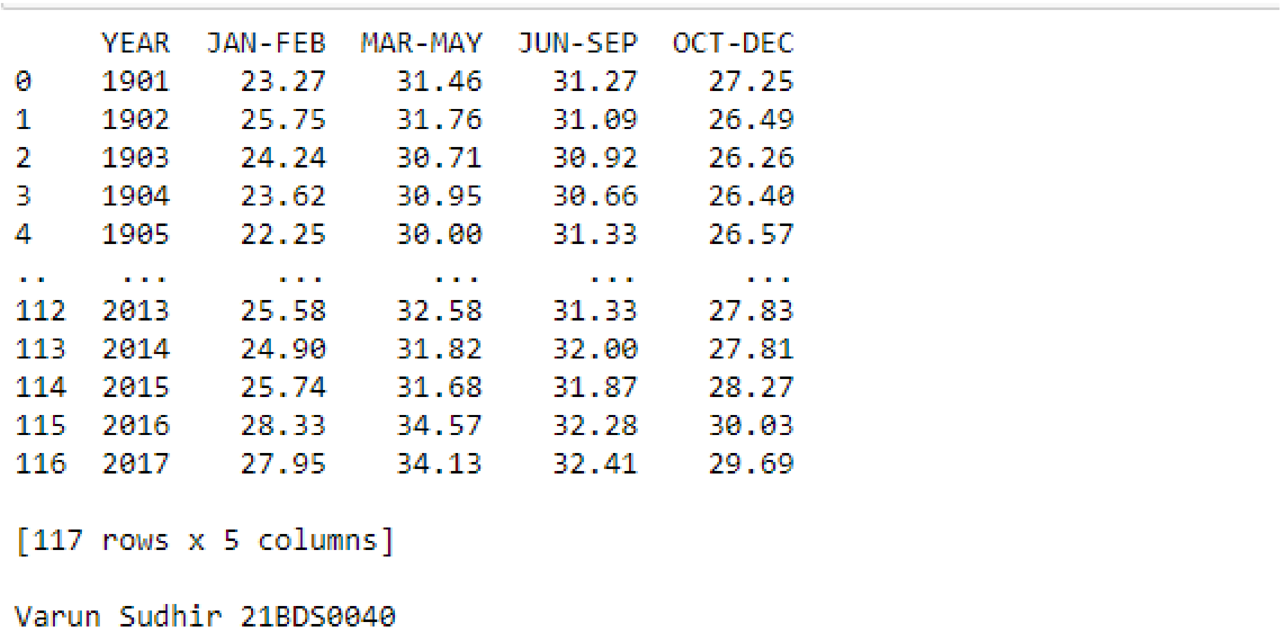
f) Display the YEAR-wise average temperature for each season (JAN-FEB, MAR-MAY, JUN-SEP, OCT-DEC)

Code:



```
In [17]: # Varun Sudhir 21BDS0040
         # Displaying the YEAR-wise average temperature for each season

         seasonal_avg_df = df[['YEAR', 'JAN-FEB', 'MAR-MAY', 'JUN-SEP', 'OCT-DEC']]
         print(seasonal_avg_df)
         print("\nVarun Sudhir 21BDS0040")
```

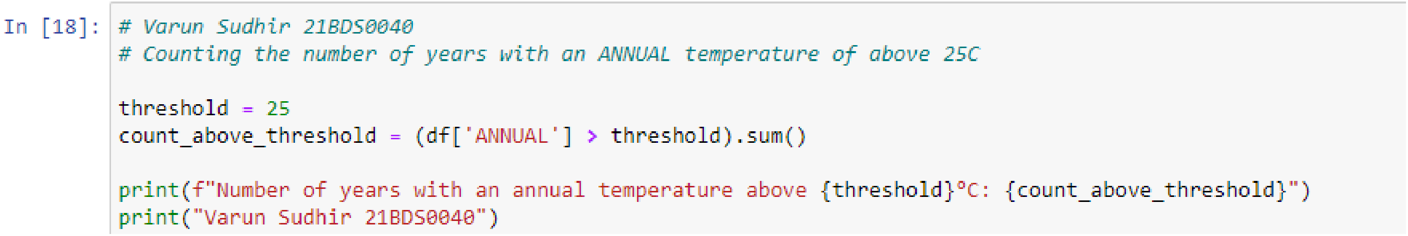
Output:



```
     YEAR  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0    1901    23.27    31.46    31.27    27.25
1    1902    25.75    31.76    31.09    26.49
2    1903    24.24    30.71    30.92    26.26
3    1904    23.62    30.95    30.66    26.40
4    1905    22.25    30.00    31.33    26.57
..    ...      ...      ...      ...      ...
112  2013    25.58    32.58    31.33    27.83
113  2014    24.90    31.82    32.00    27.81
114  2015    25.74    31.68    31.87    28.27
115  2016    28.33    34.57    32.28    30.03
116  2017    27.95    34.13    32.41    29.69

[117 rows x 5 columns]

Varun Sudhir 21BDS0040
```

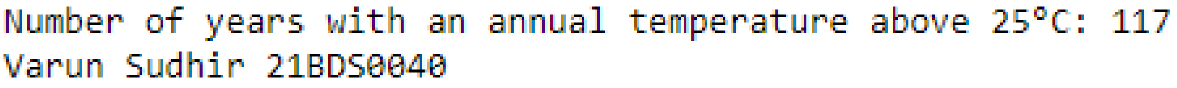
g) Count the number of years with an ANNUAL temperature above a certain threshold (e.g., 25°C).

Code:



```
In [18]: # Varun Sudhir 21BDS0040
         # Counting the number of years with an ANNUAL temperature of above 25C

         threshold = 25
         count_above_threshold = (df['ANNUAL'] > threshold).sum()

         print(f"Number of years with an annual temperature above {threshold}°C: {count_above_threshold}")
         print("Varun Sudhir 21BDS0040")
```

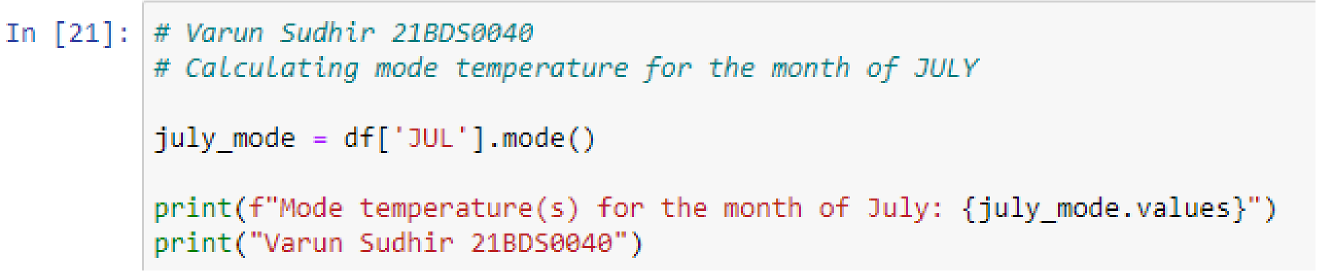
Output:



```
Number of years with an annual temperature above 25°C: 117
Varun Sudhir 21BDS0040
```

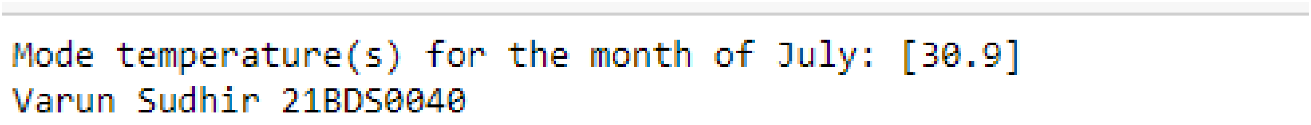
h) Print the mode temperature for the month of JULY

Code:



```
In [21]: # Varun Sudhir 21BDS0040
         # Calculating mode temperature for the month of JULY

         july_mode = df['JUL'].mode()

         print(f"Mode temperature(s) for the month of July: {july_mode.values}")
         print("Varun Sudhir 21BDS0040")
```

Output:



```
Mode temperature(s) for the month of July: [30.9]
Varun Sudhir 21BDS0040
```

i) Plot a histogram with the following information. Collect the minimum and maximum temperature data for each month from the temperature dataset.

Create a DataFrame with the following columns:

YEAR

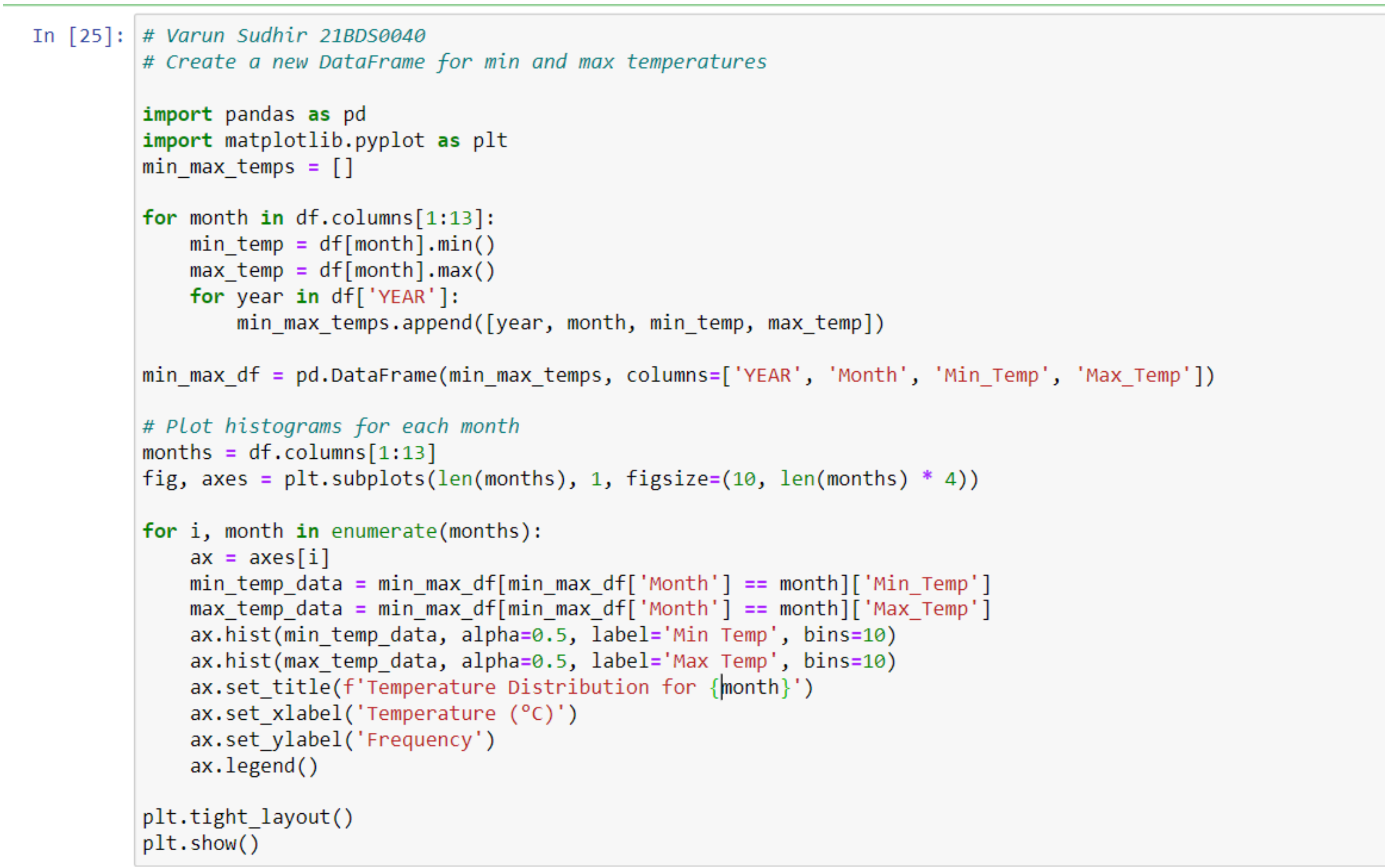
MONTH

MIN_TEMP

MAX_TEMP

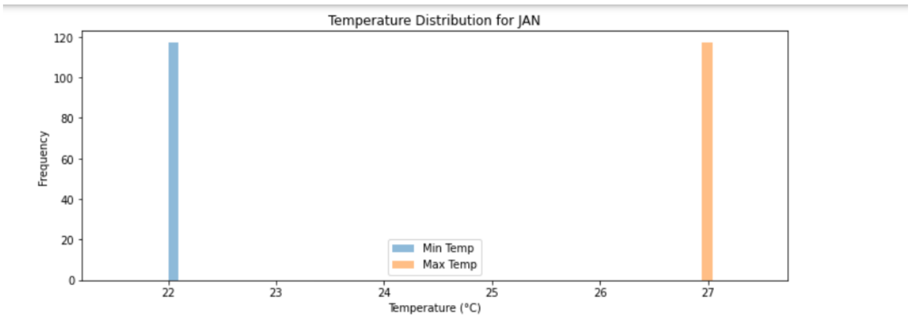
Plot a histogram for each month showing the distribution of minimum and maximum temperatures.

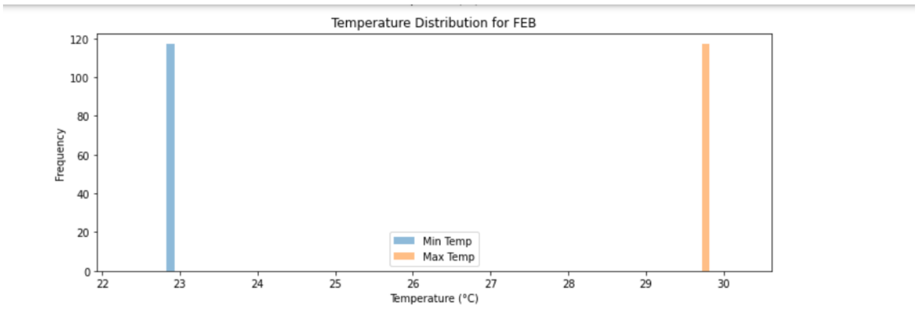
Code:

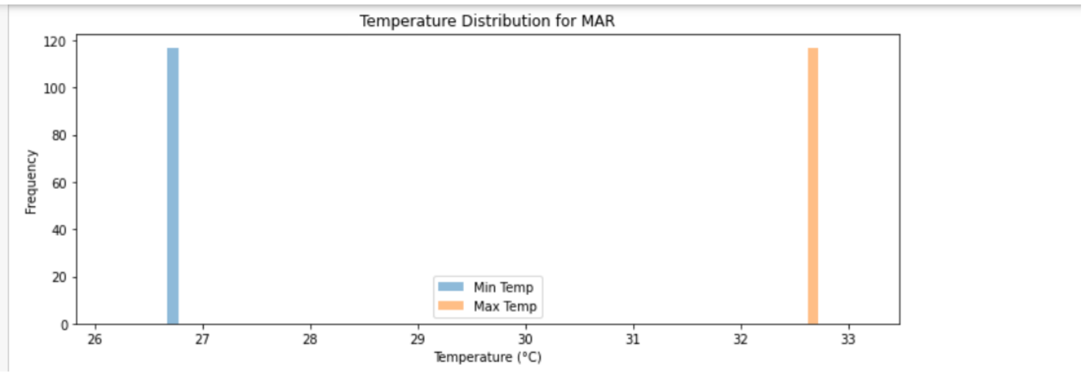


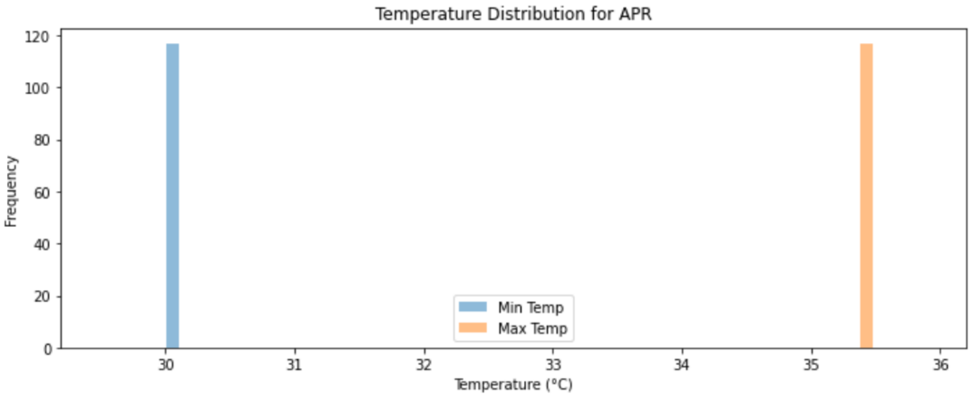
```
In [25]: # Varun Sudhir 21BDS0040
         # Create a new DataFrame for min and max temperatures

         import pandas as pd
         import matplotlib.pyplot as plt
         min_max_temps = []

         for month in df.columns[1:13]:
             min_temp = df[month].min()
             max_temp = df[month].max()
             for year in df['YEAR']:
                 min_max_temps.append([year, month, min_temp, max_temp])

         min_max_df = pd.DataFrame(min_max_temps, columns=['YEAR', 'Month', 'Min_Temp', 'Max_Temp'])

         # Plot histograms for each month
         months = df.columns[1:13]
         fig, axes = plt.subplots(len(months), 1, figsize=(10, len(months) * 4))

         for i, month in enumerate(months):
             ax = axes[i]
             min_temp_data = min_max_df[min_max_df['Month'] == month]['Min_Temp']
             max_temp_data = min_max_df[min_max_df['Month'] == month]['Max_Temp']
             ax.hist(min_temp_data, alpha=0.5, label='Min Temp', bins=10)
             ax.hist(max_temp_data, alpha=0.5, label='Max Temp', bins=10)
             ax.set_title(f'Temperature Distribution for {month}')
             ax.set_xlabel('Temperature (°C)')
             ax.set_ylabel('Frequency')
             ax.legend()

         plt.tight_layout()
         plt.show()
```

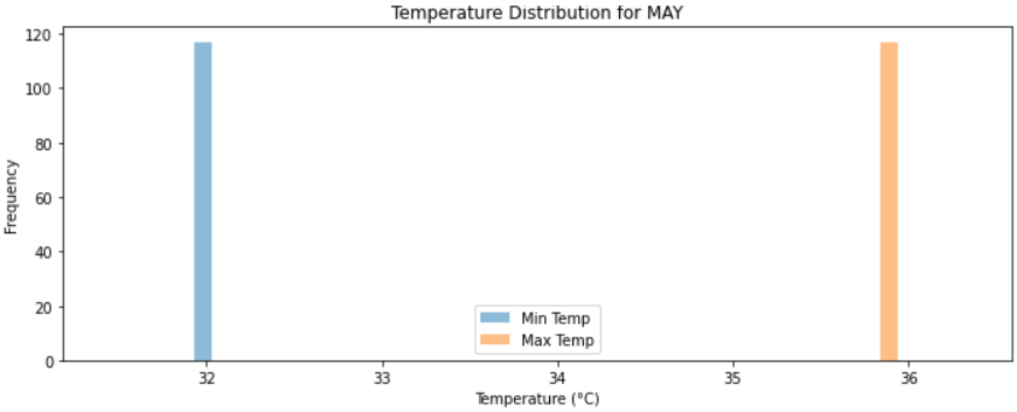
Output:

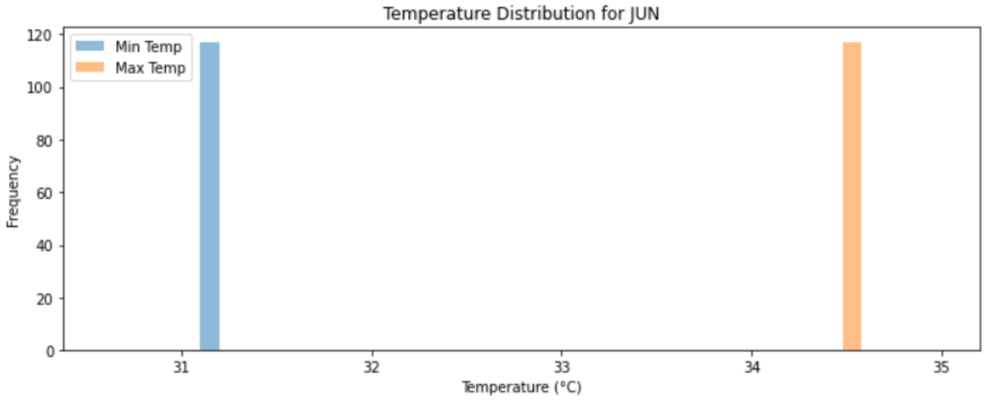


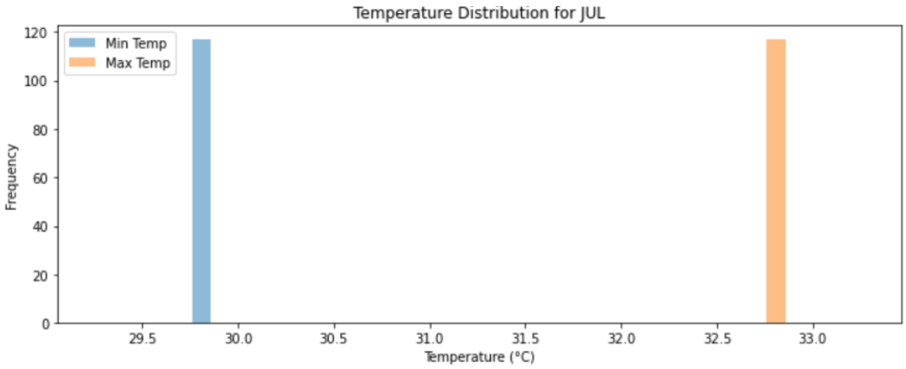


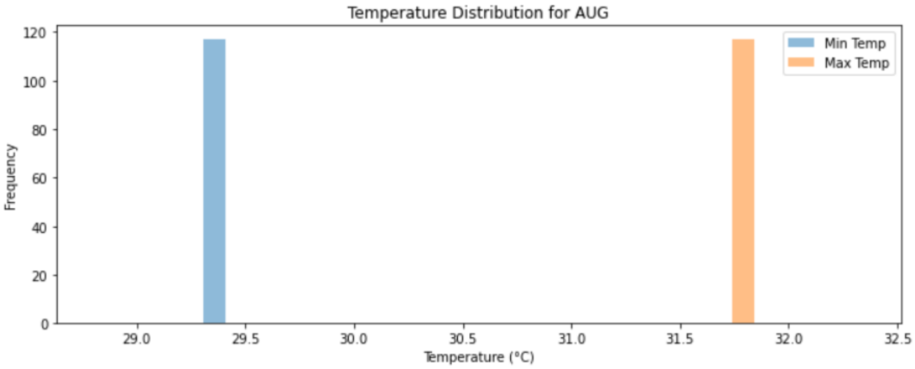

Temperature Distribution for MAR
Frequency
0
20
40
60
80
100
120
Min Temp
Max Temp
26
27
28
29
30
31
32
33
Temperature (°C)

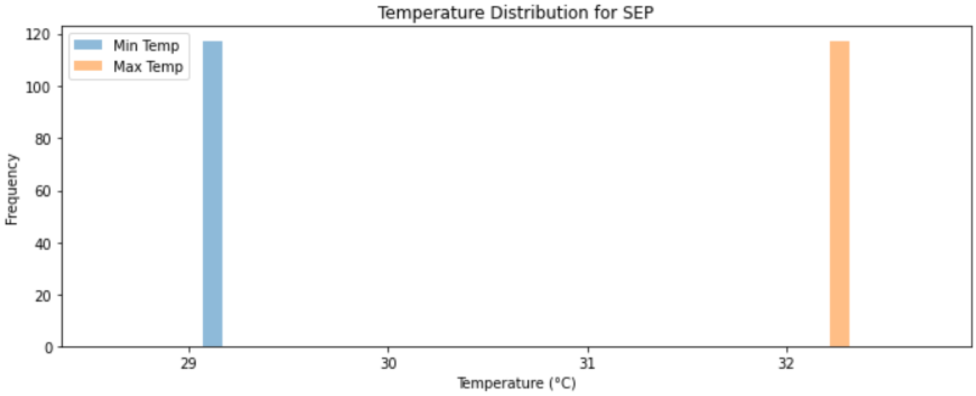

Temperature Distribution for APR
Frequency
0
20
40
60
80
100
120
Min Temp
Max Temp
30
31
32
33
34
35
36
Temperature (°C)

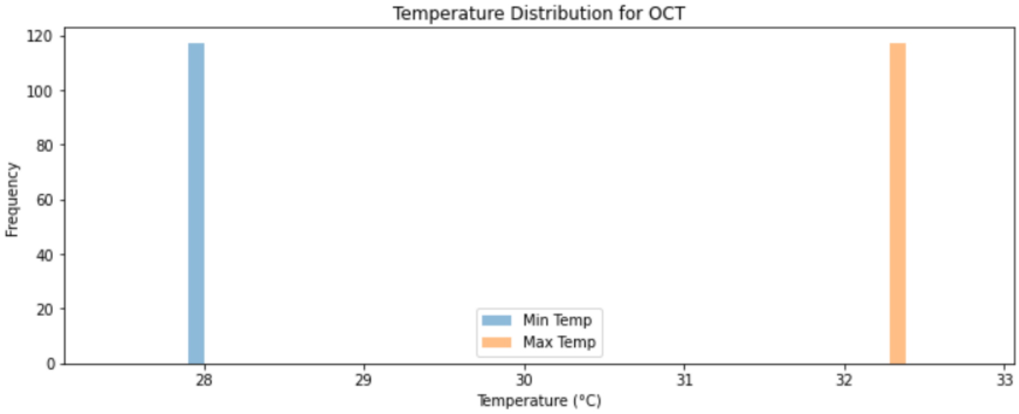

Temperature Distribution for MAY
Frequency
0
20
40
60
80
100
120
Min Temp
Max Temp
32
33
34
35
36
Temperature (°C)

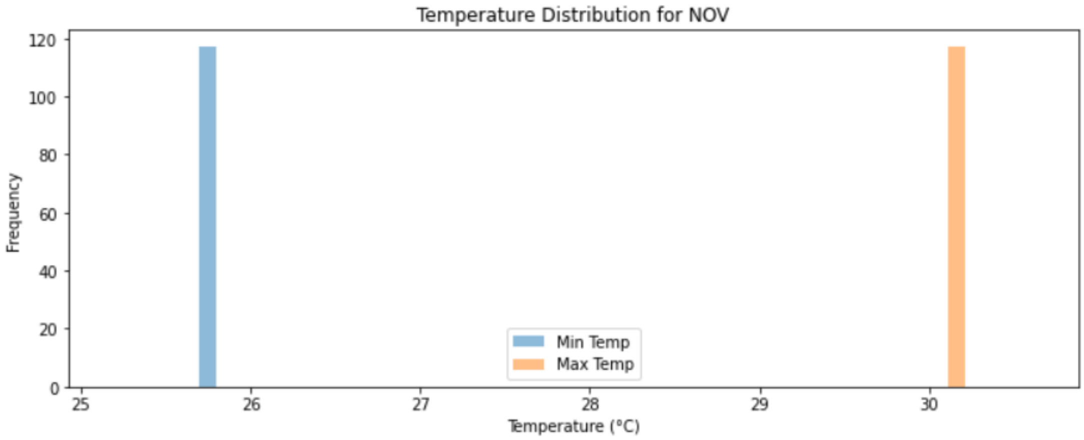

Temperature Distribution for JUN
Frequency
0
20
40
60
80
100
120
Min Temp
Max Temp
31
32
33
34
35
Temperature (°C)


Temperature Distribution for JUL
Min Temp
Max Temp
Frequency
0
20
40
60
80
100
120
29.5
30.0
30.5
31.0
31.5
32.0
32.5
33.0
Temperature (°C)

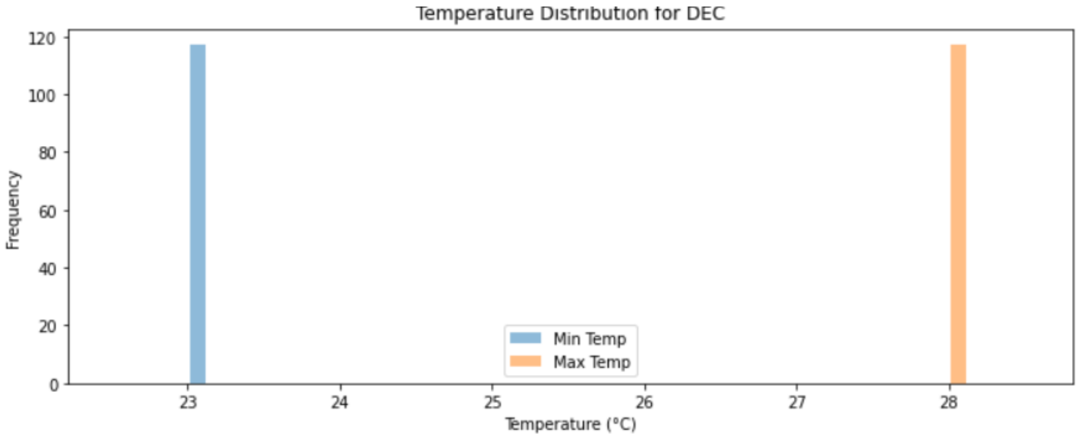

Temperature Distribution for AUG
Min Temp
Max Temp
Frequency
0
20
40
60
80
100
120
29.0
29.5
30.0
30.5
31.0
31.5
32.0
32.5
Temperature (°C)


Temperature Distribution for SEP
Min Temp
Max Temp
Frequency
0
20
40
60
80
100
120
29
30
31
32
Temperature (°C)


Temperature Distribution for OCT
Min Temp
Max Temp
Frequency
0
20
40
60
80
100
120
28
29
30
31
32
33
Temperature (°C)



Temperature Distribution for NOV
Frequency
0
20
40
60
80
100
120
25
26
27
28
29
30
Temperature (°C)
Min Temp
Max Temp



Temperature Distribution for DEC
Frequency
0
20
40
60
80
100
120
23
24
25
26
27
28
Temperature (°C)
Min Temp
Max Temp

j. Plot a heatmap to visualize the temperature data:

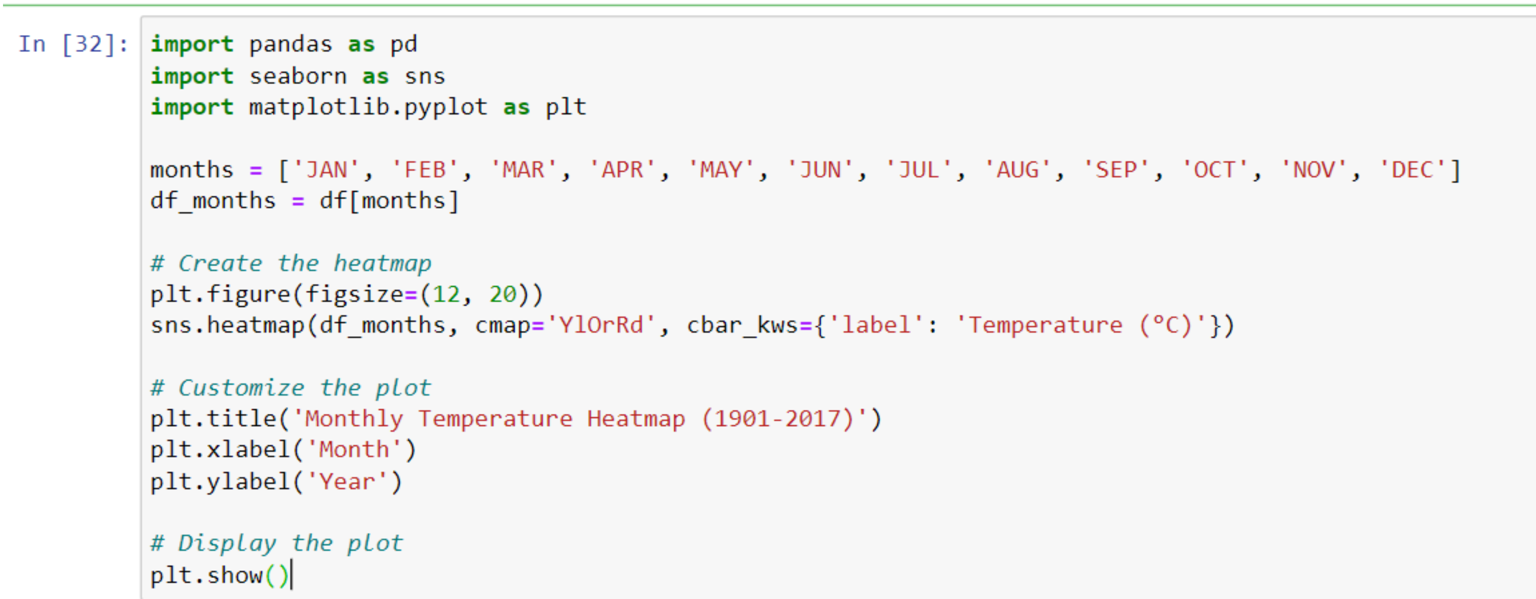
X-axis should represent the months (JAN, FEB, MAR, etc.).

Y-axis should represent the years.

The heatmap should display the temperature values.

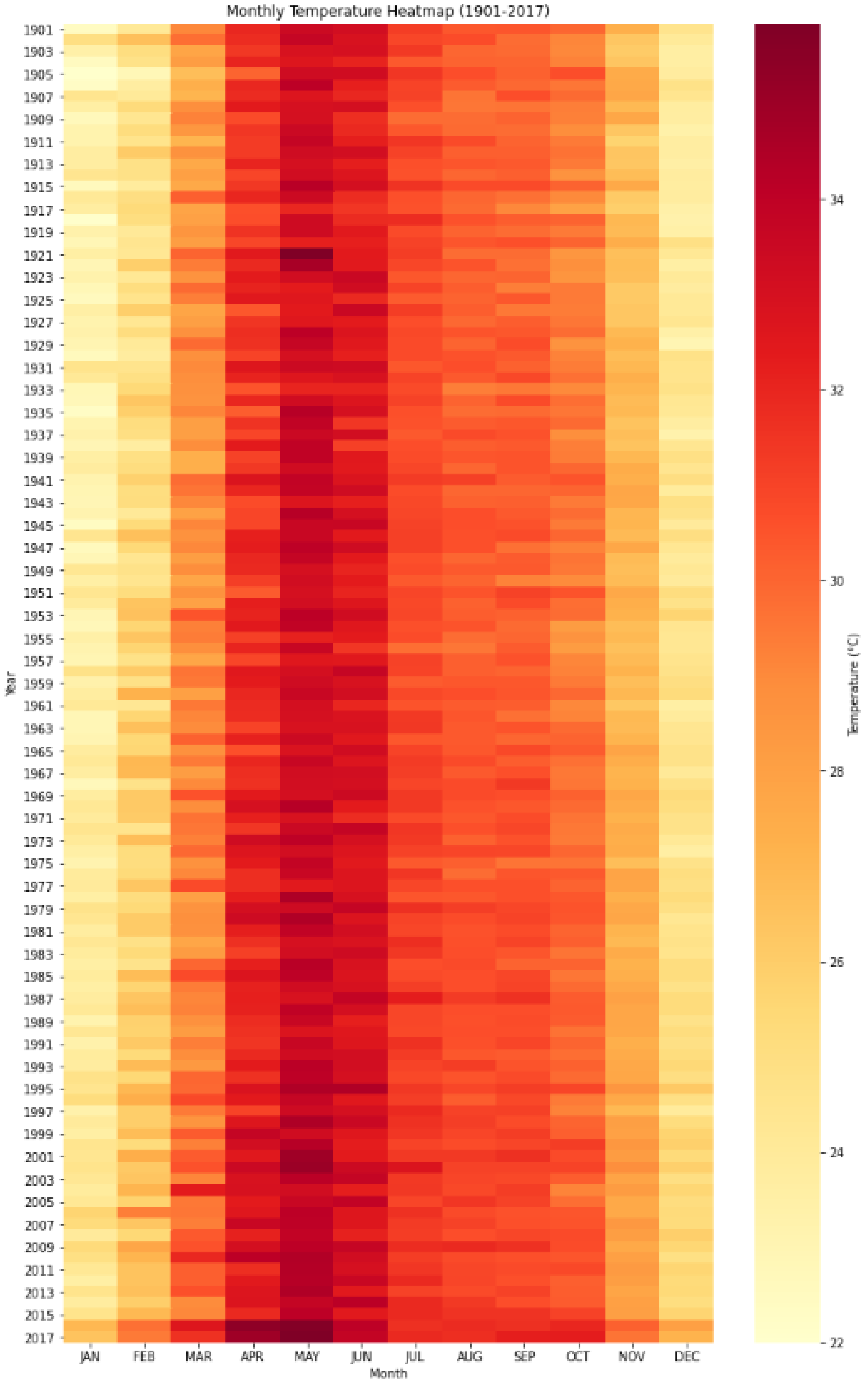
Use different colors to represent different temperature ranges

Code:



```
In [32]: import pandas as pd
         import seaborn as sns
         import matplotlib.pyplot as plt

         months = ['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC']
         df_months = df[months]

         # Create the heatmap
         plt.figure(figsize=(12, 20))
         sns.heatmap(df_months, cmap='YlOrRd', cbar_kws={'label': 'Temperature (°C)'})

         # Customize the plot
         plt.title('Monthly Temperature Heatmap (1901-2017)')
         plt.xlabel('Month')
         plt.ylabel('Year')

         # Display the plot
         plt.show()
```

Output:


Monthly Temperature Heatmap (1901-2017)
Year
1901
1903
1905
1907
1909
1911
1913
1915
1917
1919
1921
1923
1925
1927
1929
1931
1933
1935
1937
1939
1941
1943
1945
1947
1949
1951
1953
1955
1957
1959
1961
1963
1965
1967
1969
1971
1973
1975
1977
1979
1981
1983
1985
1987
1989
1991
1993
1995
1997
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
Month
34
32
30
28
26
24
22
Temperature (°C)