Question 4: Data Visualization (10 points)

Your task is to design a visualization that you believe effectively communicates the data and provide a short write-up describing your design. You can use any data visualization tool you want.

Start with the dataset weather.csv which contains weather measurements nearest to Cornell Tech every day from 1950 to the present.

Notice that the temperature (Ktemp) is in Kelvin rather than in Fahrenheit.

Define a variable that expresses the temperature in Fahrenheit, using the following formula:

Ftemp = (Ktemp − 273.15) * (9/5) + 32

Data Loading and Preprocessing

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider

# Read the weather data
weather_data = pd.read_csv('/content/weather.csv')
# Convert temperature from Kelvin to Fahrenheit
weather_data['Ftemp'] = (weather_data['Ktemp'] - 273.15) * (9/5) + 32

# Extract month and year from the Date column and add it to the weather_data dataframe
weather_data['Month'] = pd.to_datetime(weather_data['time']).dt.month
weather_data['Year'] = pd.to_datetime(weather_data['time']).dt.year
```

PART A

For every month of the year, plot the average temperature (in Fahrenheit) using a scatter plot or line plot. Your visual should be configurable so that a user can easily look at a specific year's weather curve (use a sliding scale filter). (6 points)

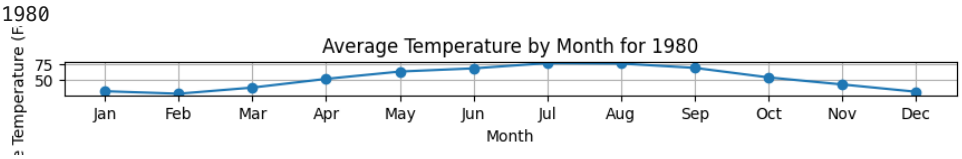
```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider

# Load weather data and preprocess it
# Assuming monthly_avg_temp is already defined

def update_plot_helper(val):
    year = int(slider.val)
    plt.cla()
    if year in monthly_avg_temp.index:
        plt.plot(monthly_avg_temp.loc[year], marker='o')
        plt.title(f'Average Temperature by Month for {year}')
        plt.xlabel('Month')
        plt.ylabel('Average Temperature (Fahrenheit)')
        plt.xticks(range(1, 13), ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
        plt.grid(True)
    else:
        plt.text(0.5, 0.5, f'Data for the year {year} is not available', ha='center', va='center', fontsize=12)
    plt.show()

# User can use this function to create the plot
def create_plot(year):
  plt.figure(figsize=(10, 6))
  ax_slider = plt.axes([0.1, 0.02, 0.8, 0.05])
  slider = Slider(ax_slider, 'Year', 1950, 2021, valinit=year, valstep=1)
  slider.on_changed(update_plot_helper)
  update_plot_helper(year)
  plt.show()

create_plot(1980) # example of how to use this function
```



PART B

Based on all of the data, when is the first year where the year's average temperature passes 55 degrees (when will Cornell Tech finally be warm?) (2 points)

```
yearly_avg_temp = weather_data.groupby('Year')['Ftemp'].mean()
first_warm_year = yearly_avg_temp[yearly_avg_temp > 55].index.min()

print(f"The first year where the average temperature exceeds 55 degrees is: {first_warm_year}")
```

```
The first year where the average temperature exceeds 55 degrees is: 1953
```

PART C

Create a new sheet where you do something creative through data visualization. Express something about the temperature over time(e.g. Look in the cycle of temperature over seasons, etc) using this dataset, or add a new dataset(any available dataset online is fine) and find some correlation with the temperature(e.g. Number of some kind of fish in the ocean, etc, does the number of it go up and down following the temperature trend? Etc.). (2 points)

You will be graded based on the quality of completion. A simple plot of temperature in Celsius will not get you full points.

```
# Data processing
walmart_store_sales = pd.read_csv('/content/walmart_store_sales.csv')

# Convert the Date column to datetime format and extract month from the Date column
walmart_store_sales['Date'] = pd.to_datetime(walmart_store_sales['Date'])
walmart_store_sales['Month'] = walmart_store_sales['Date'].dt.month

# Group weekly sale by month and store, and calculate the total sales per store
total_monthly_sales_per_store = walmart_store_sales.groupby(['Month', 'Store']).agg({'Weekly_Sales': 'mean'}).reset_ind

# Calculate the average weekly sales per store per month
average_weekly_sales = total_monthly_sales_per_store.groupby('Month')['Weekly_Sales'].mean().reset_index()

# Calculate the average temperature per month
monthly_avg_temp = monthly_avg_temp.groupby('Month')['Ftemp'].mean().reset_index()
```

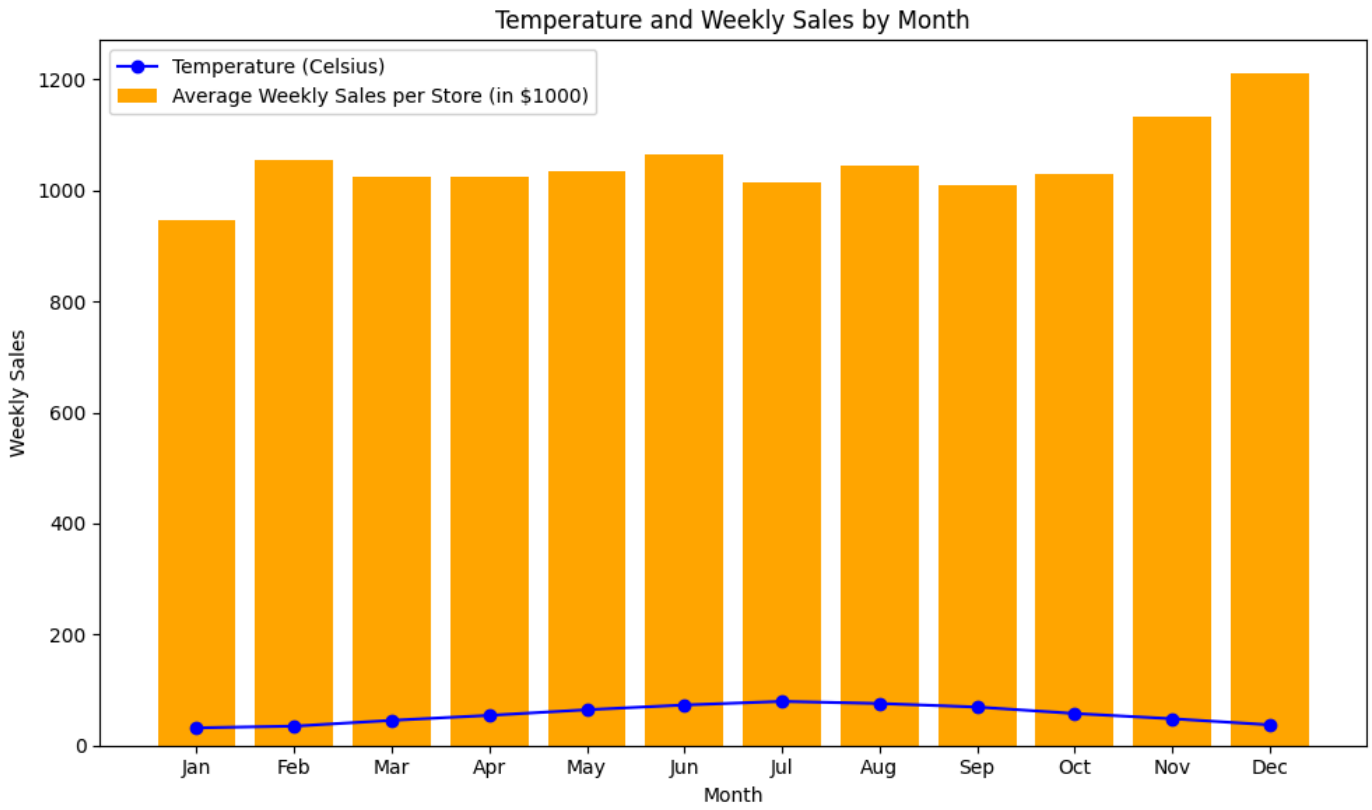
```
<ipython-input-113-8aa5948ede8e>:5: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the defau
  walmart_store_sales['Date'] = pd.to_datetime(walmart_store_sales['Date'])
```

```
from scipy.stats import pearsonr

# Plotting the data
plt.figure(figsize=(10, 6))
plt.plot(monthly_avg_temp['Month'], monthly_avg_temp['Ftemp'], marker='o', color='blue', label='Temperature (Celsius)')
plt.bar(average_weekly_sales['Month'], average_weekly_sales['Weekly_Sales'] / 1000, color='orange', label='Average Week
plt.xlabel('Month')
plt.ylabel('Weekly Sales')
plt.title('Temperature and Weekly Sales by Month')
plt.xticks(range(1, 13), ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
plt.legend()
plt.tight_layout()
plt.show()

# Calculate the correlation coefficient to draw conclusion on the relationship between temperature and Week
correlation_coefficient, p_value = pearsonr(average_weekly_sales['Weekly_Sales'], monthly_avg_temp['Ftemp'])
```

```
print("Correlation Coefficient:", correlation_coefficient)
print("p-value:", p_value)
```




```
Correlation Coefficient: -0.19456559761122155
p-value: 0.5445453761536656
```

The cofficient between the average weekly sales and the average temperature per month is approximately -0.195, which means there is a weak negative correlation. In another words, when the temperature is high, the weekly sales is low. When the temperature is low, the weekly sales is high.

However, with a p-value of 0.545, the correlation is not statistically significant. This means that there is likely no meaningful relationship between weekly sales and temperature.