## Original Code Flow

1. **Importing Libraries**: The code imports necessary libraries: csv for reading CSV files, datetime for date manipulation, and matplotlib.pyplot for plotting.
2. **Reading the CSV File**: The program opens the CSV file sitka\_weather\_2018\_simple.csv and initializes a CSV reader.
3. **Extracting Data**: It skips the header row and iterates through the remaining rows to extract dates and high temperatures, which are stored in two separate lists.
4. **Plotting**: Finally, it plots the high temperatures against the dates using a red line and formats the plot for better readability.

import csv

from datetime import datetime

from matplotlib import pyplot as plt

filename = 'sitka\_weather\_2018\_simple.csv'

with open(filename) as f:

reader = csv.reader(f)

header\_row = next(reader)

# Get dates and high temperatures from this file.

dates, highs = [], []

for row in reader:

current\_date = datetime.strptime(row[2], '%Y-%m-%d')

dates.append(current\_date)

high = int(row[5])

highs.append(high)

# Plot the high temperatures.

#plt.style.use('seaborn')

fig, ax = plt.subplots()

ax.plot(dates, highs, c='red')

# Format plot.

plt.title("Daily high temperatures - 2018", fontsize=24)

plt.xlabel('', fontsize=16)

fig.autofmt\_xdate()

plt.ylabel("Temperature (F)", fontsize=16)

plt.tick\_params(axis='both', which='major', labelsize=16)

plt.show()



## Modify code flow

To implement the required changes, we will follow these steps:

1. **User Menu**: Introduce a menu that allows users to choose between viewing high temperatures, low temperatures, or exiting the program.
2. **Graphing Low Temperatures**: Modify the code to extract and plot low temperatures when the user selects that option.
3. **Looping Mechanism**: Implement a loop that continues to prompt the user until they choose to exit.
4. **Exit Message**: Provide a friendly exit message when the user decides to leave the program.

# Program modify by Steve Stylin

# Bellevue University, Module 4.2 Assignment: High/Low Temperatures

print("\nProgram modify by Steve Stylin\n Bellevue University,\nModule 4.2 Assignment: High/Low Temperatures ")

import csv

from datetime import datetime

from matplotlib import pyplot as plt

import sys

def main():

filename = 'sitka\_weather\_2018\_simple.csv'

while True:

print("\nMenu:")

print("1. View High Temperatures")

print("2. View Low Temperatures")

print("3. Exit")

choice = input("Please select an option (1, 2, or 3): ")

if choice == '1':

plot\_temperatures(filename, 'high')

elif choice == '2':

plot\_temperatures(filename, 'low')

elif choice == '3':

print("Thank you for using the program. Goodbye!")

sys.exit()

else:

print("Invalid choice. Please try again.")

def plot\_temperatures(filename, temp\_type):

with open(filename) as f:

reader = csv.reader(f)

header\_row = next(reader)

# Get dates and temperatures from this file.

dates, highs, lows = [], [], []

for row in reader:

current\_date = datetime.strptime(row[2], '%Y-%m-%d')

dates.append(current\_date)

high = int(row[5])

low = int(row[6]) # Assuming low temperatures are in the 6th column

highs.append(high)

lows.append(low)

# Plot the selected temperatures.

fig, ax = plt.subplots()

if temp\_type == 'high':

ax.plot(dates, highs, c='red', label='High Temperatures')

plt.title("Daily High Temperatures - 2018", fontsize=24)

else:

ax.plot(dates, lows, c='blue', label='Low Temperatures')

plt.title("Daily Low Temperatures - 2018", fontsize=24)

# Format plot.

plt.xlabel('', fontsize=16)

fig.autofmt\_xdate()

plt.ylabel("Temperature (F)", fontsize=16)

plt.tick\_params(axis='both', which='major', labelsize=16)

plt.legend()

plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

main()



## Changes Explanation

1. **Menu Implementation**: The program now begins with a menu that prompts the user for input.
2. **Temperature Selection**: Depending on the user's choice, the program either plots high or low temperatures.
3. **Looping**: The while True loop ensures that the menu is displayed repeatedly until the user opts to exit.
4. **Exit Handling**: The sys.exit() function gracefully terminates the program and displays a friendly exit message.

The modifications made to the original program enhance its interactivity and usability. By allowing users to choose between high and low temperatures and implementing a looping mechanism, the program becomes more user-friendly. The use of the sys module to exit the program adds a layer of professionalism to the code. This revised version, sitka\_high\_low.py, is now ready for use in the specified module-4 directory.