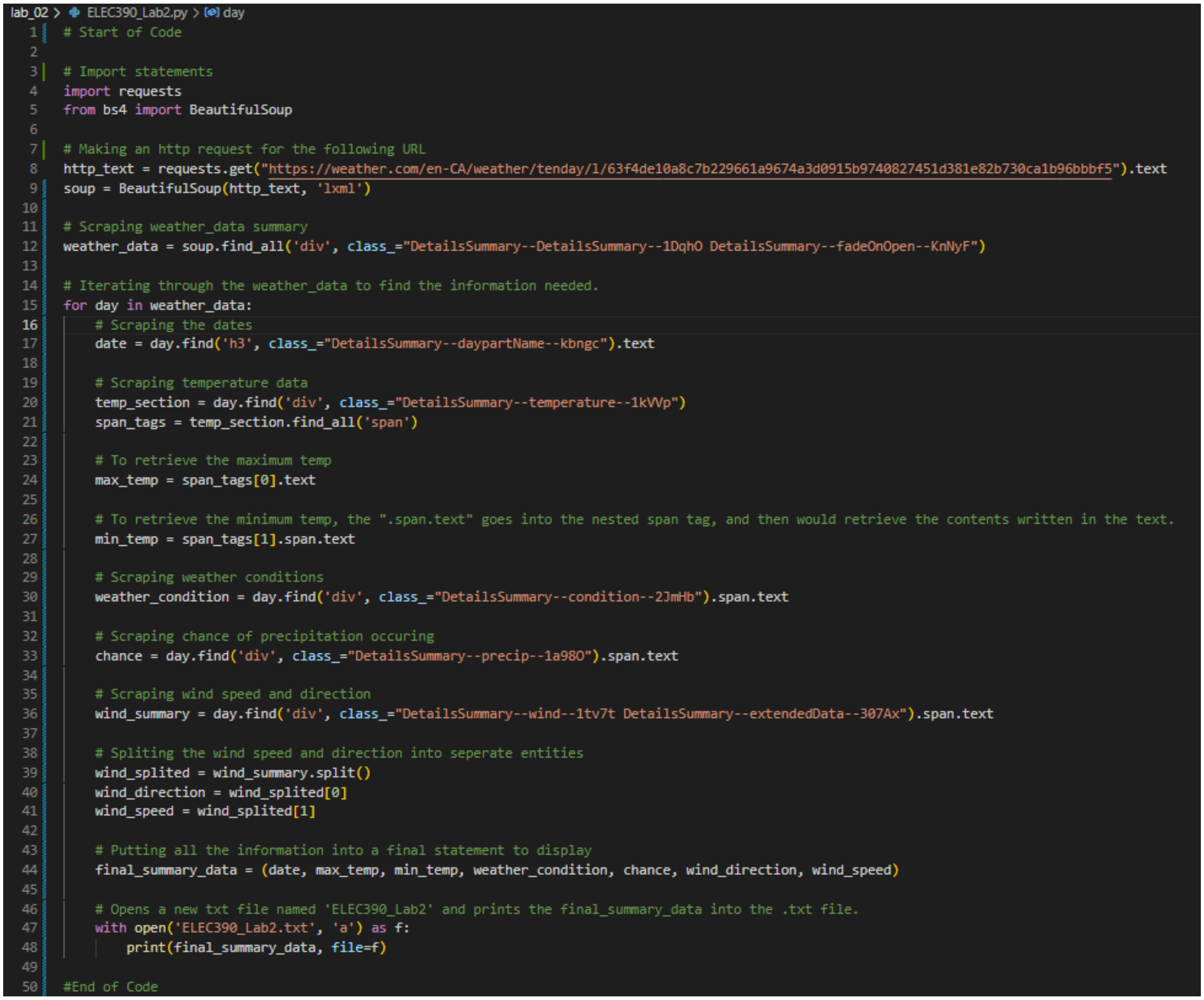
ELEC 390 – Lab 02

Department of Electrical and Computer Engineering
Queen’s University

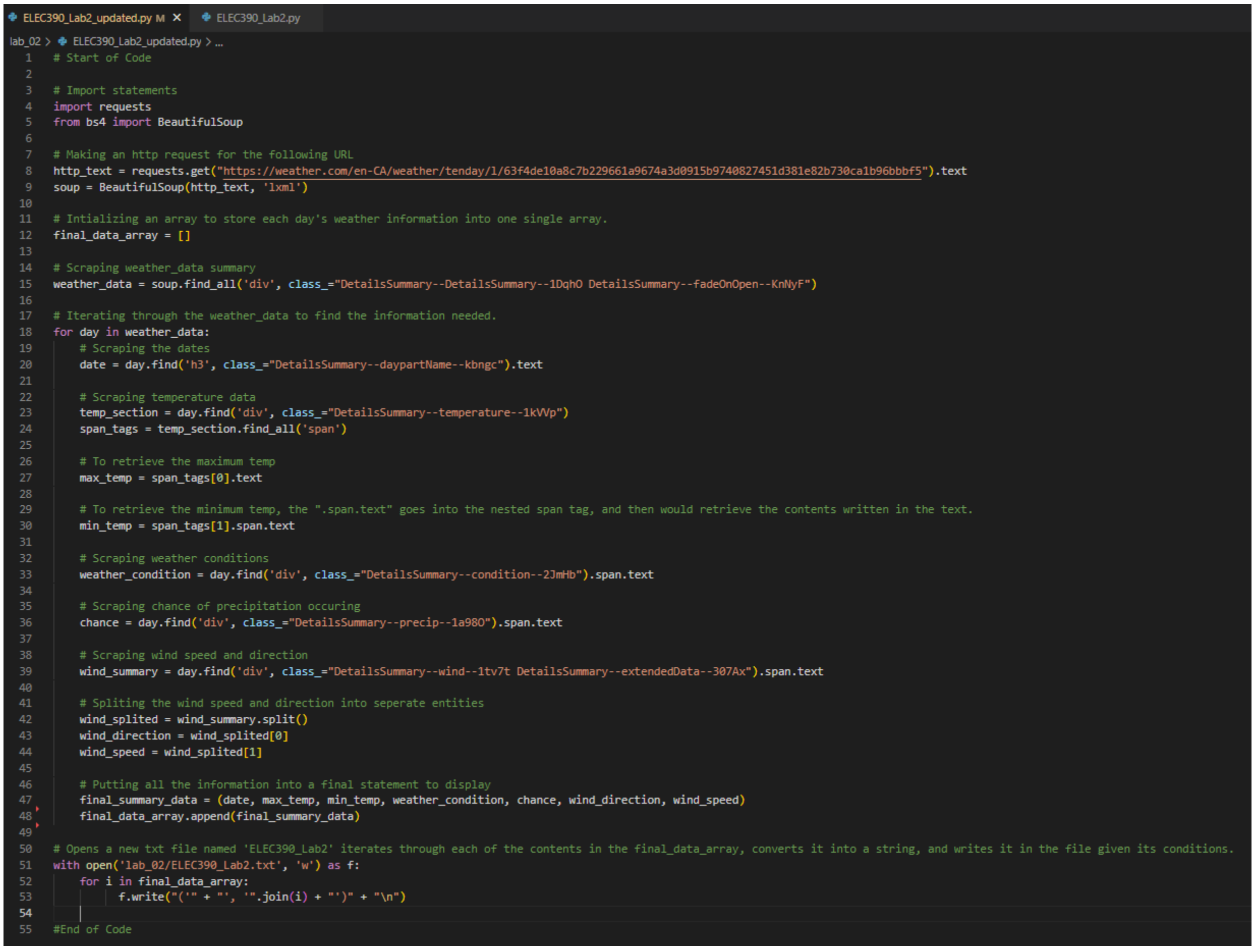
Composed By
Nicholas Seegobin (20246787)
Zeerak Asim (20237955)
Lauren Steel (20218337)
Saman Saeidi (20217992)

Section 03
Date of Submission
2023 February 2nd



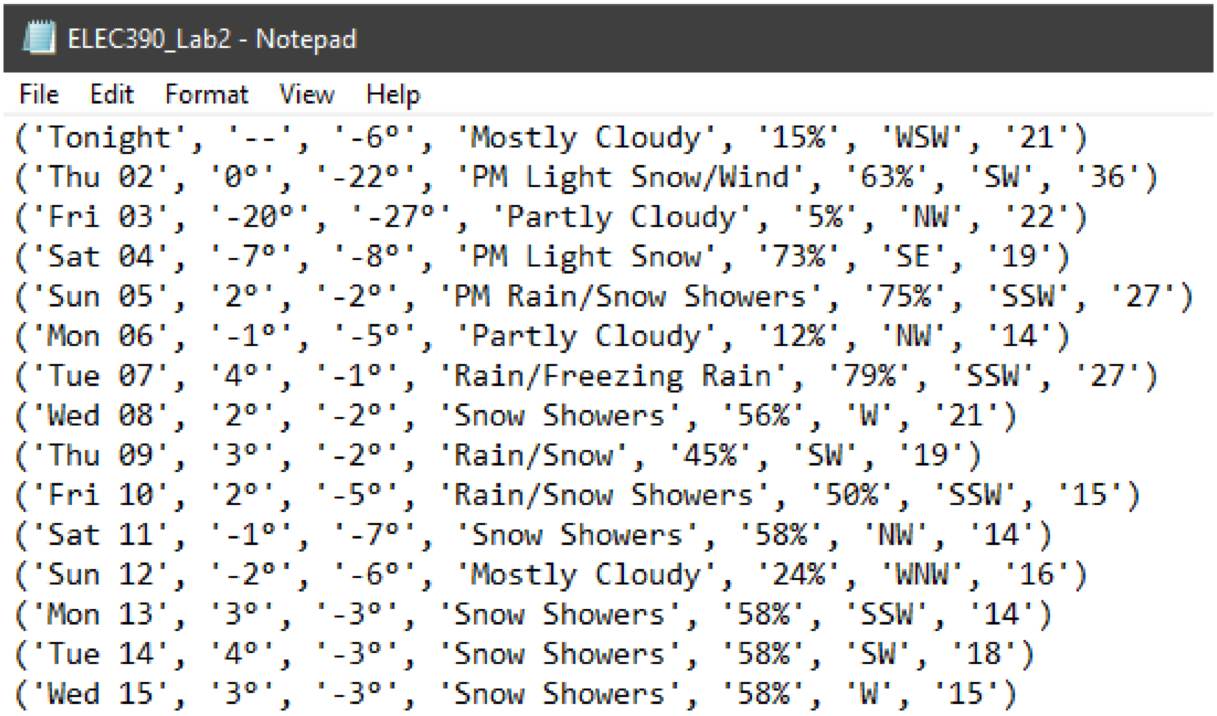
```python
# Start of Code

# Import statements
import requests
from bs4 import BeautifulSoup

# Making an http request for the following URL
http_text = requests.get("https://weather.com/en-CA/weather/tenday/l/63f4de10a8c7b229661a9674a3d0915b9740827451d381e82b730ca1b96bbbf5").text
soup = BeautifulSoup(http_text, 'lxml')

# Scraping weather_data summary
weather_data = soup.find_all('div', class_="DetailsSummary--DetailsSummary--1DqhO DetailsSummary--fadeOnOpen--KnNyF")

# Iterating through the weather_data to find the information needed.
for day in weather_data:
    # Scraping the dates
    date = day.find('h3', class_="DetailsSummary--daypartName--kbngc").text

    # Scraping temperature data
    temp_section = day.find('div', class_="DetailsSummary--temperature--1kVVp")
    span_tags = temp_section.find_all('span')

    # To retrieve the maximum temp
    max_temp = span_tags[0].text

    # To retrieve the minimum temp, the ".span.text" goes into the nested span tag, and then would retrieve the contents written in the text.
    min_temp = span_tags[1].span.text

    # Scraping weather conditions
    weather_condition = day.find('div', class_="DetailsSummary--condition--2JmHb").span.text

    # Scraping chance of precipitation occuring
    chance = day.find('div', class_="DetailsSummary--precip--1a98O").span.text

    # Scraping wind speed and direction
    wind_summary = day.find('div', class_="DetailsSummary--wind--1tv7t DetailsSummary--extendedData--307Ax").span.text

    # Spliting the wind speed and direction into seperate entities
    wind_splited = wind_summary.split()
    wind_direction = wind_splited[0]
    wind_speed = wind_splited[1]

    # Putting all the information into a final statement to display
    final_summary_data = (date, max_temp, min_temp, weather_condition, chance, wind_direction, wind_speed)

    # Opens a new txt file named 'ELEC390_Lab2' and prints the final_summary_data into the .txt file.
    with open('ELEC390_Lab2.txt', 'a') as f:
        print(final_summary_data, file=f)

#End of Code
```

Figure 1: Code Snippet of ELEC390_LAB2.py


ELEC390_Lab2_updated.py M
ELEC390_Lab2.py
lab_02 > ELEC390_Lab2_updated.py > ...

```python
# Start of Code

# Import statements
import requests
from bs4 import BeautifulSoup

# Making an http request for the following URL
http_text = requests.get("https://weather.com/en-CA/weather/tenday/l/63f4de10a8c7b229661a9674a3d0915b9740827451d381e82b730ca1b96bbbf5").text
soup = BeautifulSoup(http_text, 'lxml')

# Intializing an array to store each day's weather information into one single array.
final_data_array = []

# Scraping weather_data summary
weather_data = soup.find_all('div', class_="DetailsSummary--DetailsSummary--1DqhO DetailsSummary--fadeOnOpen--KnNyF")

# Iterating through the weather_data to find the information needed.
for day in weather_data:
    # Scraping the dates
    date = day.find('h3', class_="DetailsSummary--daypartName--kbngc").text

    # Scraping temperature data
    temp_section = day.find('div', class_="DetailsSummary--temperature--1kVVp")
    span_tags = temp_section.find_all('span')

    # To retrieve the maximum temp
    max_temp = span_tags[0].text

    # To retrieve the minimum temp, the ".span.text" goes into the nested span tag, and then would retrieve the contents written in the text.
    min_temp = span_tags[1].span.text

    # Scraping weather conditions
    weather_condition = day.find('div', class_="DetailsSummary--condition--2JmHb").span.text

    # Scraping chance of precipitation occuring
    chance = day.find('div', class_="DetailsSummary--precip--1a98O").span.text

    # Scraping wind speed and direction
    wind_summary = day.find('div', class_="DetailsSummary--wind--1tv7t DetailsSummary--extendedData--307Ax").span.text

    # Spliting the wind speed and direction into seperate entities
    wind_splited = wind_summary.split()
    wind_direction = wind_splited[0]
    wind_speed = wind_splited[1]

    # Putting all the information into a final statement to display
    final_summary_data = (date, max_temp, min_temp, weather_condition, chance, wind_direction, wind_speed)
    final_data_array.append(final_summary_data)

# Opens a new txt file named 'ELEC390_Lab2' iterates through each of the contents in the final_data_array, converts it into a string, and writes it in the file given its conditions.
with open('lab_02/ELEC390_Lab2.txt', 'w') as f:
    for i in final_data_array:
            f.write("('" + "', '".join(i) + "')" + "\n")

#End of Code
```

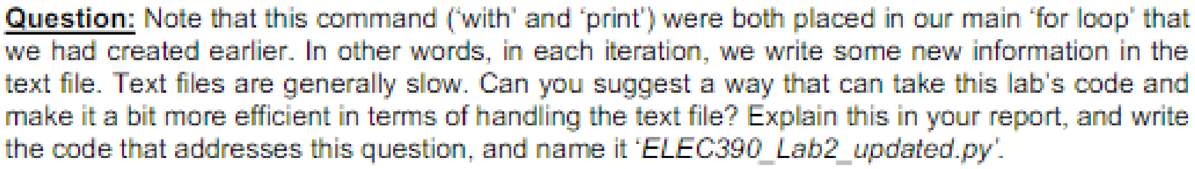
Figure 2: Code Snippet of ELEC390_LAB2_updated.py



```
('Tonight', '--', '-6°', 'Mostly Cloudy', '15%', 'WSW', '21')
('Thu 02', '0°', '-22°', 'PM Light Snow/Wind', '63%', 'SW', '36')
('Fri 03', '-20°', '-27°', 'Partly Cloudy', '5%', 'NW', '22')
('Sat 04', '-7°', '-8°', 'PM Light Snow', '73%', 'SE', '19')
('Sun 05', '2°', '-2°', 'PM Rain/Snow Showers', '75%', 'SSW', '27')
('Mon 06', '-1°', '-5°', 'Partly Cloudy', '12%', 'NW', '14')
('Tue 07', '4°', '-1°', 'Rain/Freezing Rain', '79%', 'SSW', '27')
('Wed 08', '2°', '-2°', 'Snow Showers', '56%', 'W', '21')
('Thu 09', '3°', '-2°', 'Rain/Snow', '45%', 'SW', '19')
('Fri 10', '2°', '-5°', 'Rain/Snow Showers', '50%', 'SSW', '15')
('Sat 11', '-1°', '-7°', 'Snow Showers', '58%', 'NW', '14')
('Sun 12', '-2°', '-6°', 'Mostly Cloudy', '24%', 'WNW', '16')
('Mon 13', '3°', '-3°', 'Snow Showers', '58%', 'SSW', '14')
('Tue 14', '4°', '-3°', 'Snow Showers', '58%', 'SW', '18')
('Wed 15', '3°', '-3°', 'Snow Showers', '58%', 'W', '15')
```

Figure 3: The contents in ELEC390_LAB2.txt

Question



Question: Note that this command ('with' and 'print') were both placed in our main 'for loop' that we had created earlier. In other words, in each iteration, we write some new information in the text file. Text files are generally slow. Can you suggest a way that can take this lab's code and make it a bit more efficient in terms of handling the text file? Explain this in your report, and write the code that addresses this question, and name it '*ELEC390_Lab2_updated.py*'.

Answer:

The original method was not the most efficient since it opened the file and wrote data in every iteration of the for loop. The updated code, 'ELEC390_LAB2_updated.py', is more efficient because it accesses the .txt file only once at the end of the code. It does this by appending the data of each iteration into the array. After appending the data, the array is written into the .txt file with the provided formatting using the f.write() command.