In [6]:
```
#HEADER
#// File: Week 12 Assignment.py
#// Name: Benjamin Bartek
#// Date: February 28, 2019
#// Course: DSC 510 - Introduction to Programming
#// Desc: This program is a simple API program that retrieves weather API data from https://openweathermap.org.
#//       The program allows the user to enter a U.S. zip code, a U.S. city name, or an international city name.
#//       It also allows the user to make a new request until they enter input to make the program quit.
#// Usage: The program 1) displays a welcome message 2) Presents 4 choices: a) enter zip b) entery city c) enter
#//        international city. 3) The program then prompts for entry of the location information and attempts
#//        to retrieve API weather data for that location. If unsuccessful, error messages are displayed.
#//        If successful, the weather data is displayed in a readable format, displaying imperial units for U.S.
#//        locations and metric units for international locations (or U.S. selected from int'l menu). The program
#//        uses functions, loops, if statements, try blocks, error messaging, & main function with script execution.
#//Known Limits: 1) Sunrise/Sunset data is available only in GMT is retrieved and displayed only in GMT. Future versions
#//        will support user entry of time zone. 2) Unable to distinguish between identical city names in multiple
#//        states, which requires further development to lookup location using OpenWeatherMap's list of city data.
#//        3) Wind direction was displayed as an angle. Future versions can use if statements to define N, S, E, W, etc.
#//        It is currently disabled due to Traceback Errors when API data does not return with 'dir'.
#//        4) Future versions should include a GUI.
#//---------------------------------------------------------------------------------------------------------------------

#GLOBAL ITEMS
#Imports
import requests, json, sys, time

#Variables
api_key = '3e0c37d13b9c4ee1d770b4b1c769a0eb'
url = "http://api.openweathermap.org/data/2.5/weather"
imperial = 'imperial'
metric = 'metric'
error_free = True

class color:
   bold = '\033[1m'
   end = '\033[0m'

#FUNCTION DEFINITIONS
#Welcome displays a welcome message in the Main function
def Welcome():
    welcome_msg1 = (color.bold + u'\U0001F324' + ' WEATHER FOX ' + u'\U0001F327' + color.end)
    welcome_msg2 = (color.bold + 'Powered by OpenWeatherMap.org' + color.end)
    print(welcome_msg1.center(115))
    print(welcome_msg2.center(117))

#Quit displays a final message and ends the program. It is called from the Main function
def Quit():
    quit_msg1 = (color.bold + 'WEATHER FOX' + color.end)
    quit_msg2 = (color.bold + 'The Weather Drives Us Crazy \u27A0 Crazy Like A Fox.' + color.end)
    quit_msg3 = (color.bold + '\u00A9 2019 Benjamin Bartek  \U0001F32A  Powered by OpenWeatherMap.org API' + color.end)
    print('\n')
    print(quit_msg1.center(119))
    print(quit_msg2.center(121))
    print(quit_msg3.center(124))

#Prints Main Menu Selections
def forkMsg():
    print("\n\n\t\t\t\t\tPlease select from the following options:\n")
    print("\t\t\t\t\t1. Search by Zip Code\t(U.S. Only)")
    print("\t\t\t\t\t2. Search by City Name\t(U.S. Only)")
    print("\t\t\t\t\t3. Search by City Name\t(International)")
    print("\t\t\t\t\t4. Exit\n")

#Invalid Input Message
def invalidInput():
    print('\n\n\t\t\t\t\t\u26A0 ERROR: Invalid Input. \u26A0')

#Error Message for a bad location
def invalidLoc():
    print('\n\n\t\t\t\t\t\t\t\U0001F98A')
    print('\t\t\t\t\t\tGoing to Ground...')
    print('\t\t\t\u26A0 ERROR: The location you entered cannot be found. Please try again. \u26A0\n')

#Error message for a bad server connection
def badConnect():
    print('\n\n\t\t\t\t\t\t\t\U0001F98A')
    print('\t\t\t\t\t\tGoing to Ground...')
    print('\t\t\t\u26A0 ERROR: Connection Not Successful. Please try again. \u26A0n')

#Confirmation message for successful data retrieval
def success():
    print('\n\n\t\t\t\t\t\t\t\U0001F98A')
    print('\t\t\t\t\t\tGoing to Ground...')
    print('\t\t\t\t\tSuccess! Weather Data Retrieved. \u2714')

#Retrieves data by U.S. zip code input
def getWeather1():
    zip_code = input('\t\t\t\t\tEnter Your U.S. Zip Code: ')
    zip_query = {"zip":zip_code, "APPID":api_key, "units":imperial}
    headers = {'cache-control':'no-cache'}
    try:
        response_raw = requests.get(url, headers=headers, params=zip_query) #API request
        response_string = str(response_raw.text) #Intermediary to turn into string
        response_dict = json.loads(response_string) #JSON dictionary

        if response_dict['cod'] == '400' or response_dict['cod'] == '401' or response_dict['cod'] == '404' or response_dict['cod'] ==
            invalidLoc()
            error free = False
```

```
                  _
        else:
            error_free = True

    except:
        badConnect()
        error_free = False

    if error_free:
        success()
        printWeather(response_dict)

#Retrieves data by U.S. city input
def getWeather2():
    city_name = input('\t\t\t\t\tEnter U.S. City Name: ')
    city_query = {'q':city_name, "APPID":api_key, "units":imperial}
    headers = {'cache-control':'no-cache'}

    #ERROR CHECKING
    try:
        response_raw = requests.get(url, headers=headers, params=city_query) #API request
        response_string = str(response_raw.text) #Intermediary to turn into string
        response_dict = json.loads(response_string) #JSON dictionary

        if response_dict['cod'] == '400' or response_dict['cod'] == '401' or response_dict['cod'] == '404' or response_dict['cod'] ==
            invalidLoc()
            error_free = False

        else:
            error_free = True

    except:
        badConnect()
        error_free = False

    if error_free:
        success()
        printWeather(response_dict)

#Retrieves data by international city & country code input
def getWeather3():
    city_name2 = input('\t\t\t\t\tCity Name & Country Code: ')
    city_query2 = {'q':city_name2, "APPID":api_key, "units":metric}
    headers = {'cache-control':'no-cache'}

    #ERROR CHECKING
    try:
        response_raw = requests.get(url, headers=headers, params=city_query2)  #API request
        response_string = str(response_raw.text) #Intermediary to turn into string
        response_dict = json.loads(response_string) #JSON dictionary
        if response_dict['cod'] == '400' or response_dict['cod'] == '401' or response_dict['cod'] == '404' or response_dict['cod'] ==
            invalidLoc()
            error_free = False

        else:
            error_free = True

    except:
        badConnect()
        error_free = False

    if error_free:
        success()
        printWeather_intl(response_dict)

#Prints U.S. Weather Data in Imperial Units
def printWeather(response_dict):
    #Variables used when I need to append formatting
    main_header = (color.bold + 'Current Weather Conditions for ' + response_dict['name'] + color.end)
    main_msg = (color.bold + response_dict['weather'][0]['main'] + color.end)
    response_sys = response_dict['sys']
    sunrise_ts = time.gmtime(response_dict['sys']['sunrise'])
    sunset_ts = time.gmtime(response_dict['sys']['sunset'])
    sub_desc = response_dict['weather'][0]['description']

    print('\n\n',main_header.center(117),':\n') #Data Header
    print(main_msg.center(117),'\n') #main description
    print('\t\t\t\t\tDescription:\t', sub_desc.title()) #sub description
    print('\t\t\t\t\tCurrent Temp:\t', round(response_dict['main']['temp']), '°F') #current temp
    print('\t\t\t\t\tHigh/Low:\t', round(response_dict['main']['temp_max']), '°F /', round(response_dict['main']['temp_min']), '°F')
    print('\t\t\t\t\tPressure:\t', response_dict['main']['pressure'], 'mbar') #pressure 1 hPA = 1 mbar
    print('\t\t\t\t\tHumidity:\t', response_dict['main']['humidity'], '%\n') #humidity
    print('\t\t\t\t\tSunrise:\t', time.strftime("%x %X", sunrise_ts), 'GMT') #sunrise
    print('\t\t\t\t\tSunset:\t\t', time.strftime("%x %X", sunset_ts), 'GMT\n') #sunset
    print('\t\t\t\t\tWind Speed:\t', round(response_dict['wind']['speed']), ' mph\n\n') #wind speed
    #print('\t\t\t\t\tWind Dir.:\t', round(response_dict['wind']['deg']), 'degrees\n\n') #wind direction
    #NOTE: wind direction disabled due to traceback error when 'deg' doesn't exist. Discovered during testing with Los Angeles

#Prints International Weather Data in Metric Units - Includes U.S. Data if Called From International Function
def printWeather_intl(response_dict):
    #Variables used when I need to append formatting
    main_header = (color.bold + 'Current Weather Conditions for ' + response_dict['name'] + color.end)
    main_msg = (color.bold + response_dict['weather'][0]['main'] + color.end)
    response_sys = response_dict['sys']
    sunrise_ts = time.gmtime(response_dict['sys']['sunrise'])
    sunset_ts = time.gmtime(response_dict['sys']['sunset'])
    sub_desc = response_dict['weather'][0]['description']

    print('\n\n',main_header.center(117),':\n') #Data Header
    print(main_msg.center(117),'\n') #main description
    print(main_msg.center(11), '\n') #main description
    print('\t\t\t\t\tDescription:\t', sub_desc.title()) #sub description
    print('\t\t\t\t\tCurrent Temp:\t', round(response_dict['main']['temp']), '°C') #current temp
    print('\t\t\t\t\tHigh/Low:\t', round(response_dict['main']['temp_max']), '°C /', round(response_dict['main']['temp_min']), '°C')
    print('\t\t\t\t\tPressure:\t', response_dict['main']['pressure'], 'mbar') #pressure 1 hPA = 1 mbar
    print('\t\t\t\t\tHumidity:\t', response_dict['main']['humidity'], '%\n') #humidity
    print('\t\t\t\t\tSunrise:\t', time.strftime("%x %X", sunrise_ts), 'GMT') #sunrise
    print('\t\t\t\t\tSunset:\t\t', time.strftime("%x %X", sunset_ts), 'GMT\n') #sunset
    print('\t\t\t\t\tWind Speed:\t', round(response_dict['wind']['speed']), ' kph\n\n') #wind speed
    #print('\t\t\t\t\tWind Dir.:\t', round(response_dict['wind']['deg']), 'degrees\n\n')#wind direction
    #NOTE: wind direction disabled due to traceback error when 'deg' doesn't exist. Discovered during testing with Los Angeles

#Main Program
def main():
    Welcome()
    #Main Loop
    while True:
        forkMsg() #Menu

        CityZipFork = input("\t\t\t\t\tEnter Option 1, 2, 3, or 4: ") #Menu Input

        #Checks for valid menu input
        try:
            int(CityZipFork)
        except:
            invalidInput()
            continue

        if CityZipFork == '1':
            getWeather1()

        elif CityZipFork == '2':
            getWeather2()

        elif CityZipFork == '3':
            getWeather3()

        elif CityZipFork == '4':
            Quit()
            break

        else:
            invalidInput()
            continue

        #Allows user to re-run the main loop and obtain weather data for additional locations
        again = input('\t\t\t\t\tFetch Another City? Y or N: ')
        if again.startswith('y') or again.startswith('Y'):
            continue
        elif again.startswith('n') or again.startswith('N'):
            Quit()
```

```
                break
            else:
                print('\n\n\t\t\t\t\u26A0 ERROR: Invalid Input. Redirecting to the Main Menu. \u26A0')

#Run Main Program As Script
if __name__ == "__main__":
    main()
```

```
                            ⛅ WEATHER FOX 🌨
                        Powered by OpenWeatherMap.org


                        Please select from the following options:

                        1. Search by Zip Code   (U.S. Only)
                        2. Search by City Name  (U.S. Only)
                        3. Search by City Name  (International)
                        4. Exit

                        Enter Option 1, 2, 3, or 4: 1
                        Enter Your U.S. Zip Code: 49546


                                        🦊
                                Going to Ground...
                        Success! Weather Data Retrieved. ✔


                  Current Weather Conditions for Grand Rapids                              :

                                  Clouds

                        Description:      Overcast Clouds
                        Current Temp:     24 °F
                        High/Low:         27 °F / 22 °F
                        Pressure:         1022 mbar
                        Humidity:         63 %

                        Sunrise:          03/01/19 12:18:25 GMT
                        Sunset:           03/01/19 23:31:27 GMT

                        Wind Speed:       4  mph


                        Fetch Another City? Y or N: y


                        Please select from the following options:

                        1. Search by Zip Code   (U.S. Only)
                        2. Search by City Name  (U.S. Only)
```

```
3. Search by City Name  (International)
4. Exit

Enter Option 1, 2, 3, or 4: 2
Enter U.S. City Name: Detroit


                🦊
        Going to Ground...
Success! Weather Data Retrieved. ✔


Current Weather Conditions for Detroit                                  :

                Snow

Description:     Heavy Snow
Current Temp:    24 °F
High/Low:        27 °F / 22 °F
Pressure:        1023 mbar
Humidity:        58 %

Sunrise:         03/01/19 12:07:52 GMT
Sunset:          03/01/19 23:21:54 GMT

Wind Speed:      3  mph


Fetch Another City? Y or N: y


Please select from the following options:

1. Search by Zip Code   (U.S. Only)
2. Search by City Name  (U.S. Only)
3. Search by City Name  (International)
4. Exit

Enter Option 1, 2, 3, or 4: 3
City Name & Country Code: Moscow, RU

                🦊
        Going to Ground...
Success! Weather Data Retrieved. ✔


Current Weather Conditions for Moscow                                   :
```

```
                 Snow

Description:     Light Shower Snow
Current Temp:    0 °C
High/Low:        1 °C / -1 °C
Pressure:        981 mbar
Humidity:        92 %

Sunrise:         03/01/19 04:21:40 GMT
Sunset:          03/01/19 15:03:10 GMT

Wind Speed:      2  kph


Fetch Another City? Y or N: y


Please select from the following options:

1. Search by Zip Code   (U.S. Only)
2. Search by City Name  (U.S. Only)
3. Search by City Name  (International)
4. Exit

Enter Option 1, 2, 3, or 4: and;flkj


⚠ ERROR: Invalid Input. ⚠


Please select from the following options:

1. Search by Zip Code   (U.S. Only)
2. Search by City Name  (U.S. Only)
3. Search by City Name  (International)
4. Exit

Enter Option 1, 2, 3, or 4: 1
Enter Your U.S. Zip Code: as;dlfkj


                    🦊
              Going to Ground...
⚠ ERROR: The location you entered cannot be found. Please try again. ⚠

Fetch Another City? Y or N: a;sdlfkj


⚠ ERROR: Invalid Input. Redirecting to the Main Menu. ⚠
```

```
                    Please select from the following options:

                    1. Search by Zip Code   (U.S. Only)
                    2. Search by City Name  (U.S. Only)
                    3. Search by City Name  (International)
                    4. Exit

                    Enter Option 1, 2, 3, or 4: 2
                    Enter U.S. City Name: as;dlfkj


                                        🦊
                                 Going to Ground...
         ⚠ ERROR: The location you entered cannot be found. Please try again. ⚠

                    Fetch Another City? Y or N: y


                    Please select from the following options:

                    1. Search by Zip Code   (U.S. Only)
                    2. Search by City Name  (U.S. Only)
                    3. Search by City Name  (International)
                    4. Exit

                    Enter Option 1, 2, 3, or 4: 3
                    City Name & Country Code: a;sdflkj


                                        🦊
                                 Going to Ground...
         ⚠ ERROR: The location you entered cannot be found. Please try again. ⚠

                    Fetch Another City? Y or N: n


                                     WEATHER FOX
                   The Weather Drives Us Crazy ➠ Crazy Like A Fox.
               © 2019 Benjamin Bartek  🌪  Powered by OpenWeatherMap.org API
```

```
In [ ]:
```