# Term Project DSC - 540

In this project I am trying to scrape ZIPcodes from www.zip-codes.com (http://www.zip-codes.com) for North Carolina and use those ZIPcodes to pull current weather data from https://openweathermap.org/current (https://openweathermap.org/current) and do some data transformations to display data in readable format.

In [1]:
```
# import libraries
import requests
from bs4 import BeautifulSoup
import requests
import json
import pandas as pd
import pprint
import datetime
import time
```

## Scrape ZIP codes from https://www.zip-codes.com/state/nc.asp (https://www.zip-codes.com/state/nc.asp)

In [2]:
```
# use get method to send a GET request to the URL
page = requests.post('https://www.zip-codes.com/state/nc.asp')

#passes the HTML of the page into the BeautifulSoup class
bs = BeautifulSoup(page.content)

#Uses Beautiful Soup to find and return all tables with id containing the string
_tables = bs.find_all("table", id ="tblZIP")
print(len(_tables))
print(_tables)
```

```
1
[<table border="0" cellpadding="0" cellspacing="0" class="statTable" id="tblZ
IP" title="All North Carolina ZIP Codes, City, County, Classification, and Ar
ea Codes." width="99%">
<tr>
<td class="label" title="All ZIP Codes for North Carolina"><strong>ZIP Code</
strong></td>
<td class="info" title="The official city name as designated by the USPS."><s
trong>City</strong></td>
<td class="info" title="The primary county or parish this ZIP Code serves."><
strong>County</strong></td>
<td class="info" title="The classification type of this ZIP Code."><strong>Ty
pe</strong></td>
</tr><tr><td><a href="/zip-code/27006/zip-code-27006.asp" title="ZIP Code 270
06">ZIP Code 27006</a></td><td><a href="/city/nc-advance.asp" title="Advance,
NC">Advance</a></td><td><a href="/county/nc-davie.asp">Davie</a></td><td>Stan
dard</td></tr><tr><td><a href="/zip-code/27007/zip-code-27007.asp" title="ZIP
Code 27007">ZIP Code 27007</a></td><td><a href="/city/nc-ararat.asp" title="A
rarat, NC">Ararat</a></td><td><a href="/county/nc-surry.asp">Surry</a></td><t
```

In [3]:
```
# create zipcodeslist
_zipcodeList = []

# find all a's in the first table
_AList = _tables[0].find_all('a')

# extract 'title' for all a's
for _a in _AList:
    if 'title' in _a.attrs.keys():
        _title = _a['title']

        # extract zipcodes string and append zipcodelist
        if _title.startswith('ZIP'):
            _zipcodeList.append(_title.split(' ')[-1])

# find length of the zipcodelist and print
print(len(_zipcodeList))
print(_zipcodeList)
```

```
1083
['27006', '27007', '27009', '27010', '27011', '27012', '27013', '27014', '2701
6', '27017', '27018', '27019', '27020', '27021', '27022', '27023', '27024', '27
025', '27027', '27028', '27030', '27031', '27040', '27041', '27042', '27043',
'27045', '27046', '27047', '27048', '27049', '27050', '27051', '27052', '2705
3', '27054', '27055', '27094', '27098', '27099', '27101', '27102', '27103', '27
104', '27105', '27106', '27107', '27108', '27109', '27110', '27111', '27113',
'27114', '27115', '27116', '27117', '27120', '27127', '27130', '27150', '2715
2', '27155', '27157', '27198', '27199', '27201', '27202', '27203', '27204', '27
205', '27207', '27208', '27209', '27212', '27213', '27214', '27215', '27216',
'27217', '27228', '27229', '27230', '27231', '27233', '27235', '27237', '2723
9', '27242', '27243', '27244', '27247', '27248', '27249', '27252', '27253', '27
256', '27258', '27259', '27260', '27261', '27262', '27263', '27264', '27265',
'27268', '27278', '27281', '27282', '27283', '27284', '27285', '27288', '2728
9', '27291', '27292', '27293', '27294', '27295', '27298', '27299', '27301', '27
302', '27305', '27306', '27310', '27311', '27312', '27313', '27314', '27315',
'27316', '27317', '27320', '27323', '27325', '27326', '27330', '27331', '2733
2', '27340', '27341', '27342', '27343', '27344', '27349', '27350', '27351', '27
355', '27356', '27357', '27358', '27359', '27360', '27361', '27370', '27371',
'27373', '27374', '27375', '27376', '27377', '27379', '27401', '27402', '2740
3', '27404', '27405', '27406', '27407', '27408', '27409', '27410', '27411', '27
412', '27413', '27415', '27416', '27417', '27419', '27420', '27425', '27427',
'27429', '27435', '27438', '27455', '27495', '27497', '27498', '27499', '2750
1', '27502', '27503', '27504', '27505', '27506', '27507', '27508', '27509', '27
510', '27511', '27512', '27513', '27514', '27515', '27516', '27517', '27518',
'27519', '27520', '27521', '27522', '27523', '27524', '27525', '27526', '2752
7', '27528', '27529', '27530', '27531', '27532', '27533', '27534', '27536', '27
537', '27539', '27540', '27541', '27542', '27543', '27544', '27545', '27546',
'27549', '27551', '27552', '27553', '27555', '27556', '27557', '27559', '2756
0', '27562', '27563', '27565', '27568', '27569', '27570', '27571', '27572', '27
573', '27574', '27576', '27577', '27581', '27582', '27583', '27584', '27586',
'27587', '27588', '27589', '27591', '27592', '27593', '27594', '27596', '2759
7', '27599', '27601', '27602', '27603', '27604', '27605', '27606', '27607', '27
608', '27609', '27610', '27611', '27612', '27613', '27614', '27615', '27616',
'27617', '27619', '27620', '27621', '27622', '27623', '27624', '27625', '2762
6', '27627', '27628', '27629', '27634', '27635', '27636', '27640', '27650', '27
```

```
656', '27658', '27661', '27668', '27675', '27676', '27690', '27695', '27697',
'27698', '27699', '27701', '27702', '27703', '27704', '27705', '27706', '2770
7', '27708', '27709', '27709', '27710', '27711', '27711', '27712', '27713', '27
715', '27717', '27722', '27801', '27802', '27803', '27804', '27805', '27806',
'27807', '27808', '27809', '27810', '27811', '27812', '27813', '27814', '2781
5', '27816', '27817', '27818', '27819', '27820', '27821', '27822', '27823', '27
824', '27825', '27826', '27827', '27828', '27829', '27830', '27831', '27832',
'27833', '27834', '27835', '27836', '27837', '27839', '27840', '27841', '2784
2', '27843', '27844', '27845', '27846', '27847', '27849', '27850', '27851', '27
852', '27853', '27855', '27856', '27857', '27858', '27860', '27861', '27862',
'27863', '27864', '27865', '27866', '27867', '27868', '27869', '27870', '2787
1', '27872', '27873', '27874', '27875', '27876', '27877', '27878', '27879', '27
880', '27881', '27882', '27883', '27884', '27885', '27886', '27887', '27888',
'27889', '27890', '27891', '27892', '27893', '27894', '27895', '27896', '2789
7', '27906', '27907', '27909', '27910', '27915', '27916', '27917', '27919', '27
920', '27921', '27922', '27923', '27924', '27925', '27926', '27927', '27928',
'27929', '27930', '27932', '27935', '27936', '27937', '27938', '27939', '2794
1', '27942', '27943', '27944', '27946', '27947', '27948', '27949', '27950', '27
953', '27954', '27956', '27957', '27958', '27959', '27960', '27962', '27964',
'27965', '27966', '27967', '27968', '27969', '27970', '27972', '27973', '2797
4', '27976', '27978', '27979', '27980', '27981', '27982', '27983', '27985', '27
986', '28001', '28002', '28006', '28007', '28009', '28010', '28012', '28016',
'28017', '28018', '28019', '28020', '28021', '28023', '28024', '28025', '2802
6', '28027', '28031', '28032', '28033', '28034', '28035', '28036', '28037', '28
038', '28039', '28040', '28041', '28042', '28043', '28052', '28053', '28054',
'28055', '28056', '28070', '28071', '28072', '28073', '28074', '28075', '2807
6', '28077', '28078', '28079', '28080', '28081', '28082', '28083', '28086', '28
088', '28089', '28090', '28091', '28092', '28093', '28097', '28098', '28101',
'28102', '28103', '28104', '28105', '28106', '28107', '28108', '28109', '2811
0', '28111', '28112', '28114', '28115', '28117', '28119', '28120', '28123', '28
124', '28125', '28126', '28127', '28128', '28129', '28130', '28133', '28134',
'28135', '28136', '28137', '28138', '28139', '28144', '28145', '28146', '2814
7', '28150', '28151', '28152', '28159', '28160', '28163', '28164', '28166', '28
167', '28168', '28169', '28170', '28173', '28174', '28201', '28202', '28203',
'28204', '28205', '28206', '28207', '28208', '28209', '28210', '28211', '2821
2', '28213', '28214', '28215', '28216', '28217', '28218', '28219', '28220', '28
221', '28222', '28223', '28224', '28226', '28227', '28228', '28229', '28230',
'28231', '28232', '28233', '28234', '28235', '28236', '28237', '28241', '2824
2', '28243', '28244', '28246', '28247', '28253', '28254', '28255', '28256', '28
258', '28260', '28262', '28263', '28265', '28266', '28269', '28270', '28271',
'28272', '28273', '28274', '28275', '28277', '28278', '28280', '28281', '2828
2', '28284', '28285', '28287', '28288', '28289', '28290', '28296', '28297', '28
299', '28301', '28301', '28302', '28303', '28304', '28305', '28306', '28307',
'28308', '28309', '28310', '28311', '28312', '28314', '28315', '28318', '2831
9', '28320', '28323', '28325', '28326', '28327', '28328', '28329', '28330', '28
331', '28332', '28333', '28334', '28335', '28337', '28338', '28339', '28340',
'28341', '28342', '28343', '28344', '28345', '28347', '28348', '28349', '2835
0', '28351', '28352', '28353', '28355', '28356', '28357', '28358', '28359', '28
360', '28362', '28363', '28364', '28365', '28366', '28367', '28368', '28369',
'28370', '28371', '28372', '28373', '28374', '28375', '28376', '28377', '2837
8', '28379', '28380', '28382', '28383', '28384', '28385', '28386', '28387', '28
388', '28390', '28391', '28392', '28393', '28394', '28395', '28396', '28398',
'28399', '28401', '28402', '28403', '28404', '28405', '28406', '28407', '2840
8', '28409', '28410', '28411', '28412', '28420', '28421', '28422', '28423', '28
424', '28425', '28428', '28429', '28430', '28431', '28432', '28433', '28434',
'28435', '28436', '28438', '28439', '28441', '28442', '28443', '28444', '2844
5', '28447', '28448', '28449', '28450', '28451', '28452', '28453', '28454', '28
```

```
455', '28456', '28457', '28458', '28459', '28460', '28461', '28462', '28463',
'28464', '28465', '28466', '28467', '28468', '28469', '28470', '28472', '2847
8', '28479', '28480', '28501', '28502', '28503', '28504', '28508', '28509', '28
510', '28511', '28512', '28513', '28515', '28516', '28518', '28519', '28520',
'28521', '28522', '28523', '28524', '28525', '28526', '28527', '28528', '2852
9', '28530', '28531', '28532', '28533', '28537', '28538', '28539', '28540', '28
541', '28542', '28543', '28544', '28545', '28546', '28547', '28551', '28552',
'28553', '28554', '28555', '28556', '28557', '28560', '28561', '28562', '2856
3', '28564', '28570', '28571', '28572', '28573', '28574', '28575', '28577', '28
578', '28579', '28580', '28581', '28582', '28583', '28584', '28585', '28586',
'28587', '28589', '28590', '28594', '28601', '28602', '28603', '28604', '2860
5', '28606', '28607', '28608', '28609', '28610', '28611', '28612', '28613', '28
615', '28616', '28617', '28618', '28619', '28621', '28622', '28623', '28624',
'28625', '28626', '28627', '28628', '28629', '28630', '28631', '28633', '2863
4', '28635', '28636', '28637', '28638', '28640', '28641', '28642', '28643', '28
644', '28645', '28646', '28647', '28649', '28650', '28651', '28652', '28653',
'28654', '28655', '28656', '28657', '28658', '28659', '28660', '28661', '2866
2', '28663', '28664', '28665', '28666', '28667', '28668', '28669', '28670', '28
671', '28672', '28673', '28675', '28676', '28677', '28678', '28679', '28680',
'28681', '28682', '28683', '28684', '28685', '28687', '28688', '28689', '2869
0', '28691', '28692', '28693', '28694', '28697', '28698', '28699', '28701', '28
702', '28704', '28705', '28707', '28708', '28709', '28710', '28711', '28712',
'28713', '28714', '28715', '28716', '28717', '28718', '28719', '28720', '2872
1', '28722', '28723', '28724', '28725', '28726', '28727', '28728', '28729', '28
730', '28731', '28732', '28733', '28734', '28735', '28736', '28737', '28738',
'28739', '28740', '28741', '28742', '28743', '28744', '28745', '28746', '2874
7', '28748', '28749', '28750', '28751', '28752', '28753', '28754', '28755', '28
756', '28757', '28758', '28759', '28760', '28761', '28762', '28763', '28765',
'28766', '28768', '28770', '28771', '28772', '28773', '28774', '28775', '2877
6', '28777', '28778', '28779', '28781', '28782', '28783', '28784', '28785', '28
786', '28787', '28788', '28789', '28790', '28791', '28792', '28793', '28801',
'28802', '28803', '28804', '28805', '28806', '28810', '28813', '28814', '2881
5', '28816', '28901', '28902', '28903', '28904', '28905', '28906', '28909']
```

In [4]:
```
# finding duplicates in zipcodelist
# convert zipcodelist to set
x=set(_zipcodeList)
# create duplicates list
dup=[]
# find and append dulicates list
for c in x:
    if(_zipcodeList.count(c)>1):
        dup.append(c)
# print duplicate ZIP codes list
print(dup)
```

```
['28301', '27709', '27711']
```

In [5]:
```
# create final list of zipcodes with out duplicates and find length of the list
finalList = list(set(_zipcodeList))
len(finalList)
```

Out[5]:
```
1080
```

## Pull current weather data from https://openweathermap.org/current

**(https://openweathermap.org/current) and do some data transformations to display data in readable format.**

In [8]:
```
# define get_weather function to extract data from API
def get_weather(code):
    API_KEY = '303edfe18e79163b5aa9cea46e5b8e65'
    url_base = 'http://api.openweathermap.org/data/2.5/weather?'
    url = url_base+'zip='+code+',us&appid='+API_KEY+'&units=imperial'
    r = requests.get(url)
    data = r.json()
    return data

# create used zip codes list and weather data list
used_list = []
weather_data = []

# extract data from API using each zip code from list
for zc in finalList[:len(finalList)]:
    # if zip code is not in used list perform next steps and append the used zip
    if zc not in used_list :
        used_list.append(zc)
        # try to get weather data from API and append weather_data list
        try:
            data1 = get_weather(zc)
            # wait for 0.2 sec to move to another step to limit number of calls p
            time.sleep(0.2)
            weather_data.append(data1)
            print(data1)

        # prints message to user if unable to open url
        except requests.exceptions.ConnectionError as errc:
            # handle ConnectionError exception
            print('\033[91m ' +'***Connection Failure. Please try later.***'+'\0
            break

        # handle all other exceptions
        except Exception as e:
            print('\033[91m '+"Failure to Retrieve.Please try again"+'\033[0m')
```

```
26, 'country': 'US', 'sunrise': 1582631943, 'sunset': 1582672447}, 'timezon
e': -18000, 'id': 0, 'name': 'Mooresville', 'cod': 200}
{'coord': {'lon': -78.94, 'lat': 36}, 'weather': [{'id': 804, 'main': 'Cloud
s', 'description': 'overcast clouds', 'icon': '04n'}], 'base': 'stations', 'm
ain': {'temp': 58.75, 'feels_like': 58.23, 'temp_min': 55.99, 'temp_max': 61,
'pressure': 1011, 'humidity': 93}, 'visibility': 16093, 'wind': {'speed': 4.
7, 'deg': 220}, 'clouds': {'all': 90}, 'dt': 1582677804, 'sys': {'type': 1,
'id': 5645, 'country': 'US', 'sunrise': 1582631516, 'sunset': 1582671972}, 't
imezone': -18000, 'id': 0, 'name': 'Durham', 'cod': 200}
{'coord': {'lon': -80.34, 'lat': 36.18}, 'weather': [{'id': 800, 'main': 'Cle
ar', 'description': 'clear sky', 'icon': '01n'}], 'base': 'stations', 'main':
{'temp': 53.2, 'feels_like': 50.72, 'temp_min': 51.01, 'temp_max': 55.99, 'pr
essure': 1010, 'humidity': 87}, 'visibility': 16093, 'wind': {'speed': 4.27,
'deg': 203}, 'clouds': {'all': 1}, 'dt': 1582677805, 'sys': {'type': 1, 'id':
5842, 'country': 'US', 'sunrise': 1582631862, 'sunset': 1582672298}, 'timezon
e': -18000, 'id': 0, 'name': 'Bethania', 'cod': 200}
{'coord': {'lon': -77.21, 'lat': 36.33}, 'weather': [{'id': 804, 'main': 'Clo
uds', 'description': 'overcast clouds', 'icon': '04n'}], 'base': 'stations',
'main': {'temp': 57.38, 'feels_like': 57.83, 'temp_min': 53.6, 'temp_max': 6
2.6, 'pressure': 1011, 'humidity': 100}, 'visibility': 16093, 'wind': {'spee
```

In [9]:
```
len(weather_data)
```

Out[9]:
```
1080
```

In [10]:
```
len(used_list)
```

Out[10]:
```
1080
```

In [11]:
```
# using == to check if lists are equal
if used_list == finalList:
    print ("The lists are identical")
else :
    print ("The lists are not identical")
```

```
The lists are identical
```

In [12]:
```
# create 'weather.json' file with weather data
with open('weather.json', 'w') as outfile:
    json.dump(weather_data, outfile)
```

In [13]:
```
# read json file
with open('weather.json', 'r') as myfile:
    data = myfile.read()

    # parse file
    weather_data1 = json.loads(data)

    # using pprint to print each record
    # output will be a dictonary for each record
    for item in weather_data1:
        pprint.pprint(item)
```

```
{'base': 'stations',
 'clouds': {'all': 90},
 'cod': 200,
 'coord': {'lat': 35.99, 'lon': -77.85},
 'dt': 1582677348,
 'id': 0,
 'main': {'feels_like': 60.13,
          'humidity': 93,
          'pressure': 1011,
          'temp': 59.61,
          'temp_max': 62.01,
          'temp_min': 57},
 'name': 'Rocky Mount',
 'sys': {'country': 'US',
         'id': 4012,
         'sunrise': 1582631254,
         'sunset': 1582671710,
         'type': 1},
 'timezone': -18000,
```

In [14]:
```
# define extract_details_and_print function
def extract_details_and_print(raw_json_data):

    # extracting location variable from data
    location = raw_json_data["name"]

    # store the value of "main" key in variable y
    y = raw_json_data["main"]

    # store the value of "temp" in current_temperature
    current_temperature = y["temp"]

    # store the value of "pressure" in current_pressure
    current_pressure = y["pressure"]

    # store the value of "humidity" in current_humidiy
    current_humidiy = y["humidity"]

    # store the value of "weather" key in variable z
    z = raw_json_data["weather"]

    # store the value corresponding to the "description" key at the 0th index of
    weather_description = z[0]["description"]

    #extracting wind speed from data
    wind_speed = raw_json_data["wind"]["speed"]

    #extracting date, sunrise & sunset times
    date = raw_json_data["dt"]

    sunrise = raw_json_data["sys"]["sunrise"]

    sunset = raw_json_data["sys"]["sunset"]

    # converting timestamp to human readable format and print

    Date_time = datetime.datetime.fromtimestamp(date).strftime('%Y-%m-%d %H:%M:%S

    sunrise_time = datetime.datetime.fromtimestamp(sunrise).strftime('%H:%M')

    sunset_time = datetime.datetime.fromtimestamp(sunset).strftime('%H:%M')

    print(" Date & Time:", '\33[32m ' + Date_time +'\033[0m')

    print(" Sunrise: {} \t \t Sunset : {}".format('\33[34m'+ sunrise_time+'\033[(

    # print the following values
    print(  " Location : " + '\33[34m' + str(location) + '\033[0m'
            "\n Temperature (in Fahrenheit) : " +
            '\33[34m' + str(current_temperature) + '\033[0m' +
            "\n Atmospheric pressure (in hPa) : " +
            '\33[34m' + str(current_pressure) + '\033[0m' +
            "\n Humidity (in percentage) : " +
            '\33[34m'+ str(current_humidiy) + '\033[0m' +
            "\n Weather description : " +
            '\33[34m'+ str(weather_description)+ '\033[0m' +
```

```
            "\n Wind speed (in mi/hr) :" +
            '\33[34m'+ str(wind_speed)+ '\033[0m' +
            '\33[34m' + "\n**************************************" + '\033[0m'
```

In [15]:
```
# print weather data in readable format
print('\33[34m'+"**************************************"+
            "\n \t WEATHER INFORMATION" +
            "\n**************************************" + '\033[0m')

# print weather information for first 5 zipcodes
for wd in weather_data1[0:5]:
    extract_details_and_print(wd)
```

```
**************************************
         WEATHER INFORMATION
**************************************
 Date & Time:  2020-02-25 19:35:48
 Sunrise: 06:47                 Sunset : 18:01
 Location : Rocky Mount
 Temperature (in Fahrenheit) : 59.61
 Atmospheric pressure (in hPa) : 1011
 Humidity (in percentage) : 93
 Weather description : overcast clouds
 Wind speed (in mi/hr) :3.36
**************************************
 Date & Time:  2020-02-25 19:38:24
 Sunrise: 07:07                 Sunset : 18:23
 Location : Tuckasegee
 Temperature (in Fahrenheit) : 51.46
 Atmospheric pressure (in hPa) : 1012
 Humidity (in percentage) : 81
 Weather description : clear sky
```

In [ ]: