GENERAL EXPLANATIONS

In this project, we are crawling websites with python to get informations from web. For this process we are using one of the most common libraries called ‘ BeautifulSoup’ with requests library. This library helps us to parse html and find tags like id or class names.

There is also a limitation in our application. Because of request library, functions doesn’t responds to turkish characters. So searching a strings that have any of ‘ç,ğ,ı,ö,ş,ü,’ characters causes crash. Beside our efforts, this exception is not likely to handle since its working in library.

Before running code we need to install required libraries. After this process we can run our code. Here is code to install required libraries:

```
pip install requests
pip install beautifulsoup4
```

CODE EXPLANATION

Process list

```
def main():
    print("\n\nWelcome to webcrawler. Here is a menu for options:\nPlease dont use turkish characters in anyprocess[ç,ğ,ı,ö,ş,ü]\n"
      "0-Exit\n1-wikipedia\n2-weather\n3-Search cheapest product\nPlease input index of process to begin")
    process = input("")
    if process=='0':
        exit()
    elif process=='1':
        wikipedia()
    elif process == '2':
        weather()
    elif process == '3':
        searchProduct()
    else:
        print('Unexpected input. Please try again.\n')
        main()
```

Code prints a list like a menu. Input function waits for a input else gives error and starts same function again. Here is explanation of menu:

0 - exit , stops code

1 - wikipedia, searches on english wikipedia

2 – weather, uses bing search build-in weather function to show weather

3 – product search, uses Cimri.com to find searched product for cheapest price

Bracket Remover

```
def bracket_remover(text):
    pattern = r'\[.*?\]'
    paragraph = re.sub(pattern, '', text)
    print(paragraph)
```

This code helps wikipedia process to show outputs more clear to read. Function takes a text as an input. Removes brackets ([ ]) and information inside brackets. Returns cleared text to user.

There is a example in case of you search ‘Elon Musk ‘ :

CEO and product architect of Tesla, Inc.;[10][11] founder of The Boring Company;[12]

CEO and product architect of Tesla, Inc.; founder of The Boring Company;

Wikipedia Search

```
1.  def wikipedia():
2.      search = input("Please input your wikipedia search string\n")
3.      url = "https://en.wikipedia.org/wiki/" + str(search)
4.      url = url.replace(' ','%20')
5.      if (requests.get(url)).status_code == 200:
6.          source = urllib.request.urlopen(url).read()
7.          soup = bs.BeautifulSoup(source,'html.parser')
8.          print((soup.find(id="firstHeading")).text)
9.          tags = soup.find_all('p',attrs={'class':None})
10.         if len(tags)>=2:
11.             for a in range(2):
12.                 if tags[a]!=None:
13.                     paragraph = tags[a].text
14.                     bracket_remover(paragraph)
15.                 else:
16.                     continue
17.         elif len(tags)==1:
18.             paragraph = tags[0].text
19.             bracket_remover(paragraph)
20.     else:
21.         print("Cannot find results in wikipedia.")
22.     main()
```

In this function we are searching on wikipedia. We are making a request on url so all spaces will be replaced with ‘%20’ which is character set of space for HTML5. After that we are crawling website with called url like:

https://en.wikipedia.org/wiki/Elon%20Musk

We dont want all website to show up in our code. So we are just searching tags with <p> which refers to paragraphs in HTML5. If there is more than two <p> we just get first two paragraphs. If there is only one we are getting only one paragraph. If page response code is not equal to ‘200’ which refers to unsuccesfull connection, we are printing ‘Cannot find results in wikipedia’.

Weather Search With Bing

```
1.  def weather():
2.      location = input("Please input your weather search location\nInput empty for aut
    omatic location[may be wrong!]\n")
3.      url= "https://www.bing.com/search?q="+str(location)+"%20hava%20durumu"
4.      url = url.replace(' ','%20')
5.      source = urllib.request.urlopen(url).read()
6.      soup = bs.BeautifulSoup(source,'html.parser')
7.      if soup.find('div',attrs={'class':'wtr_locTitle wtr_nowrap wtr_ellipses'}) != No
    ne:
8.          print(soup.find('div',attrs={'class':'wtr_locTitle wtr_nowrap wtr_ellipses'}
    ).text +" forecast for right now
```

```
    "+ soup.find('div',attrs={'class':'wtr_caption'}).text + " " + soup.find('div',attrs
    ={'class':'wtr_currTemp b_focusTextLarge'}).text + " degree")
9.      else:
10.         print("Cant find location.")
11.     main()
```

In this part we are crawling in bing search engine for weather forecast. We are waiting for user to input a location. User can either input a location or leave blank to get forecast on current location. To handle URL errors we are using same tecnique to replace spaces in text to '%20'.

If bing engine finds any value that have class names like in line 7 ('wtr_locTitle wtr_nowrap wtr_ellipses'), we understands there is a succesfull call to bing. Then our crawler search for weather forecast and prints results.

Cheapest Product Search With Cimri.com

```
1.  def searchProduct():
2.      productName = input("Please enter your word to search. We will find cheapest in
    web ( dont use turkish characters[ç,ğ,ı,ö,ş,ü]:\n")
3.      url= "https://www.cimri.com/arama?q="+str(productName)
4.      url = url.replace(' ','%20')
5.      source = urllib.request.urlopen(url).read()
6.      soup = bs.BeautifulSoup(source,'html.parser')
7.      product =soup.find_all('h3',attrs={'class':'product-title'})
8.      if len(product)!=0:
9.          productprice=soup.find_all('a',attrs={'s14oa9nh-0 fFCyge'})
10.         print(product[0].text+"\n")
11.         print(productprice[0].text)
12.     else:
13.         print("Cant find product")
14.     main()
```

Cimri.com is one of a good example for webcrawler itself. Website searchs a product on different sites and compares them. Shows prices on different places for same item. We are using their website to make almost something similar. Of course we are replacing spaces with '%20'.

After user inputs a text like 'kolonya' cimri.com searches for cheapest products. Actually website shows many result for query 'kolonya' but we are interested in for first one in list. We are crawling full product name and price that includes seller informations. If there is no product, code print 'Cant find product'.