IS 452

Jianshan Yang (NetID: jy12 4 credit hour)

Final project narrative

My final project is about building a program to collect the deals information I am interested in. Based on my shopping experience and my interested products, I decide to make this program to collect the deals information from 2 different websites for myself, one is a general deals information post website which is slickdeals, the other one is UNIQLO. For UNIQLO, it owns the same format of deals information for different categories. I can use the same process to deal with the different categories.

First, I imported all the modules that I think I may use in this program.

1. # import modules
2. **import** urllib.request  as urllib2
3. **import** re
4. **from** bs4 **import** BeautifulSoup
5. **import** numpy as np
6. **import** pandas as pd
7. **import** string

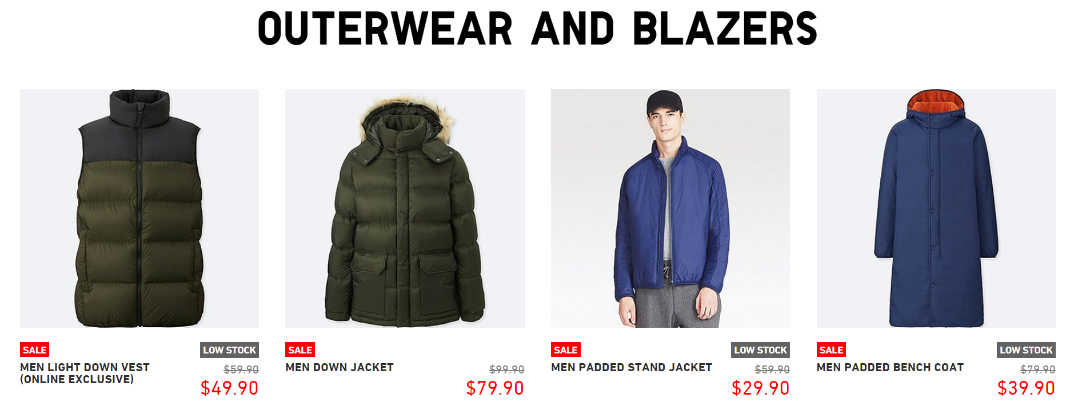
I used six modules in this program. Both urllib2 and BeautifulSoup are used for html file request and read, while BeautifulSoup makes it very easy to extract the content we need, which is prefect for my project.

Since my targets are two different websites, while they used different frames. My plan is design different collection program for them, and then connect them together. And the program will interactive with me when I run, it will show me different result while I input different choice. I decided to start it with UNIQLO Men category. For the information I want is item’s name, sale price, old price, how much we save and the save percentage.

In order to do that, I used urllib2 to open and read all the information, then store it as content and transfer the content to “soup”, which is lxml type we can apply BeautifulSoup modules. The code shows as below:

1. content = urllib2.urlopen('https://www.uniqlo.com/us/en/men/sale').read()
2. soup = BeautifulSoup(content,'lxml')

The original website looks as below:



From the picture, I found most of information I need are showed on the page. Then I checked the page sources by google chrome.

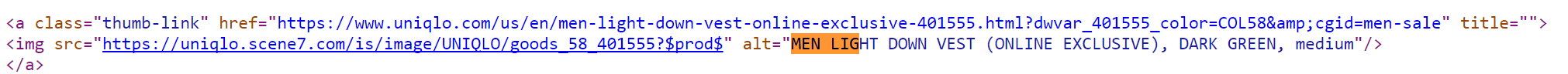


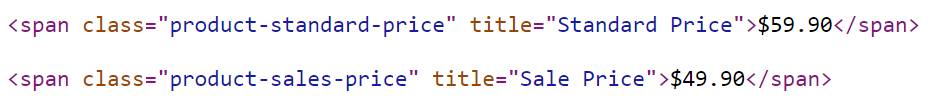
I tried to search the first items showed on the page. And there were many tags I can use for that. Then I decided to use “div” with class “product-tile-info”.

1. items = soup.find\_all("div", class\_='product-tile-info')
2. **print**(len(items))

This step I extracted all the sale items’ information from the website. It gave me a list of all the items. All the elements in the list are bs4.elements, which can be used to future information extract. And I got 107 sale items totally.

Next step I extracted the information I need for all sale items. This process I set them in a loop. As I mentioned before, the information I need includes name, current price, old price, how much we save the save percentages.





From the page sources, I found that name information is in tag ‘img’, and price information is in tag ‘span’ with different title.

1. items\_list = []
2. prices = []
3. old\_prices = []
4. **for** item1 **in** items:
5. # get the list of items' name
6. item = item1.find('img').get('alt')
7. items\_list.append(item)
9. # get the current price
10. price = item1.find('span', class\_='product-sales-price').text
11. prices.append(price)
13. # get the original prices
14. old\_price = item1.find('span', class\_='product-standard-price').text
15. old\_prices.append(old\_price)

So I created different empty list for different information.

For the information, if it’s in tag’s name, I can use ‘get()’, such as item’s name. if it’s the value of a tag, I can use ‘.text’, such as the different price information.

But I still want two more information, one is the how much we save, the other one is save percentage.

I need to calculate in the loop to get this information. I found the prices I got are string, and them begin with a dollar symbol. So, what I need to do is remove the dollar symbol and convert it to float. It’s easy to remove first character for string, I just simply used “string[1:]”. Then I can old price minus current price. For the save percentage I used save divide by the old price.

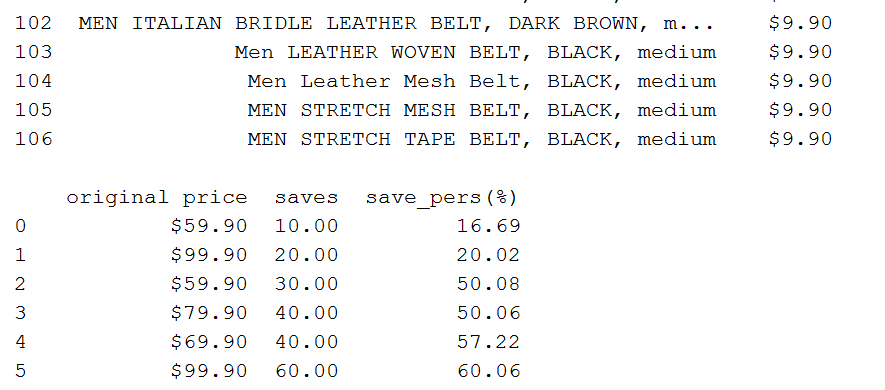
1. # calculate saving money for each item
2. save = float(old\_price[1:]) - float(price[1:])
3. saves.append(round(save,2))
5. # calculate save percentage
6. save\_per = round(save/float(old\_price[1:])\*100,2)
7. save\_pers.append(save\_per)

So I added the above code to the loop.

After this process, I got all the information stored in the list. Then I want to create a data frame to store the information.

1. df = pd.DataFrame()
2. df['items'] = items\_list
3. df['price'] = prices
4. df['original price'] = old\_prices
5. df['saves'] = saves
6. df['save\_pers(%)'] = save\_pers
7. **print**(df)

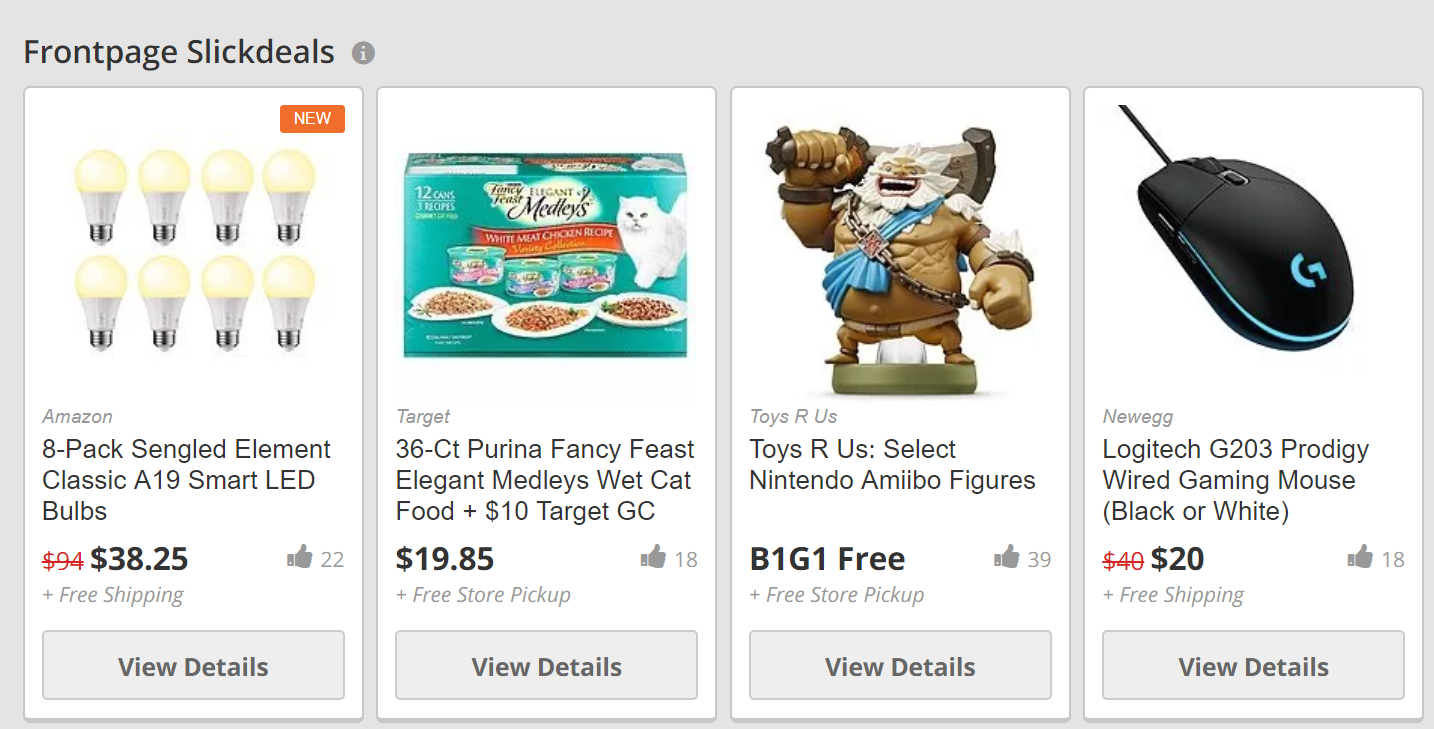
The part of output result showed as below:



As I mentioned before, I decide to get the different categories information for this website. The other category is for women. When I checked the website, it has the similar format with men’s sale webpage. So I added if statement in the front of the code.

1. choice = input('Please enter 1 or 2 to choose ')
2. **if** choice == '1':
3. content = urllib2.urlopen('https://www.uniqlo.com/us/en/men/sale').read()
4. **elif** choice == '2':
5. content = urllib2.urlopen('https://www.uniqlo.com/us/en/women/sale').read()
6. **else**:
7. **print**('Not a valid choice')

The next part, I focus on the website Slickdeals. This is a platform that the users can post the deal information they found. Since it has the different format with UNIQLO website. I can’t just change the url link to get the deal information. But I totally followed the steps I worked on the part one.



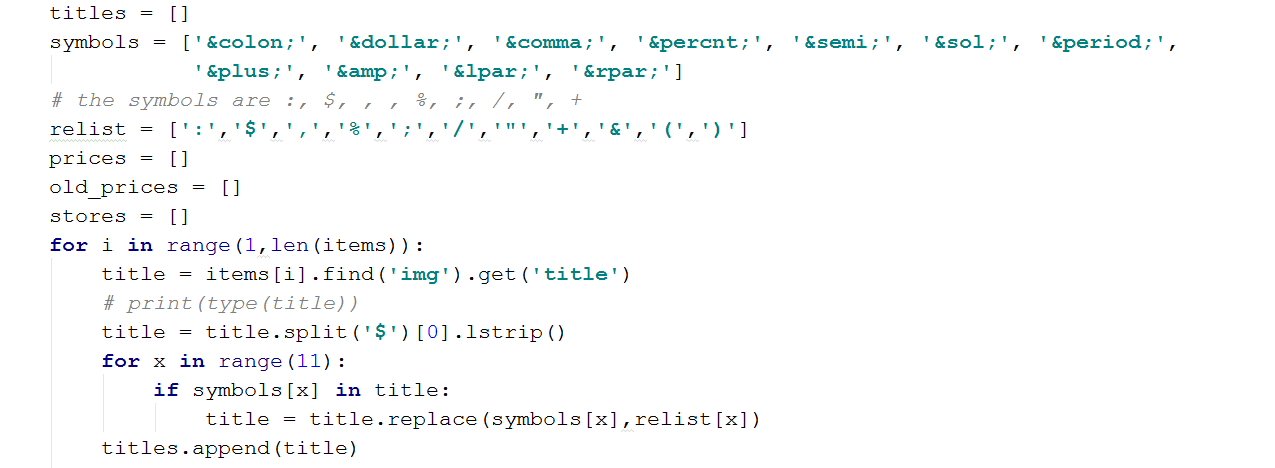


From the original page and the page source, I found I can use div tag with class equals fpGridBox grid frontpage cat to get all the items.

1. content = urllib2.urlopen('https://slickdeals.net/').read()
2. soup = BeautifulSoup(content,'lxml')
4. # after review the page source, the information I need all in the tag 'div' with the class value equals 'fpGridBox grid frontpage cat'
5. # soup.find\_all will return a list with the elements.
6. items = soup.find\_all("div", class\_='fpGridBox grid frontpage cat')

After I got the titles information, I found some problems. The title contains some symbols and it appear as the code, such as “&colon;” is “:”. But I don’t want these codes to appear in my items’ name.

So I used two list to replace all the symbols codes back to symbols.



I created two lists, one is for the symbols, the other one is code. Then I used a loop to check all the results I got, and if it contains a symbol code, the symbol code will be replaced by symbol.

The next problem I met was about the old price, since the information provide by other users, some users may not provide the old price or store information. If there is no such information if will give us None. While the object is None, I can’t use “.text” or “.get”.

For this problem I used if to check the result is None or not, if it’s None, I can just appended None to the list, if it’s not None, I can try to get the value of the tag. The code looks like below:

1. # print(type(old\_price))
2. **if** old\_price **is** None:
3. old\_prices.append(old\_price)
4. **else**:
5. old\_price = items[i].find('span', class\_='oldListPrice').text
6. old\_prices.append(old\_price)
8. # # get the store information
9. store = items[i].find('a', class\_='itemStore')
10. # store.replace(' ','')
11. **if** store **is** None:
12. stores.append(store)
13. **else**:
14. store = items[i].find('a', class\_='itemStore').text
15. store = store[:-1]
16. stores.append(store)

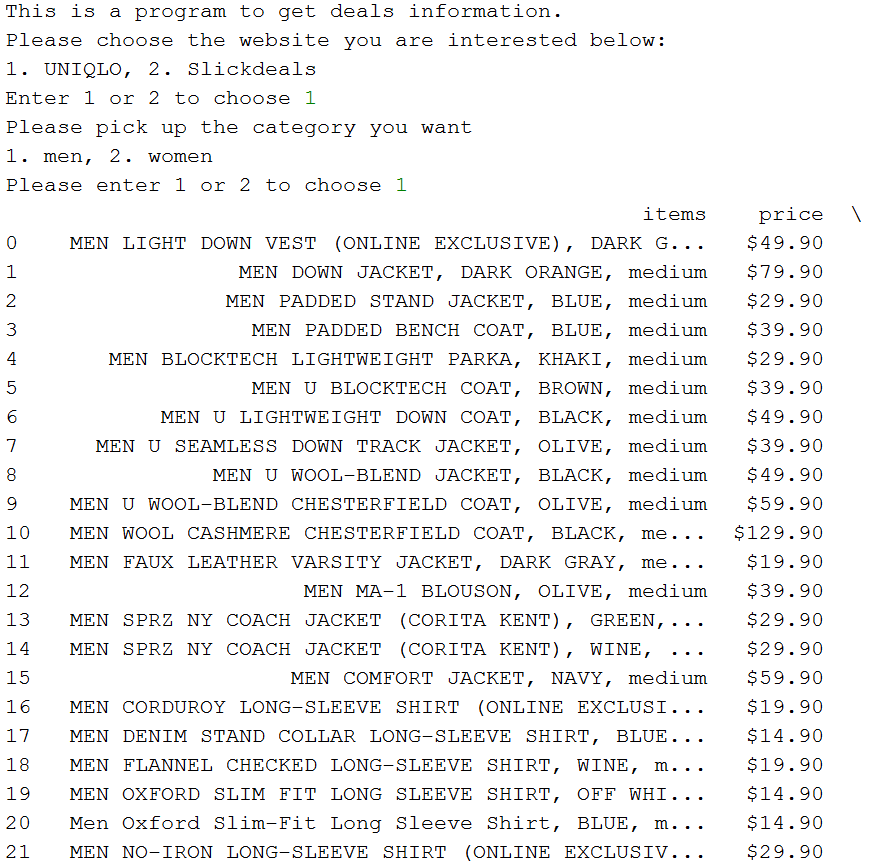
After I got all this information, I also create a data frame to store it. Then I set these two parts into two different functions. One is uniq(), the other one is sd().

Then I set a new choice after the two functions.

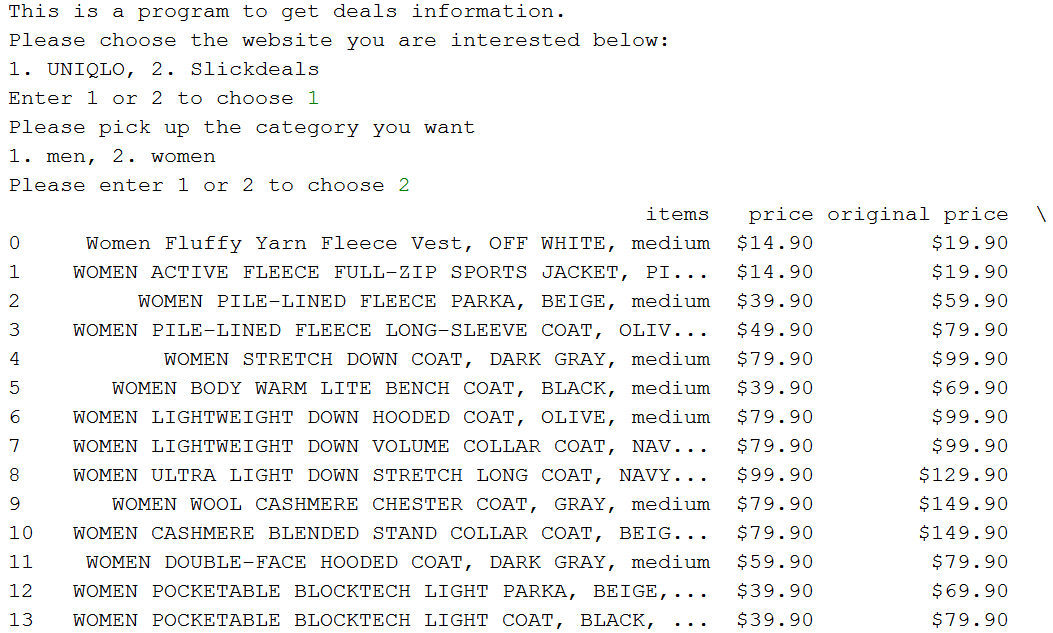
1. **print**('This is a program to get deals information.')
2. **print**('Please choose the website you are interested below:')
3. **print**('1. UNIQLO, 2. Slickdeals')
4. choice = input('Enter 1 or 2 to choose ')
6. **if** choice == '1':
7. uniq()
8. **elif** choice == '2':
9. sd()
10. **else**:
11. **print**('Not a valid choice')

Below are the test:

1. Test for UNIQLO Men’s deal



1. Test for UNIQLO Women’s deal



1. Test for Slickdeals:

