In [70]:
```
import csv
from bs4 import BeautifulSoup
import io
```

In [8]:
```
pip install selenium
```

In [9]:
```
#Firefox and Chrome
```

In [26]:
```
#Startup the webdriver
from selenium import webdriver
driver = webdriver.Chrome(r'C:\Users\prasa\Desktop\Bank\chromedriver_win32\chromedriver.exe')
```

In [27]:
```
url = "https://www.amazon.com"
driver.get(url)
```

In [30]:
```
def get_url(search_term):
    """Generate a url from search term"""
    templete = "https://www.amazon.com/s?k={}&ref=nb_sb_noss_2"
    search_term = search_term.replace(' ','+')
    return templete.format(search_term)
```

In [31]:
```
url = get_url('ultrawide monitor')
print(url)
```

```
https://www.amazon.com/s?k=ultrawide+monitor&ref=nb_sb_noss_2
```

In [33]:
```
driver.get(url)
```

Extract the collection

In [35]:
```
soup = BeautifulSoup(driver.page_source,'html.parser')
```

In [36]:
```
results = soup.find_all('div', {'data-component-type': 's-search-result'})
```

In [37]:
```
len(results)
```

Out[37]:
```
22
```

Prototype the record

In [41]:
```
item = results[0]
```

In [42]:
```
atag = item.h2.a
```

In [59]:
```
description = atag.text.strip()
```

In [47]:
```
url = 'https://www.amazon.com' + atag.get('href')
```

In [48]:
```
price_parent = item.find('span','a-price')
```

In [54]:
```
price = price_parent.find('span','a-offscreen').text
```

In [52]:
```
rating = item.i.text
```

In [57]:
```
review_count = item.find('span',{'class': 'a-size-base','dir': 'auto'}).text
```

Generalize the pattern

In [ ]:
```
def extract_record(item):
```

```
    """Extract and return data from a single record"""

    #description and url
    atag = item.h2.a
    description = atag.text.strip()
    url = 'https://www.amazon.com' + atag.get('href')

    #price
    price_parent = item.find('span','a-price')
    price = price_parent.find('span','a-offscreen').text

    #rank and rating
    rating = item.i.text
    review_count = item.find('span',{'class': 'a-size-base','dir': 'aut
o'}).text

    result = (description,price,rating,review_count,url)

    return result
```

In [ ]:
```
records = []
results = soup.find_all('div',{'data-component-type': 's-search-result'
})

for item in results:
    records.append(extract_record(item))
```

Error handling

In [60]:
```
def extract_record(item):
    """Extract and return data from a single record"""

    #description and url
    atag = item.h2.a
    description = atag.text.strip()
    url = 'https://www.amazon.com' + atag.get('href')
```

```
    try:
        #price
        price_parent = item.find('span','a-price')
        price = price_parent.find('span','a-offscreen').text
    except AttributeError:
        return

    try:
        #rank and rating
        rating = item.i.text
        review_count = item.find('span',{'class': 'a-size-base','dir':
'auto'}).text
    except AttributeError:
        rating = ''
        review_count = ''

    result = (description,price,rating,review_count,url)

    return result
```

In [61]:
```
records = []
results = soup.find_all('div',{'data-component-type': 's-search-result'
})

for item in results:
    record = extract_record(item)
    if record:
        records.append(record)
```

In [62]:
```
records[0]
```

Out[62]:
```
('Philips 292E2E 29" Frameless IPS Monitor, UltraWide Full HD 2560x108
0, 126% sRGB/110% NTSC, 75Hz FreeSync, Height Adjustable, VESA, 4Yr Adv
ance Replacement',
 '$239.99',
 '4.6 out of 5 stars',
 '8,175',
 'https://www.amazon.com/gp/slredirect/picassoRedirect.html/ref=pa_sp_a
tf_aps_sr_pg1_1?ie=UTF8&adId=A039471725XFL323IOIZX&url=%2FPhilips-292E2
```

```
E-Frameless-Adjustable-Replacement%2Fdp%2FB08KFLL9JW%2Fref%3Dsr_1_1_sspa%3Fdchild%3D1%26keywords%3Dultrawide%2Bmonitor%26qid%3D1612164455%26sr%3D8-1-spons%26psc%3D1&qualifier=1612164455&id=119926103825272&widgetName=sp_atf')
```

In [63]:
```
for row in records:
    print(row[1])
```

```
$239.99
$379.97
$226.99
$549.99
$296.99
$349.99
$278.99
$349.99
$499.99
$50.99
$449.99
$547.18
$21.99
$596.99
$349.97
$346.99
$99.99
$11.22
```

Getting the next page

In [ ]:
```
def get_url(search_term):
    """Generate a url from search term"""
    templete = "https://www.amazon.com/s?k={}&ref=nb_sb_noss_2"
    search_term = search_term.replace(' ','+')


    # add term query to url
    url = template.format(search_term)
```

```
    #add page query placeholder

    url += '&page{}'



    return url
```

Putting it all together

In [73]:

```
import csv
from bs4 import BeautifulSoup
from selenium import webdriver #For Chrome

#Getting the pages

def get_url(search_term):
    """Generate a url from search term"""
    template = "https://www.amazon.com/s?k={}&ref=nb_sb_noss_1"
    search_term = search_term.replace(' ','+')


    # add term query to url
    url = template.format(search_term)

    #add page query placeholder

    url += '&page{}'



    return url


#Extract records
```

```
def extract_record(item):
    """Extract and return data from a single record"""

    #description and url
    atag = item.h2.a
    description = atag.text.strip()
    url = 'https://www.amazon.com' + atag.get('href')

    try:
        #price
        price_parent = item.find('span','a-price')
        price = price_parent.find('span','a-offscreen').text
    except AttributeError:
        return

    try:
        #rank and rating
        rating = item.i.text
        review_count = item.find('span',{'class': 'a-size-base','dir':
'auto'}).text
    except AttributeError:
        rating = ''
        review_count = ''

    result = (description,price,rating,review_count,url)

    return result

def main(search_term):
    """Run main program routine"""
    #startup the webdriver
    driver = webdriver.Chrome(r'C:\Users\prasa\Desktop\Bank\chromedrive
r_win32\chromedriver.exe')

    record = []
    url = get_url(search_term)

    for page in range(1,21):
        driver.get(url.format(page))
```

```
        soup = BeautifulSoup(driver.page_source,'html.parser')
        results = soup.find_all('div', {'data-component-type': 's-searc
h-result'})

        for item in results:
            record = extract_record(item)
            if record:
                records.append(record)
    driver.close()

    #save data to csv file
    with open('results.csv','w',newline='',encoding='utf8') as f:
        write = csv.writer(f)
        write.writerow(['Description','Price','Rating','ReviewCount','U
rl'])
        write.writerows(records)
```

In [74]:
```
main('ultrawide monitor')
```

In [ ]:
```

```