**WebScraping**(Worksheet-2)

1. (C)

2. (D)

3. (C)

4. (D)

5. (B)

6. (C)

7. (C)

8. (A)

9. (C)

10. (B)

11. (B,D)

12. (B,C)

13. (A,B,C)

14.

driver=webdriver.Chrome('D:\\chromedriver\_win32\\chromedriver.exe')

my\_page=driver.get('https://www.flipkart.com/sunglasses/pr?p%5B%5D=facets.ideal\_for%255B%255D%3DMen&sid=26x&otracker=nmenu\_sub\_Men\_0\_Sunglasses&otracker=nmenu\_sub\_Men\_0\_Sunglasses&p%5B%5D=facets.ideal\_for%255B%255D%3DMen%2B%2526%2BWomen&p%5B%5D=facets.brand%255B%255D%3DRay-Ban')

make\_directory('Ray-Ban sunglasses')

#scraping images of further pages

page=1

total\_pages=1

try:

product=scrap\_image\_url(driver=driver)

print("Scraping page {0} of {1} pages".format(page,total\_pages))

page\_value=driver.find\_element\_by\_xpath("//div[@class='\_2zg3yZ']//a").text

#download the images

save\_images(data=product,dirname='Ray-Ban sunglasses',page=page)

print("Scraping of page {0} done".format(page))

except StaleElementReferenceException as Exception:

print('We are facing an exception')

exp\_page=driver.find\_element\_by\_xpath("//div[@class='\_2zg3yZ']//a").text

print('The page value at the time of exception is {}'.format(exp\_page))

value=driver.find\_element\_by\_xpath("//div[@class='\_2zg3yZ']//a")

link=value.get\_attribute('href')

driver.get(link)

product=scrap\_image\_url(driver=driver)

print("Scraping page {0} of {1} pages".format(page,total\_pages))

page\_value=driver.find\_element\_by\_xpath("//div[@class='\_2zg3yZ']//a").text

print('The Current Page Scrapped is {}'.format(page\_value))

#download the images

save\_images(data=product,dirname='Ray-Ban sunglasses',page=page)

print("Scraping of page {0} done".format(page))

def scrap\_image\_url(driver):

images=driver.find\_elements\_by\_xpath("//div[@class='\_3ZJShS \_31bMyl']//img")

product\_data= {}

product\_data['image\_urls']=[]

for image in images:

source=image.get\_attribute('src')

product\_data['image\_urls'].append(source)

return product\_data

def save\_images(data,dirname,page):

for index,link in enumerate(data['image\_urls']):

print('Downloading {0} of {1} images'.format(index+1,len(data['image\_urls'])))

response=requests.get(link)

with open('{0}/img\_{1}{2}.jpeg'.format(dirname,page,index),'wb') as file:

file.write(response.content)

15. no\_pages = 2

def get\_data(pageNo):

headers = {"User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0", "Accept-Encoding":"gzip, deflate", "Accept":"text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8", "DNT":"1","Connection":"close", "Upgrade-Insecure-Requests":"1"}

r = requests.get('https://www.amazon.in/s?k=mouse+for+laptop&rh=p\_36%3A-50000&crid=1CHQ1FP3NCFTB&qid=1601131149&rnid=1318502031&sprefix=mouse%2Caps%2C358&ref=sr\_nr\_p\_36\_5'+str(pageNo)+'?ie=UTF8&pg='+str(pageNo), headers=headers)#, proxies=proxies)

content = r.content

soup = BeautifulSoup(content)

#print(soup)

alls = []

for d in soup.findAll('div', attrs={'class':'sg-col-inner'}):

#print(d)

model = d.find('a', attrs={'class':'a-link-normal a-text-normal'})

price = d.find('span', attrs={'class':'a-price-whole'})

all1=[]

if model is not None:

#print(author.text)

all1.append(model.text)

if price is not None:

#print(price.text)

all1.append(price.text)

else:

all1.append('0')

alls.append(all1)

return alls

results = []

for i in range(1, no\_pages+1):

results.append(get\_data(i))

flatten = lambda l: [item for sublist in l for item in sublist]

df = pd.DataFrame(flatten(results),columns=['model','Price'])

df.to\_csv('amazon\_mouse.csv', index=False, encoding='utf-8')

df = pd.read\_csv("amazon\_mouse.csv")