CSE-3024 Web Mining

Lab Assignment 3

Alokam Nikhitha

19BCE2555

Web Crawling

Question

Experiment 4 (28.01.2022)

1. Use BeautifulSoup or Scrapy to crawl any one of the E-commerce website of your choice and perform the same. The following information needs to be extracted from the page: (Choose any one product : e.g laptop, smartphone … etc)

a) Product Name

b) Product price

c) Product discount

d) Product image

Problem statement:

To Crawl any of the E-commerce website and extract the data from the page like Name, Price, Discount and image of the Product.

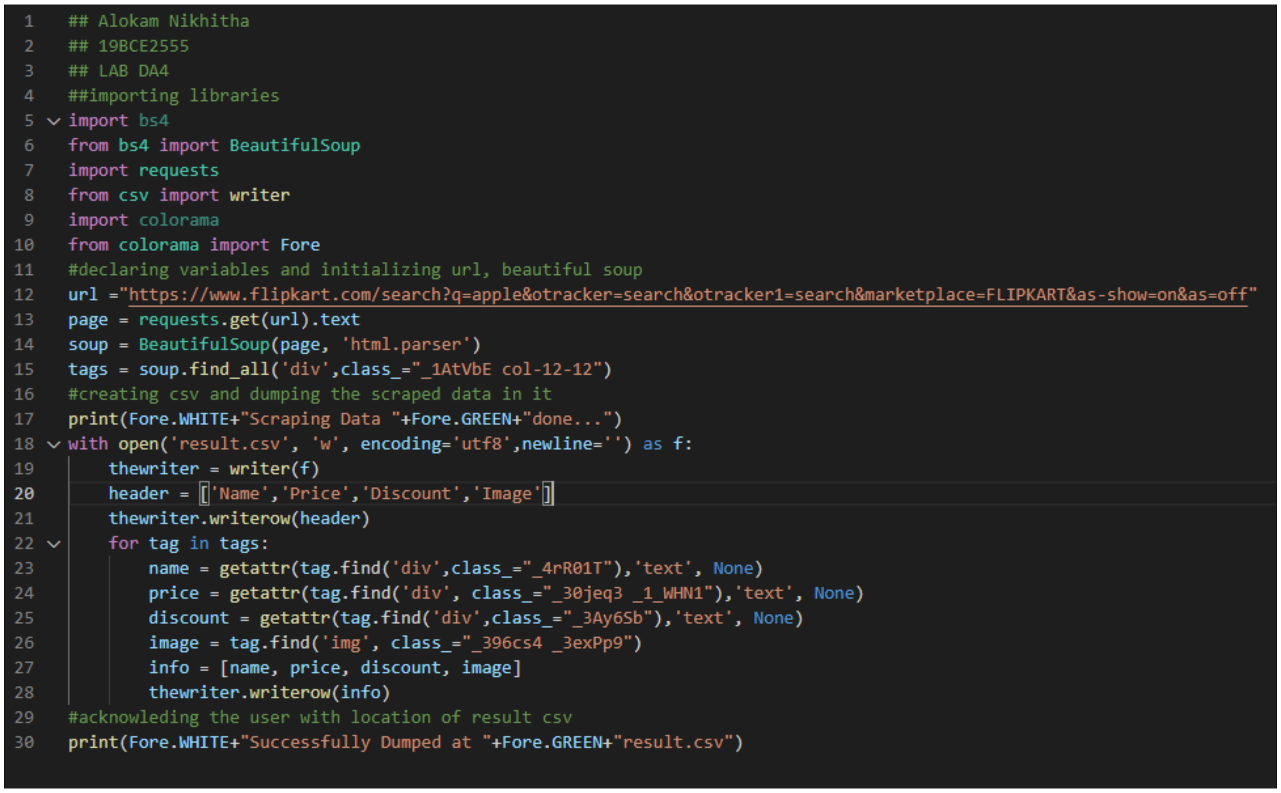
Procedure:

- We will Firstly import our libraries which are necessary in order to scrap the data from the website.
- Later we will declare the variables and also we will initialize the URL and also the beautifulsoup.
- Later we will create the file result.csv and dump the scrapped data into it. Here we will make a made the header with Name, Price, Discount and Image.
- Later we will collect the data of the product and add it to the CSV file.
- On running the python file the results.csv file will be created with scrapped data in it.
- In the result.csv file, the data of the product i.e, Name, Price, Discount and Image will be displayed .The image is returned in the form of a link.

URL of the website from which we are scrapping the data :

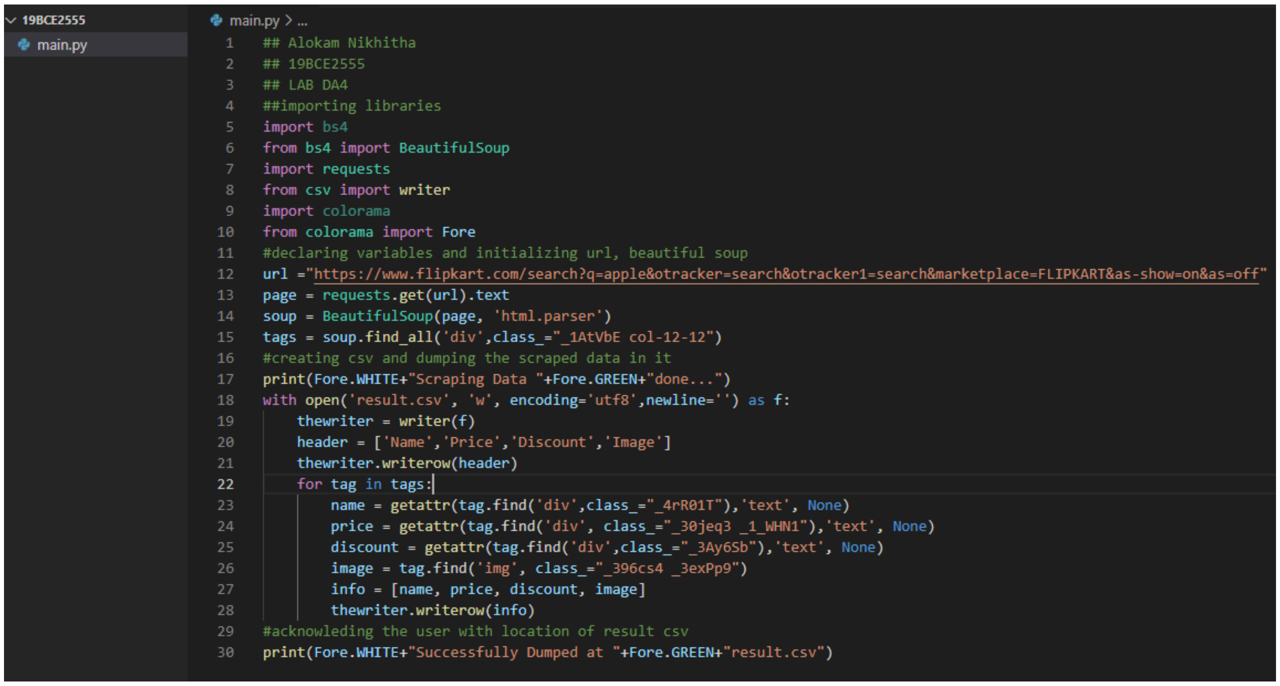
"https://www.flipkart.com/search?q=apple&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off"

Code:



```python
## Alokam Nikhitha
## 19BCE2555
## LAB DA4
##importing libraries
import bs4
from bs4 import BeautifulSoup
import requests
from csv import writer
import colorama
from colorama import Fore
#declaring variables and initializing url, beautiful soup
url ="https://www.flipkart.com/search?q=apple&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off"
page = requests.get(url).text
soup = BeautifulSoup(page, 'html.parser')
tags = soup.find_all('div',class_="_1AtVbE col-12-12")
#creating csv and dumping the scraped data in it
print(Fore.WHITE+"Scraping Data "+Fore.GREEN+"done...")
with open('result.csv', 'w', encoding='utf8',newline='') as f:
    thewriter = writer(f)
    header = ['Name','Price','Discount','Image']
    thewriter.writerow(header)
    for tag in tags:
        name = getattr(tag.find('div',class_="_4rR01T"),'text', None)
        price = getattr(tag.find('div', class_="_30jeq3 _1_WHN1"),'text', None)
        discount = getattr(tag.find('div',class_="_3Ay6Sb"),'text', None)
        image = tag.find('img', class_="_396cs4 _3exPp9")
        info = [name, price, discount, image]
        thewriter.writerow(info)
#acknowleding the user with location of result csv
print(Fore.WHITE+"Successfully Dumped at "+Fore.GREEN+"result.csv")
```

Code Snippets and Outputs:



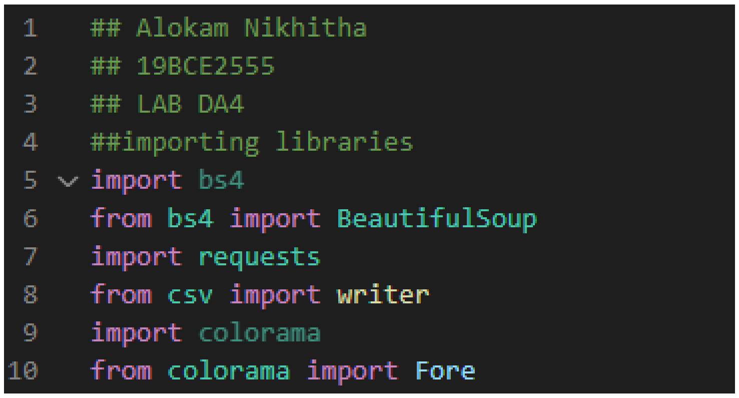
19BCE2555

main.py

main.py > ...

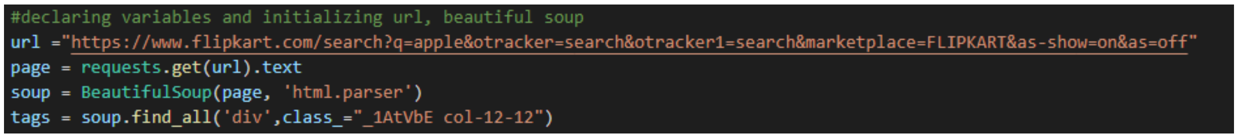
```python
## Alokam Nikhitha
## 19BCE2555
## LAB DA4
##importing libraries
import bs4
from bs4 import BeautifulSoup
import requests
from csv import writer
import colorama
from colorama import Fore
#declaring variables and initializing url, beautiful soup
url ="https://www.flipkart.com/search?q=apple&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off"
page = requests.get(url).text
soup = BeautifulSoup(page, 'html.parser')
tags = soup.find_all('div',class_="_1AtVbE col-12-12")
#creating csv and dumping the scraped data in it
print(Fore.WHITE+"Scraping Data "+Fore.GREEN+"done...")
with open('result.csv', 'w', encoding='utf8',newline='') as f:
    thewriter = writer(f)
    header = ['Name','Price','Discount','Image']
    thewriter.writerow(header)
    for tag in tags:
        name = getattr(tag.find('div',class_="_4rR01T"),'text', None)
        price = getattr(tag.find('div', class_="_30jeq3 _1_WHN1"),'text', None)
        discount = getattr(tag.find('div',class_="_3Ay6Sb"),'text', None)
        image = tag.find('img', class_="_396cs4 _3exPp9")
        info = [name, price, discount, image]
        thewriter.writerow(info)
#acknowleding the user with location of result csv
print(Fore.WHITE+"Successfully Dumped at "+Fore.GREEN+"result.csv")
```

The python file is stored in the folder named ‘19BCE2555’.



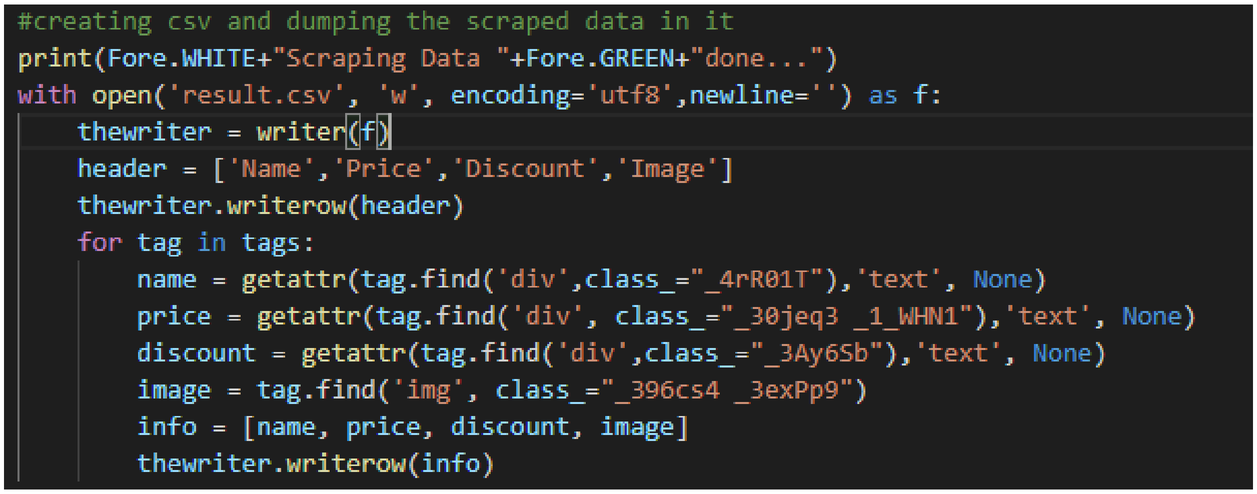
```python
## Alokam Nikhitha
## 19BCE2555
## LAB DA4
##importing libraries
import bs4
from bs4 import BeautifulSoup
import requests
from csv import writer
import colorama
from colorama import Fore
```

Here we are importing the necessary libraries inorder to Scrap the data.



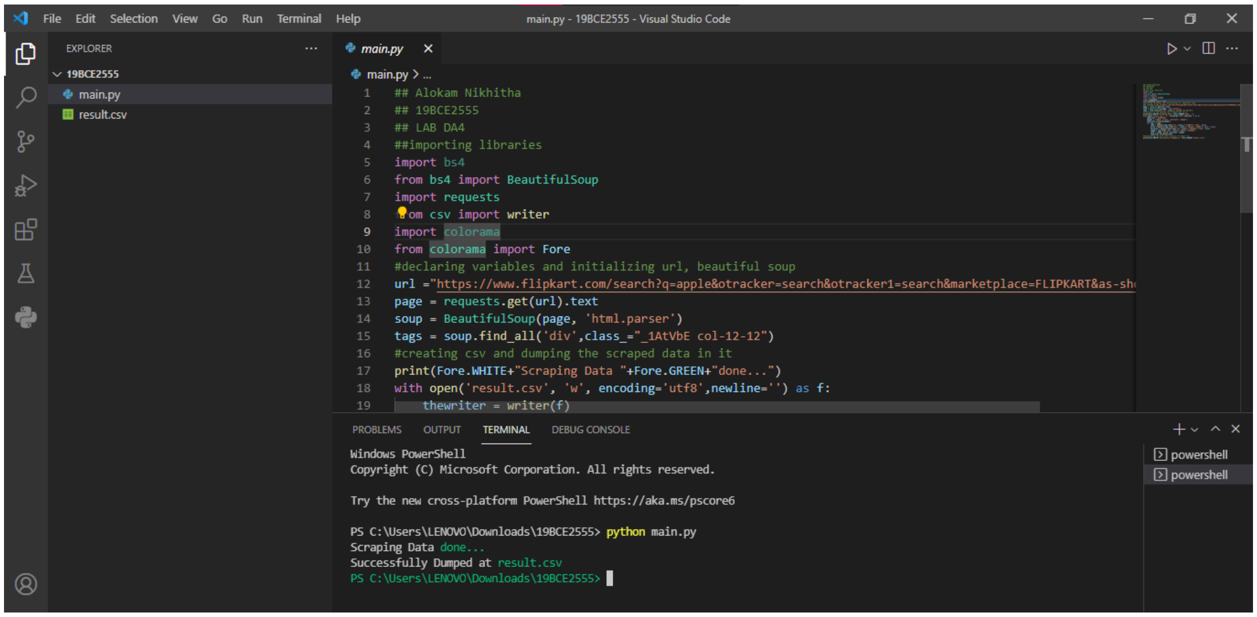
```python
#declaring variables and initializing url, beautiful soup
url ="https://www.flipkart.com/search?q=apple&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off"
page = requests.get(url).text
soup = BeautifulSoup(page, 'html.parser')
tags = soup.find_all('div',class_="_1AtVbE col-12-12")
```

Here we are declaring the variables and also we are Initializing the URL (here url is from ‘Flipkart’ site). And also initializing the beautifulsoup.



```python
#creating csv and dumping the scraped data in it
print(Fore.WHITE+"Scraping Data "+Fore.GREEN+"done...")
with open('result.csv', 'w', encoding='utf8',newline='') as f:
    thewriter = writer(f)
    header = ['Name','Price','Discount','Image']
    thewriter.writerow(header)
    for tag in tags:
        name = getattr(tag.find('div',class_="_4rR01T"),'text', None)
        price = getattr(tag.find('div', class_="_30jeq3 _1_WHN1"),'text', None)
        discount = getattr(tag.find('div',class_="_3Ay6Sb"),'text', None)
        image = tag.find('img', class_="_396cs4 _3exPp9")
        info = [name, price, discount, image]
        thewriter.writerow(info)
```

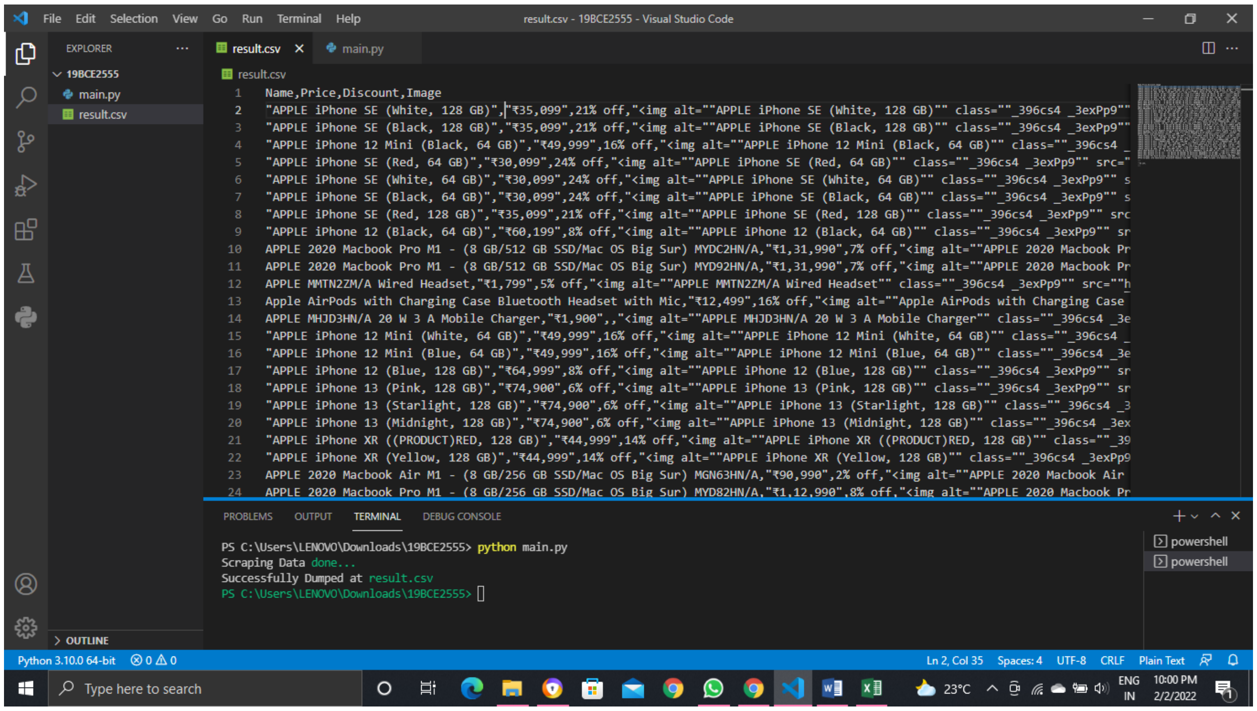
Here we are creating CSV file and dumping the scrapped data in it . Here we are opening the file result.csv and dumping the scrapped data into it. Here we made the header with Name, Price, Discount and Image. Later we are collecting the data of the product and adding it to the CSV file.


File Edit Selection View Go Run Terminal Help
main.py - 19BCE2555 - Visual Studio Code
EXPLORER
19BCE2555
main.py
result.csv
main.py
main.py > ...
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
powershell
powershell

```python
## Alokam Nikhitha
## 19BCE2555
## LAB DA4
##importing libraries
import bs4
from bs4 import BeautifulSoup
import requests
om csv import writer
import colorama
from colorama import Fore
#declaring variables and initializing url, beautiful soup
url ="https://www.flipkart.com/search?q=apple&otracker=search&otracker1=search&marketplace=FLIPKART&as-sh
page = requests.get(url).text
soup = BeautifulSoup(page, 'html.parser')
tags = soup.find_all('div',class_="_1AtVbE col-12-12")
#creating csv and dumping the scraped data in it
print(Fore.WHITE+"Scraping Data "+Fore.GREEN+"done...")
with open('result.csv', 'w', encoding='utf8',newline='') as f:
    thewriter = writer(f)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\LENOVO\Downloads\19BCE2555> python main.py
Scraping Data done...
Successfully Dumped at result.csv
PS C:\Users\LENOVO\Downloads\19BCE2555>
```

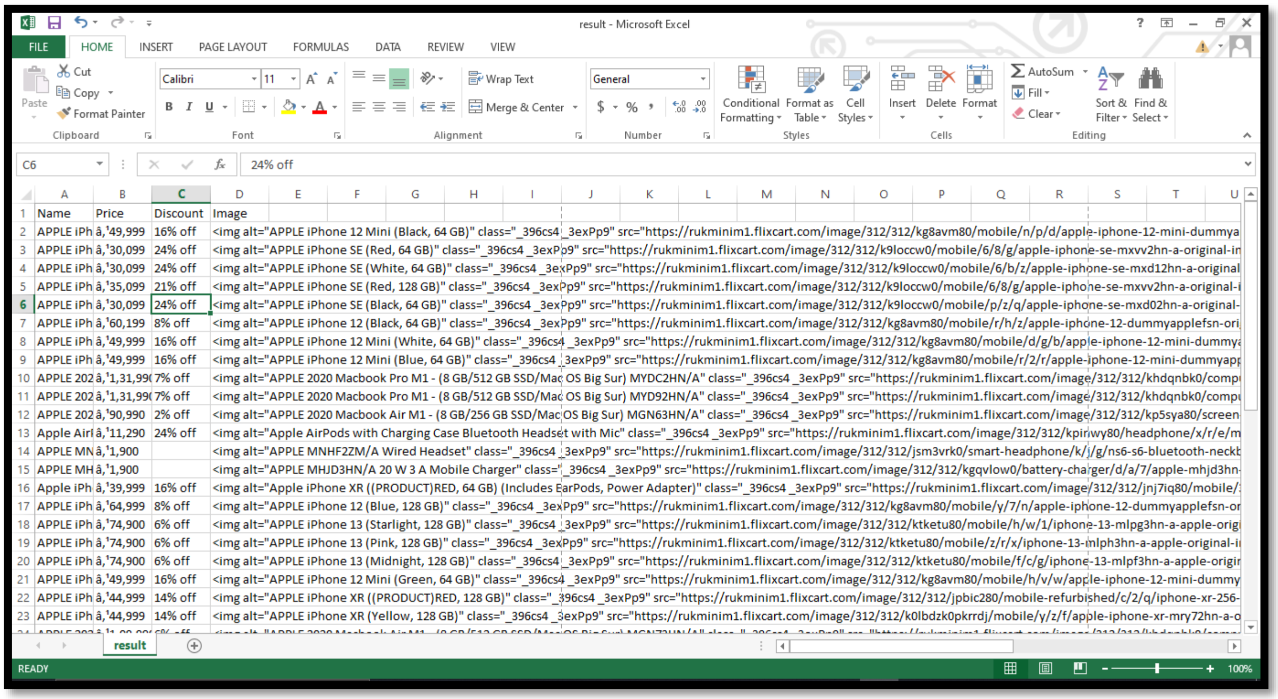
Here we are running the code using the command 'python main.py' in the terminal. Here the "result.csv" file got created in the folder after running the code.


result.csv - 19BCE2555 - Visual Studio Code

```
Name,Price,Discount,Image
"APPLE iPhone SE (White, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (White, 128 GB)"" class=""_396cs4 _3exPp9""
"APPLE iPhone SE (Black, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (Black, 128 GB)"" class=""_396cs4 _3exPp9""
"APPLE iPhone 12 Mini (Black, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (Black, 64 GB)"" class=""_396cs4 _
"APPLE iPhone SE (Red, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (Red, 64 GB)"" class=""_396cs4 _3exPp9"" src="
"APPLE iPhone SE (White, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (White, 64 GB)"" class=""_396cs4 _3exPp9"" s
"APPLE iPhone SE (Black, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (Black, 64 GB)"" class=""_396cs4 _3exPp9"" s
"APPLE iPhone SE (Red, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (Red, 128 GB)"" class=""_396cs4 _3exPp9"" src
"APPLE iPhone 12 (Black, 64 GB)","₹60,199",8% off,"<img alt=""APPLE iPhone 12 (Black, 64 GB)"" class=""_396cs4 _3exPp9"" sr
APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYDC2HN/A,"₹1,31,990",7% off,"<img alt=""APPLE 2020 Macbook Pr
APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYD92HN/A,"₹1,31,990",7% off,"<img alt=""APPLE 2020 Macbook Pr
APPLE MMTN2ZM/A Wired Headset,"₹1,799",5% off,"<img alt=""APPLE MMTN2ZM/A Wired Headset"" class=""_396cs4 _3exPp9"" src=""h
Apple AirPods with Charging Case Bluetooth Headset with Mic,"₹12,499",16% off,"<img alt=""Apple AirPods with Charging Case
APPLE MHJD3HN/A 20 W 3 A Mobile Charger,"₹1,900",,"<img alt=""APPLE MHJD3HN/A 20 W 3 A Mobile Charger"" class=""_396cs4 _3e
"APPLE iPhone 12 Mini (White, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (White, 64 GB)"" class=""_396cs4 _
"APPLE iPhone 12 Mini (Blue, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (Blue, 64 GB)"" class=""_396cs4 _3e
"APPLE iPhone 12 (Blue, 128 GB)","₹64,999",8% off,"<img alt=""APPLE iPhone 12 (Blue, 128 GB)"" class=""_396cs4 _3exPp9"" sr
"APPLE iPhone 13 (Pink, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Pink, 128 GB)"" class=""_396cs4 _3exPp9"" sr
"APPLE iPhone 13 (Starlight, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Starlight, 128 GB)"" class=""_396cs4 _3
"APPLE iPhone 13 (Midnight, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Midnight, 128 GB)"" class=""_396cs4 _3ex
"APPLE iPhone XR ((PRODUCT)RED, 128 GB)","₹44,999",14% off,"<img alt=""APPLE iPhone XR ((PRODUCT)RED, 128 GB)"" class=""_39
"APPLE iPhone XR (Yellow, 128 GB)","₹44,999",14% off,"<img alt=""APPLE iPhone XR (Yellow, 128 GB)"" class=""_396cs4 _3exPp9
APPLE 2020 Macbook Air M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MGN63HN/A,"₹90,990",2% off,"<img alt=""APPLE 2020 Macbook Air
APPLE 2020 Macbook Pro M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MYD82HN/A,"₹1,12,990",8% off,"<img alt=""APPLE 2020 Macbook Pr
```

```
PS C:\Users\LENOVO\Downloads\19BCE2555> python main.py
Scraping Data done...
Successfully Dumped at result.csv
PS C:\Users\LENOVO\Downloads\19BCE2555>
```

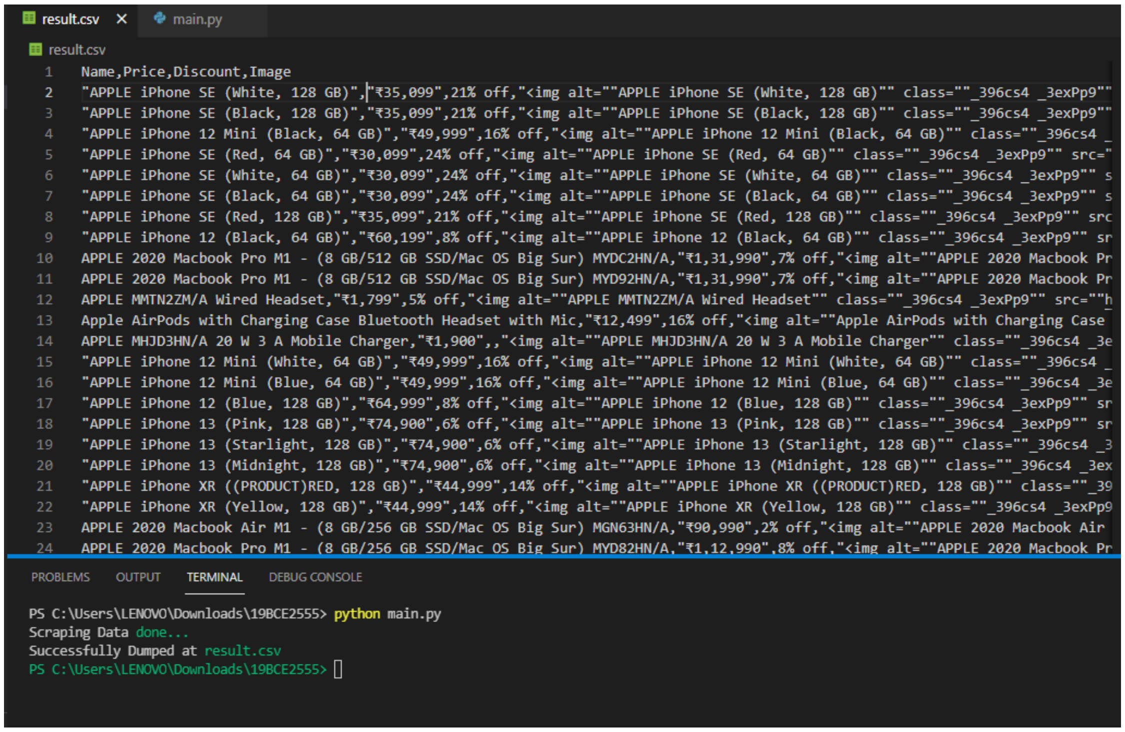
In the result.csv we can see the data that is scrapped from the Flipkart website.


result - Microsoft Excel

	A	B	C	D
1	Name	Price	Discount	Image
2	APPLE iPh	â‚¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/n/p/d/apple-iphone-12-mini-dummya
3	APPLE iPh	â‚¹30,099	24% off	<img alt="APPLE iPhone SE (Red, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/mobile/6/8/g/apple-iphone-se-mxvv2hn-a-original-in
4	APPLE iPh	â‚¹30,099	24% off	<img alt="APPLE iPhone SE (White, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/mobile/6/b/z/apple-iphone-se-mxd12hn-a-original
5	APPLE iPh	â‚¹35,099	21% off	<img alt="APPLE iPhone SE (Red, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/mobile/6/8/g/apple-iphone-se-mxvv2hn-a-original-i
6	APPLE iPh	â‚¹30,099	24% off	<img alt="APPLE iPhone SE (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/mobile/p/z/q/apple-iphone-se-mxd02hn-a-original-
7	APPLE iPh	â‚¹60,199	8% off	<img alt="APPLE iPhone 12 (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/r/h/z/apple-iphone-12-dummyapplefsn-ori
8	APPLE iPh	â‚¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (White, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/d/g/b/apple-iphone-12-mini-dummya
9	APPLE iPh	â‚¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Blue, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/r/2/r/apple-iphone-12-mini-dummyapp
10	APPLE 202	â‚¹1,31,99(	7% off	<img alt="APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYDC2HN/A" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/khdqnbk0/compu
11	APPLE 202	â‚¹1,31,99(	7% off	<img alt="APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYD92HN/A" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/khdqnbk0/compu
12	APPLE 202	â‚¹90,990	2% off	<img alt="APPLE 2020 Macbook Air M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MGN63HN/A" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kp5sya80/screen
13	Apple Airl	â‚¹11,290	24% off	<img alt="Apple AirPods with Charging Case Bluetooth Headset with Mic" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kpinwy80/headphone/x/r/e/m
14	APPLE MN	â‚¹1,900		<img alt="APPLE MNHF2ZM/A Wired Headset" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/jsm3vrk0/smart-headphone/k/j/g/ns6-s6-bluetooth-neckb
15	APPLE MH	â‚¹1,900		<img alt="APPLE MHJD3HN/A 20 W 3 A Mobile Charger" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kgqvlow0/battery-charger/d/a/7/apple-mhjd3hn-
16	Apple iPh	â‚¹39,999	16% off	<img alt="Apple iPhone XR ((PRODUCT)RED, 64 GB) (Includes EarPods, Power Adapter)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/jnj7iq80/mobile/
17	APPLE iPh	â‚¹64,999	8% off	<img alt="APPLE iPhone 12 (Blue, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/y/7/n/apple-iphone-12-dummyapplefsn-or
18	APPLE iPh	â‚¹74,900	6% off	<img alt="APPLE iPhone 13 (Starlight, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu80/mobile/h/w/1/iphone-13-mlpg3hn-a-apple-origi
19	APPLE iPh	â‚¹74,900	6% off	<img alt="APPLE iPhone 13 (Pink, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu80/mobile/z/r/x/iphone-13-mlph3hn-a-apple-original-i
20	APPLE iPh	â‚¹74,900	6% off	<img alt="APPLE iPhone 13 (Midnight, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu80/mobile/f/c/g/iphone-13-mlpf3hn-a-apple-origin
21	APPLE iPh	â‚¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Green, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/mobile/h/v/w/apple-iphone-12-mini-dummy
22	APPLE iPh	â‚¹44,999	14% off	<img alt="APPLE iPhone XR ((PRODUCT)RED, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/jpbic280/mobile-refurbished/c/2/q/iphone-xr-256-
23	APPLE iPh	â‚¹44,999	14% off	<img alt="APPLE iPhone XR (Yellow, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k0lbdzk0pkrrdj/mobile/y/z/f/apple-iphone-xr-mry72hn-a-o

Here is the Excel sheet in which we collected the scrapped data. The name of the product, price and Discount are collected. The image is returned in the form of a link.

Results and Output

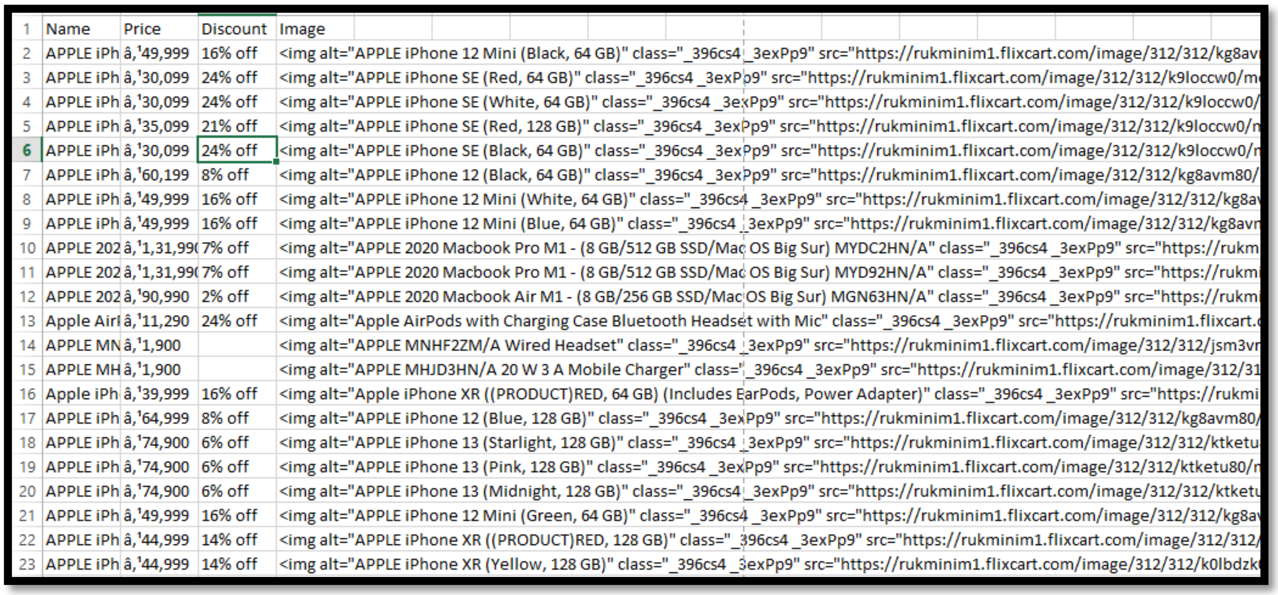


result.csv

```
Name,Price,Discount,Image
"APPLE iPhone SE (White, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (White, 128 GB)"" class=""_396cs4 _3exPp9""
"APPLE iPhone SE (Black, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (Black, 128 GB)"" class=""_396cs4 _3exPp9""
"APPLE iPhone 12 Mini (Black, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (Black, 64 GB)"" class=""_396cs4 _
"APPLE iPhone SE (Red, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (Red, 64 GB)"" class=""_396cs4 _3exPp9"" src="
"APPLE iPhone SE (White, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (White, 64 GB)"" class=""_396cs4 _3exPp9"" s
"APPLE iPhone SE (Black, 64 GB)","₹30,099",24% off,"<img alt=""APPLE iPhone SE (Black, 64 GB)"" class=""_396cs4 _3exPp9"" s
"APPLE iPhone SE (Red, 128 GB)","₹35,099",21% off,"<img alt=""APPLE iPhone SE (Red, 128 GB)"" class=""_396cs4 _3exPp9"" src
"APPLE iPhone 12 (Black, 64 GB)","₹60,199",8% off,"<img alt=""APPLE iPhone 12 (Black, 64 GB)"" class=""_396cs4 _3exPp9"" sr
APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYDC2HN/A,"₹1,31,990",7% off,"<img alt=""APPLE 2020 Macbook Pr
APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYD92HN/A,"₹1,31,990",7% off,"<img alt=""APPLE 2020 Macbook Pr
APPLE MMTN2ZM/A Wired Headset,"₹1,799",5% off,"<img alt=""APPLE MMTN2ZM/A Wired Headset"" class=""_396cs4 _3exPp9"" src=""h
Apple AirPods with Charging Case Bluetooth Headset with Mic,"₹12,499",16% off,"<img alt=""Apple AirPods with Charging Case
APPLE MHJD3HN/A 20 W 3 A Mobile Charger,"₹1,900",,"<img alt=""APPLE MHJD3HN/A 20 W 3 A Mobile Charger"" class=""_396cs4 _3e
"APPLE iPhone 12 Mini (White, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (White, 64 GB)"" class=""_396cs4 _
"APPLE iPhone 12 Mini (Blue, 64 GB)","₹49,999",16% off,"<img alt=""APPLE iPhone 12 Mini (Blue, 64 GB)"" class=""_396cs4 _3e
"APPLE iPhone 12 (Blue, 128 GB)","₹64,999",8% off,"<img alt=""APPLE iPhone 12 (Blue, 128 GB)"" class=""_396cs4 _3exPp9"" sr
"APPLE iPhone 13 (Pink, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Pink, 128 GB)"" class=""_396cs4 _3exPp9"" sr
"APPLE iPhone 13 (Starlight, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Starlight, 128 GB)"" class=""_396cs4 _3
"APPLE iPhone 13 (Midnight, 128 GB)","₹74,900",6% off,"<img alt=""APPLE iPhone 13 (Midnight, 128 GB)"" class=""_396cs4 _3ex
"APPLE iPhone XR ((PRODUCT)RED, 128 GB)","₹44,999",14% off,"<img alt=""APPLE iPhone XR ((PRODUCT)RED, 128 GB)"" class=""_39
"APPLE iPhone XR (Yellow, 128 GB)","₹44,999",14% off,"<img alt=""APPLE iPhone XR (Yellow, 128 GB)"" class=""_396cs4 _3exPp9
APPLE 2020 Macbook Air M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MGN63HN/A,"₹90,990",2% off,"<img alt=""APPLE 2020 Macbook Air
APPLE 2020 Macbook Pro M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MYD82HN/A,"₹1,12,990",8% off,"<img alt=""APPLE 2020 Macbook Pr
```

```
PS C:\Users\LENOVO\Downloads\19BCE2555> python main.py
Scraping Data done...
Successfully Dumped at result.csv
PS C:\Users\LENOVO\Downloads\19BCE2555>
```

Here is the list of data that is being scrapped from the website and dumped into result.csv file and which is formed in the folder of the code after running the code using beautifulsoup.



Name	Price	Discount	Image
APPLE iPh	â,¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8av
APPLE iPh	â,¹30,099	24% off	<img alt="APPLE iPhone SE (Red, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/m
APPLE iPh	â,¹30,099	24% off	<img alt="APPLE iPhone SE (White, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/
APPLE iPh	â,¹35,099	21% off	<img alt="APPLE iPhone SE (Red, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/
APPLE iPh	â,¹30,099	24% off	<img alt="APPLE iPhone SE (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k9loccw0/
APPLE iPh	â,¹60,199	8% off	<img alt="APPLE iPhone 12 (Black, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/
APPLE iPh	â,¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (White, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8a
APPLE iPh	â,¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Blue, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8av
APPLE 202	â,¹1,31,99	7% off	<img alt="APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYDC2HN/A" class="_396cs4 _3exPp9" src="https://rukm
APPLE 202	â,¹1,31,99	7% off	<img alt="APPLE 2020 Macbook Pro M1 - (8 GB/512 GB SSD/Mac OS Big Sur) MYD92HN/A" class="_396cs4 _3exPp9" src="https://rukmi
APPLE 202	â,¹90,990	2% off	<img alt="APPLE 2020 Macbook Air M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MGN63HN/A" class="_396cs4 _3exPp9" src="https://rukmi
Apple Airl	â,¹11,290	24% off	<img alt="Apple AirPods with Charging Case Bluetooth Headset with Mic" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.
APPLE MN	â,¹1,900		<img alt="APPLE MNHF2ZM/A Wired Headset" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/jsm3v
APPLE MH	â,¹1,900		<img alt="APPLE MHJD3HN/A 20 W 3 A Mobile Charger" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/31
Apple iPh	â,¹39,999	16% off	<img alt="Apple iPhone XR ((PRODUCT)RED, 64 GB) (Includes EarPods, Power Adapter)" class="_396cs4 _3exPp9" src="https://rukmi
APPLE iPh	â,¹64,999	8% off	<img alt="APPLE iPhone 12 (Blue, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8avm80/
APPLE iPh	â,¹74,900	6% off	<img alt="APPLE iPhone 13 (Starlight, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu
APPLE iPh	â,¹74,900	6% off	<img alt="APPLE iPhone 13 (Pink, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu80/
APPLE iPh	â,¹74,900	6% off	<img alt="APPLE iPhone 13 (Midnight, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/ktketu
APPLE iPh	â,¹49,999	16% off	<img alt="APPLE iPhone 12 Mini (Green, 64 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/kg8a
APPLE iPh	â,¹44,999	14% off	<img alt="APPLE iPhone XR ((PRODUCT)RED, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/
APPLE iPh	â,¹44,999	14% off	<img alt="APPLE iPhone XR (Yellow, 128 GB)" class="_396cs4 _3exPp9" src="https://rukminim1.flixcart.com/image/312/312/k0lbdzk

This is the view of Excel file in which the data is being stored.

Web Crawling Using Scrapy

Question

Experiment 4 b

1. Use BeautifulSoup or Scrapy to crawl any one of the E-commerce website of your choice and perform the same. The following information needs to be extracted from the page: (Choose any one product : e.g laptop, smartphone … etc)

a) Product Name

b) Product price

c) Product discount

d) Product image

Problem statement:

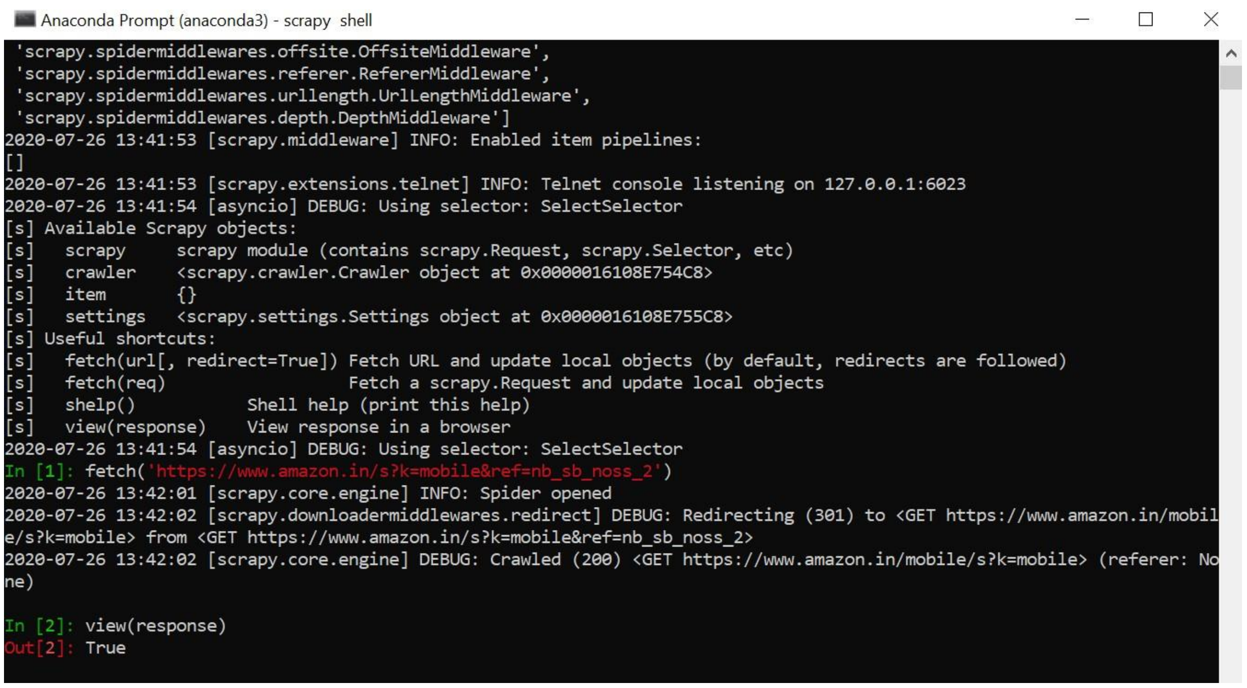
To Crawl any of the E-commerce website and extract the data from the page like Name, Price, Discount and image of the Product using only Scrapy.

Procedure:

- Firstly we install scrapy package with “pip install scrapy” in anaconda prompt
- Later, we can start Shell by “scrapy shell”
- Then Crawler run in the shell by use of the fetch and using view(response) to view fetched data.
- An object should be created for the scrapper by “scrapy startproject mobile”
- Create folder named “mobile”and move to that particular folder using command“cd mobile”
- Create a python(.py) file inside the “spider” folder by using the command “scrapy genspider ..url..”
- Here I scrapped data of amazons mobile as by product so the same url is pasted here.

- Then python code is written in the file.
- We can view the output in the terminal on typing “scrapy crawl ..name.. “ on teriminal
- Finally it is exported as csv file using command “scrapy crawl mob -o data.csv”.

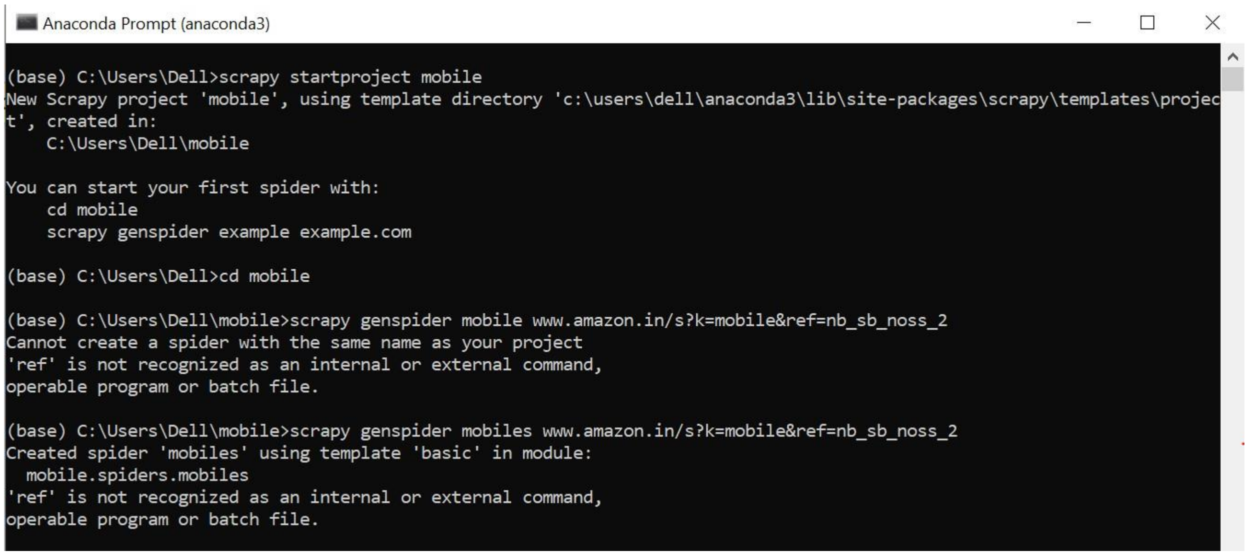
Installing Scarpy in Anaconda .



Anaconda Prompt (anaconda3) - scrapy shell

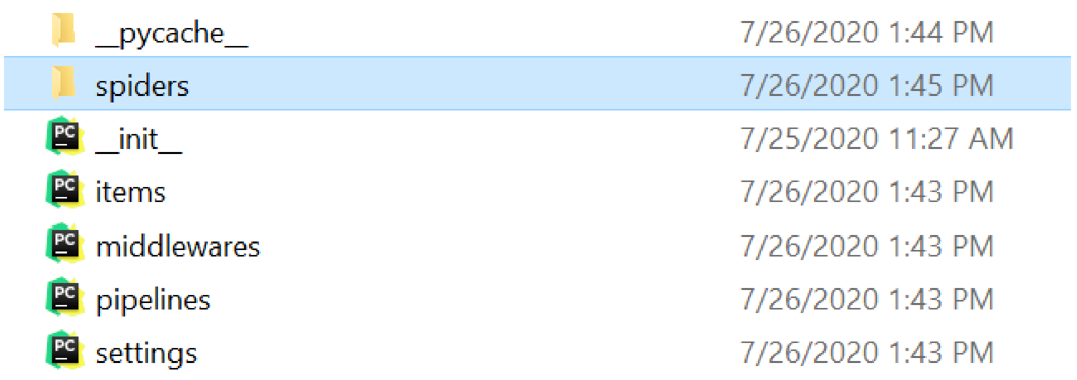
```
 'scrapy.spidermiddlewares.offsite.OffsiteMiddleware',
 'scrapy.spidermiddlewares.referer.RefererMiddleware',
 'scrapy.spidermiddlewares.urllength.UrlLengthMiddleware',
 'scrapy.spidermiddlewares.depth.DepthMiddleware']
2020-07-26 13:41:53 [scrapy.middleware] INFO: Enabled item pipelines:
[]
2020-07-26 13:41:53 [scrapy.extensions.telnet] INFO: Telnet console listening on 127.0.0.1:6023
2020-07-26 13:41:54 [asyncio] DEBUG: Using selector: SelectSelector
[s] Available Scrapy objects:
[s]   scrapy     scrapy module (contains scrapy.Request, scrapy.Selector, etc)
[s]   crawler    <scrapy.crawler.Crawler object at 0x0000016108E754C8>
[s]   item       {}
[s]   settings   <scrapy.settings.Settings object at 0x0000016108E755C8>
[s] Useful shortcuts:
[s]   fetch(url[, redirect=True]) Fetch URL and update local objects (by default, redirects are followed)
[s]   fetch(req)                  Fetch a scrapy.Request and update local objects
[s]   shelp()           Shell help (print this help)
[s]   view(response)    View response in a browser
2020-07-26 13:41:54 [asyncio] DEBUG: Using selector: SelectSelector
In [1]: fetch('https://www.amazon.in/s?k=mobile&ref=nb_sb_noss_2')
2020-07-26 13:42:01 [scrapy.core.engine] INFO: Spider opened
2020-07-26 13:42:02 [scrapy.downloadermiddlewares.redirect] DEBUG: Redirecting (301) to <GET https://www.amazon.in/mobil
e/s?k=mobile> from <GET https://www.amazon.in/s?k=mobile&ref=nb_sb_noss_2>
2020-07-26 13:42:02 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://www.amazon.in/mobile/s?k=mobile> (referer: No
ne)

In [2]: view(response)
Out[2]: True
```

#creating scrapy project as name mobiles:

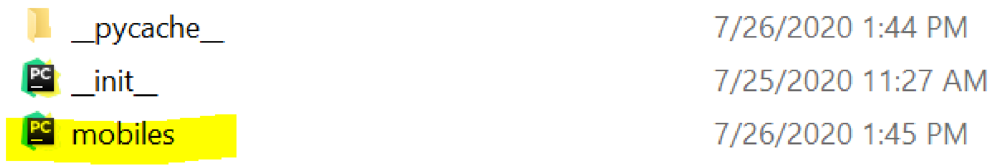


Anaconda Prompt (anaconda3)

```
(base) C:\Users\Dell>scrapy startproject mobile
New Scrapy project 'mobile', using template directory 'c:\users\dell\anaconda3\lib\site-packages\scrapy\templates\projec
t', created in:
    C:\Users\Dell\mobile

You can start your first spider with:
    cd mobile
    scrapy genspider example example.com

(base) C:\Users\Dell>cd mobile

(base) C:\Users\Dell\mobile>scrapy genspider mobile www.amazon.in/s?k=mobile&ref=nb_sb_noss_2
Cannot create a spider with the same name as your project
'ref' is not recognized as an internal or external command,
operable program or batch file.

(base) C:\Users\Dell\mobile>scrapy genspider mobiles www.amazon.in/s?k=mobile&ref=nb_sb_noss_2
Created spider 'mobiles' using template 'basic' in module:
  mobile.spiders.mobiles
'ref' is not recognized as an internal or external command,
operable program or batch file.
```



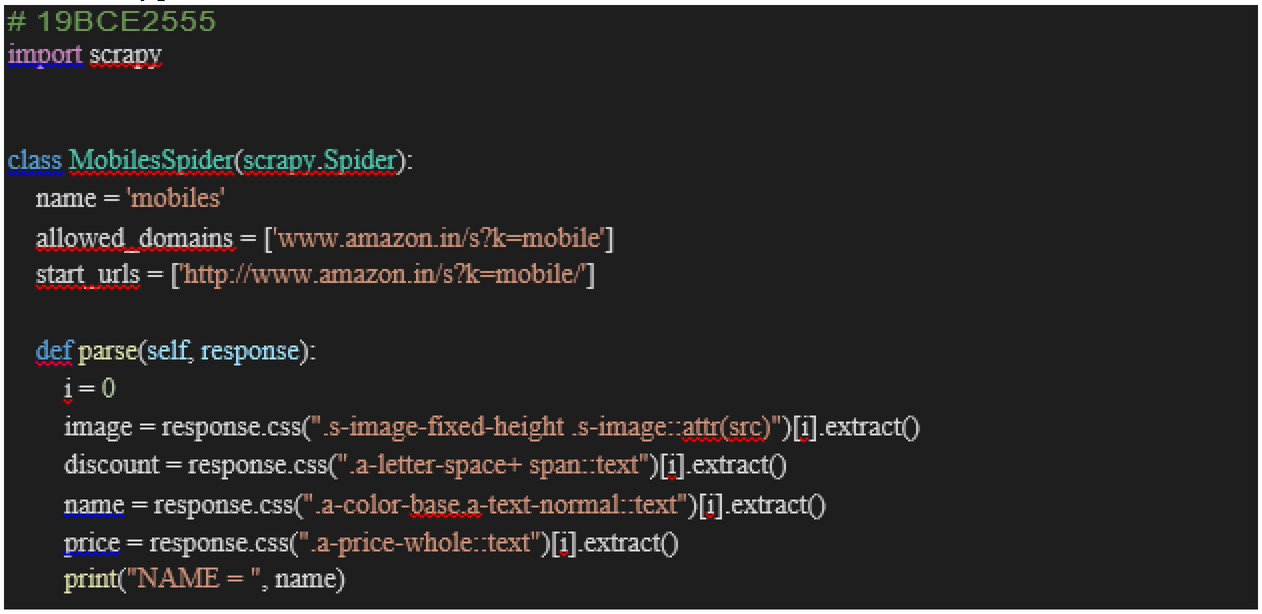
__pycache__	7/26/2020 1:44 PM
spiders	7/26/2020 1:45 PM
__init__	7/25/2020 11:27 AM
items	7/26/2020 1:43 PM
middlewares	7/26/2020 1:43 PM
pipelines	7/26/2020 1:43 PM
settings	7/26/2020 1:43 PM



__pycache__	7/26/2020 1:44 PM
__init__	7/25/2020 11:27 AM
mobiles	7/26/2020 1:45 PM

Code:

#mobiles.py:

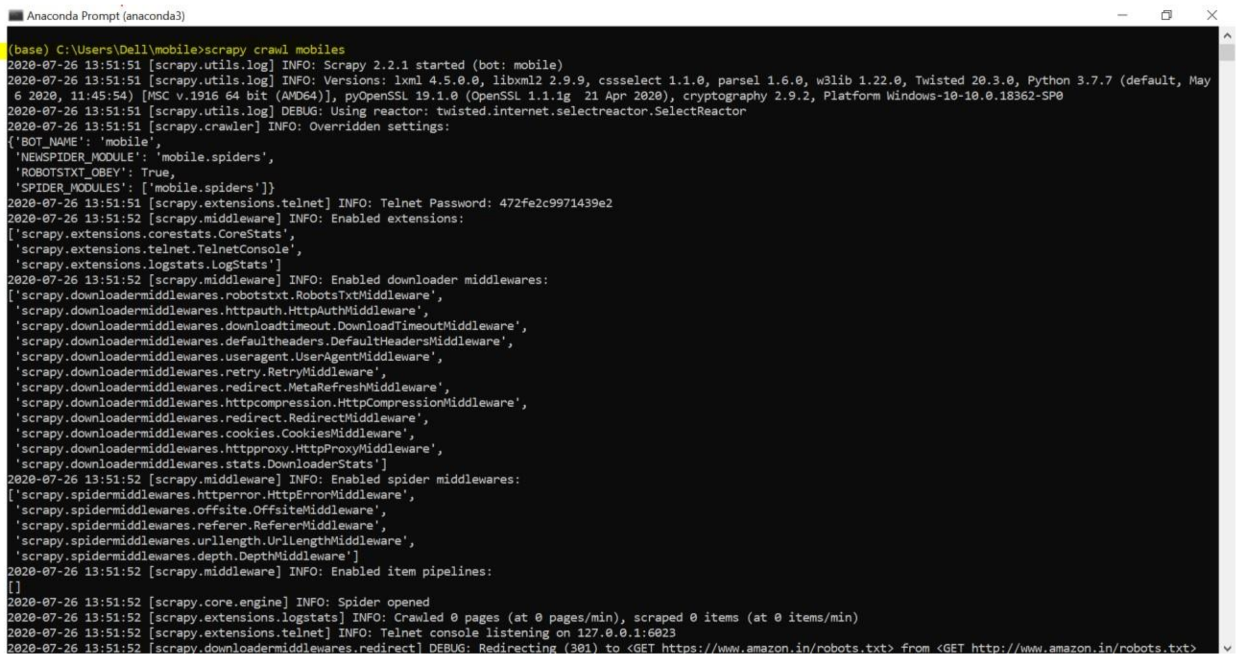


```
# 19BCE2555
import scrapy


class MobilesSpider(scrapy.Spider):
    name = 'mobiles'
    allowed_domains = ['www.amazon.in/s?k=mobile']
    start_urls = ['http://www.amazon.in/s?k=mobile/']

    def parse(self, response):
        i = 0
        image = response.css(".s-image-fixed-height .s-image::attr(src)")[i].extract()
        discount = response.css(".a-letter-space+ span::text")[i].extract()
        name = response.css(".a-color-base.a-text-normal::text")[i].extract()
        price = response.css(".a-price-whole::text")[i].extract()
        print("NAME = ", name)
        print("PRICE = ", price)
        print("DISCOUNT", discount)
        print("image url = ", image)
        f = open('img.jpg', 'wb')
        f.write(urllib.request.urlopen(image).read())
```

Items.py

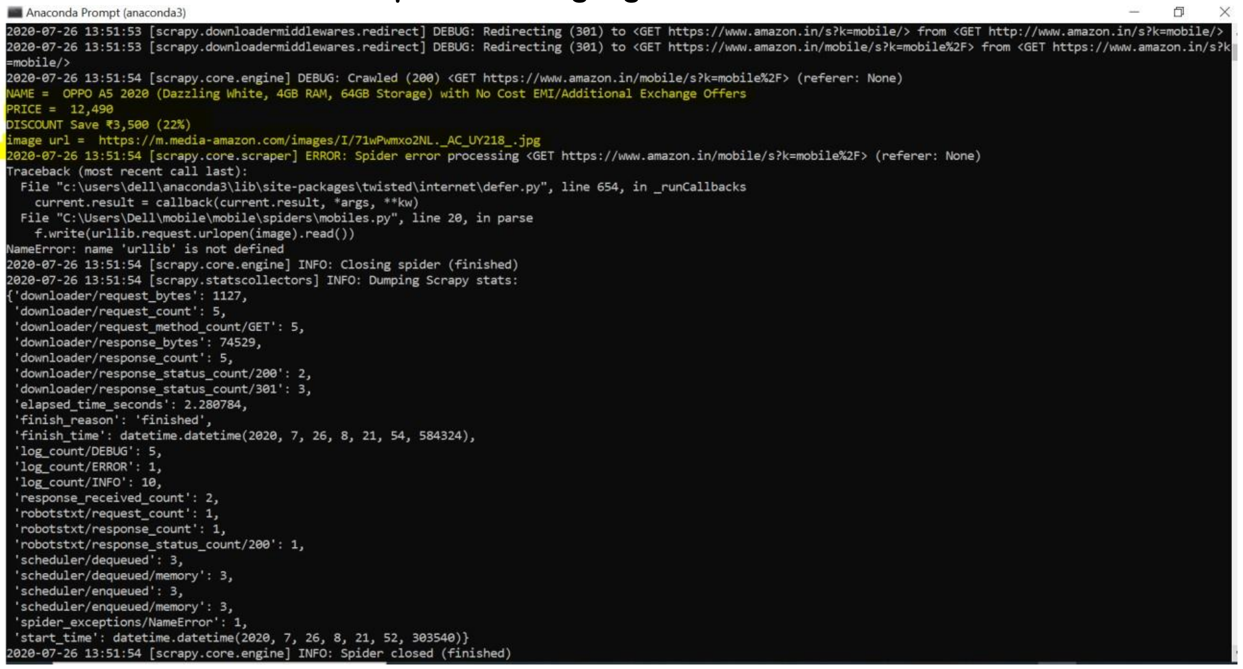
```
# Define here the models for your scraped items
# 19BCE2555
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/items.html

import scrapy


class MobileItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    product_name = scrapy.Field()
    product_price = scrapy.Field()
    product_discount = scrapy.Field()
    product_image= scrapy.Field()
    pass
```

Output



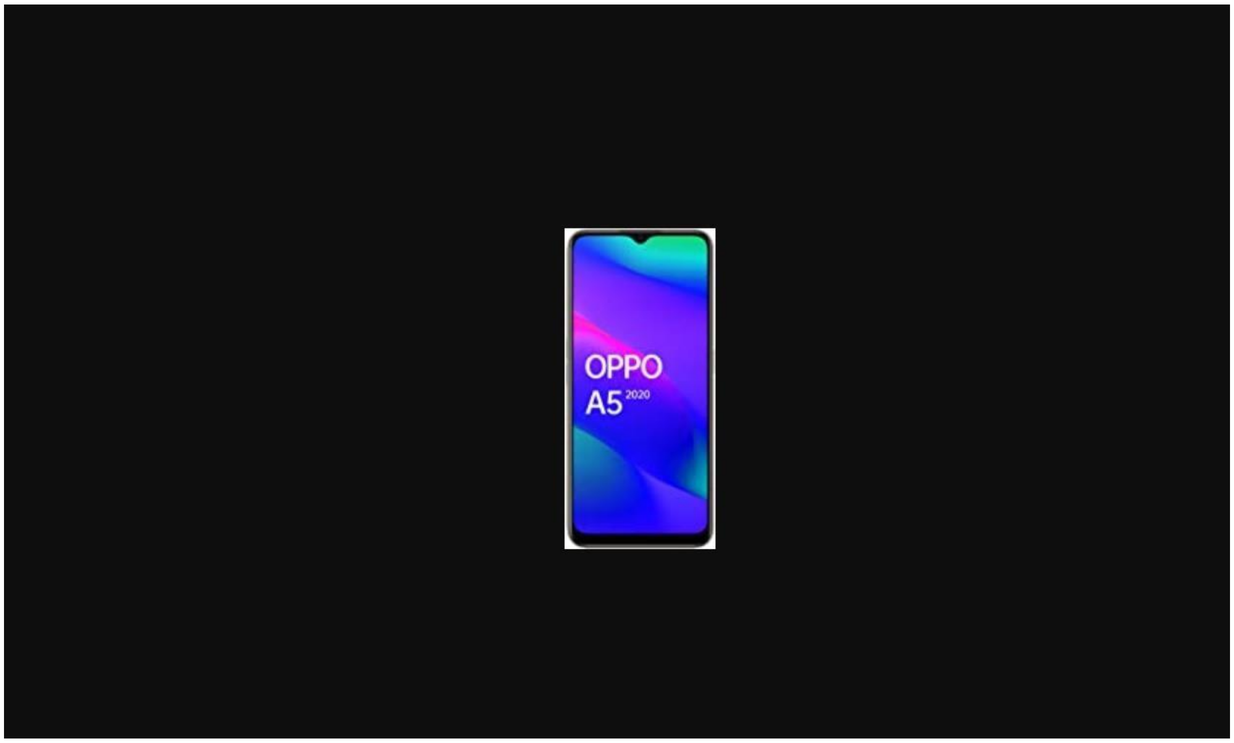
```
Anaconda Prompt (anaconda3)
(base) C:\Users\Dell\mobile>scrapy crawl mobiles
2020-07-26 13:51:51 [scrapy.utils.log] INFO: Scrapy 2.2.1 started (bot: mobile)
2020-07-26 13:51:51 [scrapy.utils.log] INFO: Versions: lxml 4.5.0.0, libxml2 2.9.9, cssselect 1.1.0, parsel 1.6.0, w3lib 1.22.0, Twisted 20.3.0, Python 3.7.7 (default, May
 6 2020, 11:45:54) [MSC v.1916 64 bit (AMD64)], pyOpenSSL 19.1.0 (OpenSSL 1.1.1g  21 Apr 2020), cryptography 2.9.2, Platform Windows-10-10.0.18362-SP0
2020-07-26 13:51:51 [scrapy.utils.log] DEBUG: Using reactor: twisted.internet.selectreactor.SelectReactor
2020-07-26 13:51:51 [scrapy.crawler] INFO: Overridden settings:
{'BOT_NAME': 'mobile',
 'NEWSPIDER_MODULE': 'mobile.spiders',
 'ROBOTSTXT_OBEY': True,
 'SPIDER_MODULES': ['mobile.spiders']}
2020-07-26 13:51:51 [scrapy.extensions.telnet] INFO: Telnet Password: 472fe2c9971439e2
2020-07-26 13:51:52 [scrapy.middleware] INFO: Enabled extensions:
['scrapy.extensions.corestats.CoreStats',
 'scrapy.extensions.telnet.TelnetConsole',
 'scrapy.extensions.logstats.LogStats']
2020-07-26 13:51:52 [scrapy.middleware] INFO: Enabled downloader middlewares:
['scrapy.downloadermiddlewares.robotstxt.RobotsTxtMiddleware',
 'scrapy.downloadermiddlewares.httpauth.HttpAuthMiddleware',
 'scrapy.downloadermiddlewares.downloadtimeout.DownloadTimeoutMiddleware',
 'scrapy.downloadermiddlewares.defaultheaders.DefaultHeadersMiddleware',
 'scrapy.downloadermiddlewares.useragent.UserAgentMiddleware',
 'scrapy.downloadermiddlewares.retry.RetryMiddleware',
 'scrapy.downloadermiddlewares.redirect.MetaRefreshMiddleware',
 'scrapy.downloadermiddlewares.httpcompression.HttpCompressionMiddleware',
 'scrapy.downloadermiddlewares.redirect.RedirectMiddleware',
 'scrapy.downloadermiddlewares.cookies.CookiesMiddleware',
 'scrapy.downloadermiddlewares.httpproxy.HttpProxyMiddleware',
 'scrapy.downloadermiddlewares.stats.DownloaderStats']
2020-07-26 13:51:52 [scrapy.middleware] INFO: Enabled spider middlewares:
['scrapy.spidermiddlewares.httperror.HttpErrorMiddleware',
 'scrapy.spidermiddlewares.offsite.OffsiteMiddleware',
 'scrapy.spidermiddlewares.referer.RefererMiddleware',
 'scrapy.spidermiddlewares.urllength.UrlLengthMiddleware',
 'scrapy.spidermiddlewares.depth.DepthMiddleware']
2020-07-26 13:51:52 [scrapy.middleware] INFO: Enabled item pipelines:
[]
2020-07-26 13:51:52 [scrapy.core.engine] INFO: Spider opened
2020-07-26 13:51:52 [scrapy.extensions.logstats] INFO: Crawled 0 pages (at 0 pages/min), scraped 0 items (at 0 items/min)
2020-07-26 13:51:52 [scrapy.extensions.telnet] INFO: Telnet console listening on 127.0.0.1:6023
2020-07-26 13:51:52 [scrapy.downloadermiddlewares.redirect] DEBUG: Redirecting (301) to <GET https://www.amazon.in/robots.txt> from <GET http://www.amazon.in/robots.txt>
```

The basic information of the product is highlighted below



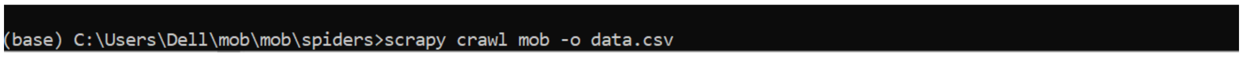
```
Anaconda Prompt (anaconda3)
2020-07-26 13:51:53 [scrapy.downloadermiddlewares.redirect] DEBUG: Redirecting (301) to <GET https://www.amazon.in/s?k=mobile/> from <GET http://www.amazon.in/s?k=mobile/>
2020-07-26 13:51:53 [scrapy.downloadermiddlewares.redirect] DEBUG: Redirecting (301) to <GET https://www.amazon.in/mobile/s?k=mobile%2F> from <GET https://www.amazon.in/s?k
=mobile/>
2020-07-26 13:51:54 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://www.amazon.in/mobile/s?k=mobile%2F> (referer: None)
NAME =  OPPO A5 2020 (Dazzling White, 4GB RAM, 64GB Storage) with No Cost EMI/Additional Exchange Offers
PRICE =  12,490
DISCOUNT Save ₹3,500 (22%)
image url =  https://m.media-amazon.com/images/I/71wPwmxo2NL._AC_UY218_.jpg
2020-07-26 13:51:54 [scrapy.core.scraper] ERROR: Spider error processing <GET https://www.amazon.in/mobile/s?k=mobile%2F> (referer: None)
Traceback (most recent call last):
  File "c:\users\dell\anaconda3\lib\site-packages\twisted\internet\defer.py", line 654, in _runCallbacks
    current.result = callback(current.result, *args, **kw)
  File "C:\Users\Dell\mobile\mobile\spiders\mobiles.py", line 20, in parse
    f.write(urllib.request.urlopen(image).read())
NameError: name 'urllib' is not defined
2020-07-26 13:51:54 [scrapy.core.engine] INFO: Closing spider (finished)
2020-07-26 13:51:54 [scrapy.statscollectors] INFO: Dumping Scrapy stats:
{'downloader/request_bytes': 1127,
 'downloader/request_count': 5,
 'downloader/request_method_count/GET': 5,
 'downloader/response_bytes': 74529,
 'downloader/response_count': 5,
 'downloader/response_status_count/200': 2,
 'downloader/response_status_count/301': 3,
 'elapsed_time_seconds': 2.280784,
 'finish_reason': 'finished',
 'finish_time': datetime.datetime(2020, 7, 26, 8, 21, 54, 584324),
 'log_count/DEBUG': 5,
 'log_count/ERROR': 1,
 'log_count/INFO': 10,
 'response_received_count': 2,
 'robotstxt/request_count': 1,
 'robotstxt/response_count': 1,
 'robotstxt/response_status_count/200': 1,
 'scheduler/dequeued': 3,
 'scheduler/dequeued/memory': 3,
 'scheduler/enqueued': 3,
 'scheduler/enqueued/memory': 3,
 'spider_exceptions/NameError': 1,
 'start_time': datetime.datetime(2020, 7, 26, 8, 21, 52, 303540)}
2020-07-26 13:51:54 [scrapy.core.engine] INFO: Spider closed (finished)
```

By clicking on the link extracted from the webpage we get the following image: Link: https://m.media-amazon.com/images/I/71wPwmxo2NL._AC_UY218_.jpg

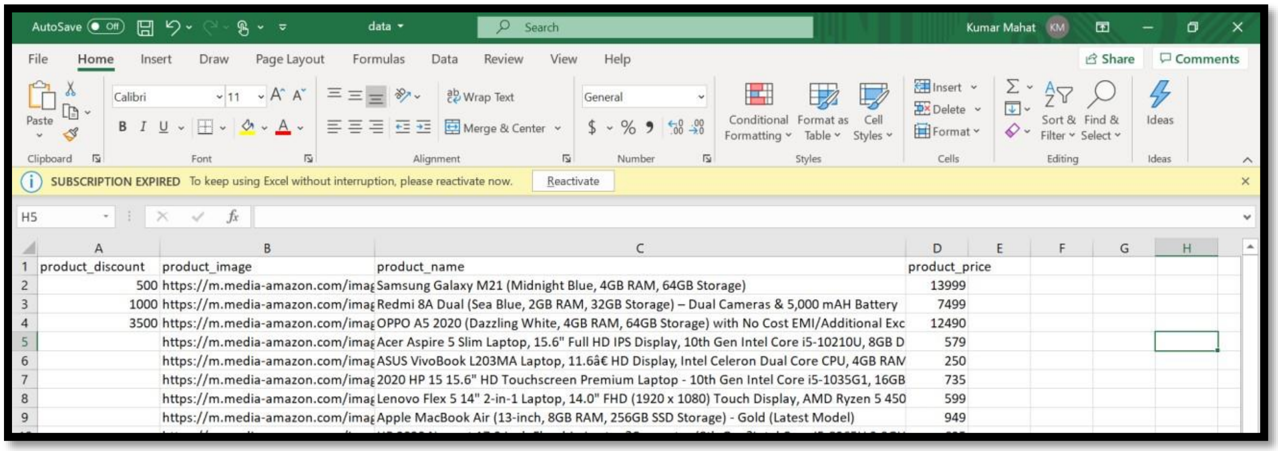


Results and Output

#Exporting scrapped data as csv



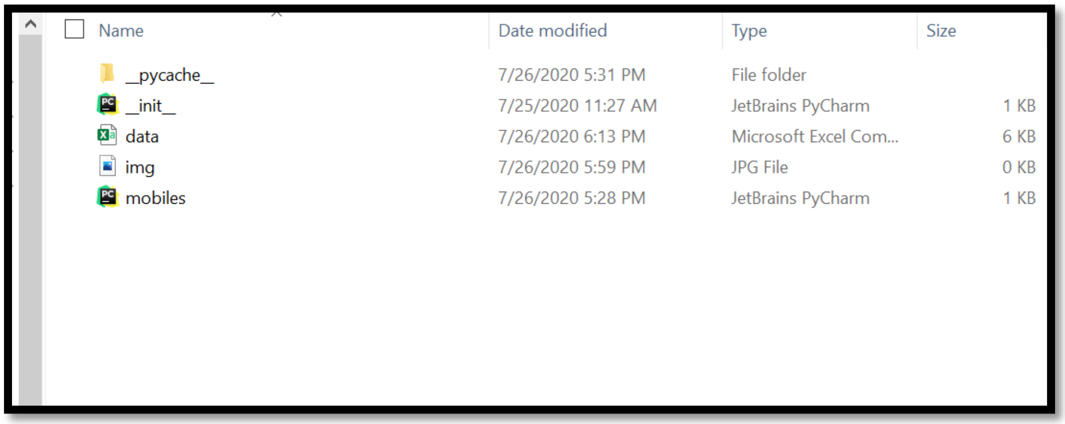
```
(base) C:\Users\Dell\mob\mob\spiders>scrapy crawl mob -o data.csv
```



SUBSCRIPTION EXPIRED To keep using Excel without interruption, please reactivate now.

	A	B	C	D
1	product_discount	product_image	product_name	product_price
2	500	https://m.media-amazon.com/imag	Samsung Galaxy M21 (Midnight Blue, 4GB RAM, 64GB Storage)	13999
3	1000	https://m.media-amazon.com/imag	Redmi 8A Dual (Sea Blue, 2GB RAM, 32GB Storage) – Dual Cameras & 5,000 mAH Battery	7499
4	3500	https://m.media-amazon.com/imag	OPPO A5 2020 (Dazzling White, 4GB RAM, 64GB Storage) with No Cost EMI/Additional Exc	12490
5		https://m.media-amazon.com/imag	Acer Aspire 5 Slim Laptop, 15.6" Full HD IPS Display, 10th Gen Intel Core i5-10210U, 8GB D	579
6		https://m.media-amazon.com/imag	ASUS VivoBook L203MA Laptop, 11.6â€ HD Display, Intel Celeron Dual Core CPU, 4GB RAM	250
7		https://m.media-amazon.com/imag	2020 HP 15 15.6" HD Touchscreen Premium Laptop - 10th Gen Intel Core i5-1035G1, 16GB	735
8		https://m.media-amazon.com/imag	Lenovo Flex 5 14" 2-in-1 Laptop, 14.0" FHD (1920 x 1080) Touch Display, AMD Ryzen 5 450	599
9		https://m.media-amazon.com/imag	Apple MacBook Air (13-inch, 8GB RAM, 256GB SSD Storage) - Gold (Latest Model)	949

We can see that the data is scrapped and it is dumped in excel sheet



Name	Date modified	Type	Size
__pycache__	7/26/2020 5:31 PM	File folder	
__init__	7/25/2020 11:27 AM	JetBrains PyCharm	1 KB
data	7/26/2020 6:13 PM	Microsoft Excel Com...	6 KB
img	7/26/2020 5:59 PM	JPG File	0 KB
mobiles	7/26/2020 5:28 PM	JetBrains PyCharm	1 KB

Encoding

Question

Experiment 5

Write a python program to perform the following encoding and decoding for the EVEN numbers between 1-20

1) Unary
2) Elias Gamma
3) Elias Delta
4) Golomb (b=10)

Problem statement:

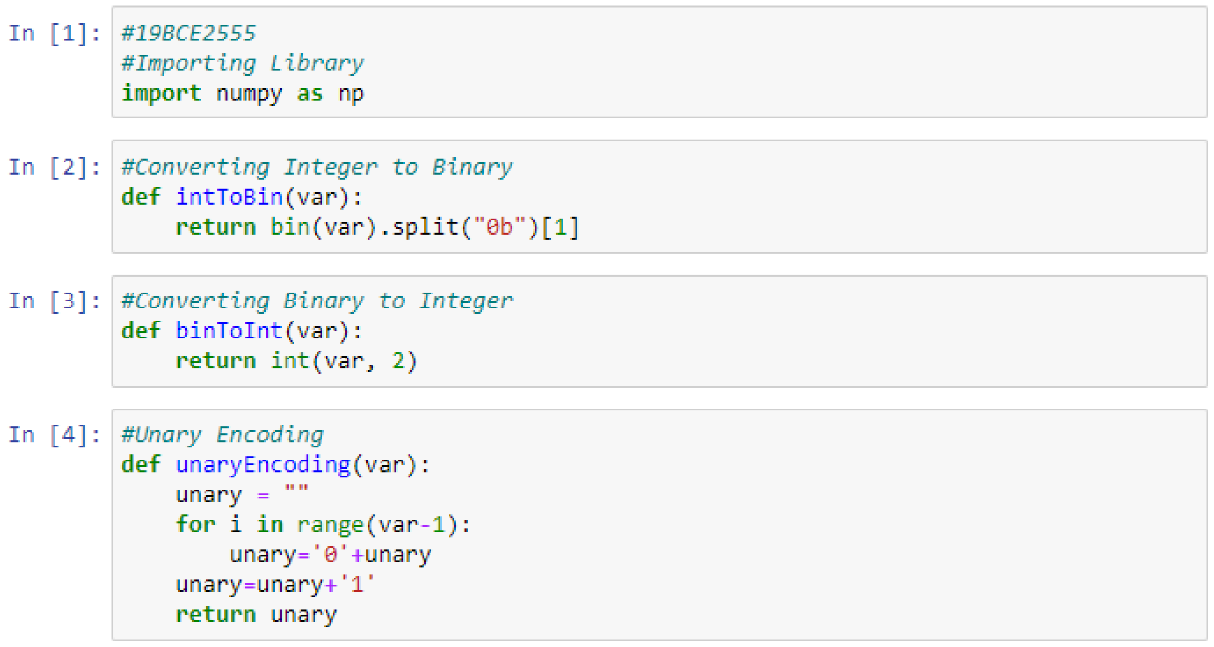
To perform the following encoding and decoding for the EVEN numbers between 1-20

Procedure:

- Firstly, we will import the necessary numpy library to use mathematical functions like logarithm in our code.
- Next, We will create 2 functions, one to convert integer to binary and the other for converting binary to integer.
- Next, We will write respective functions for each and every method given.
- The functions that are corresponding to Unary Encoding, Unary Decoding, Elias Gamma Encoding, Elias Gamma Decoding, Elias Delta Encoding, Elias Delta Decoding, Golomb Encoding and Golomb Decoding.

- In main program, we will run a loop from numbers 2 to 21 with a jump of 2 to in order to get even numbers in the range from 1-20.
- We will finally perform the above functions to each of the iterators in the above loop

Code:

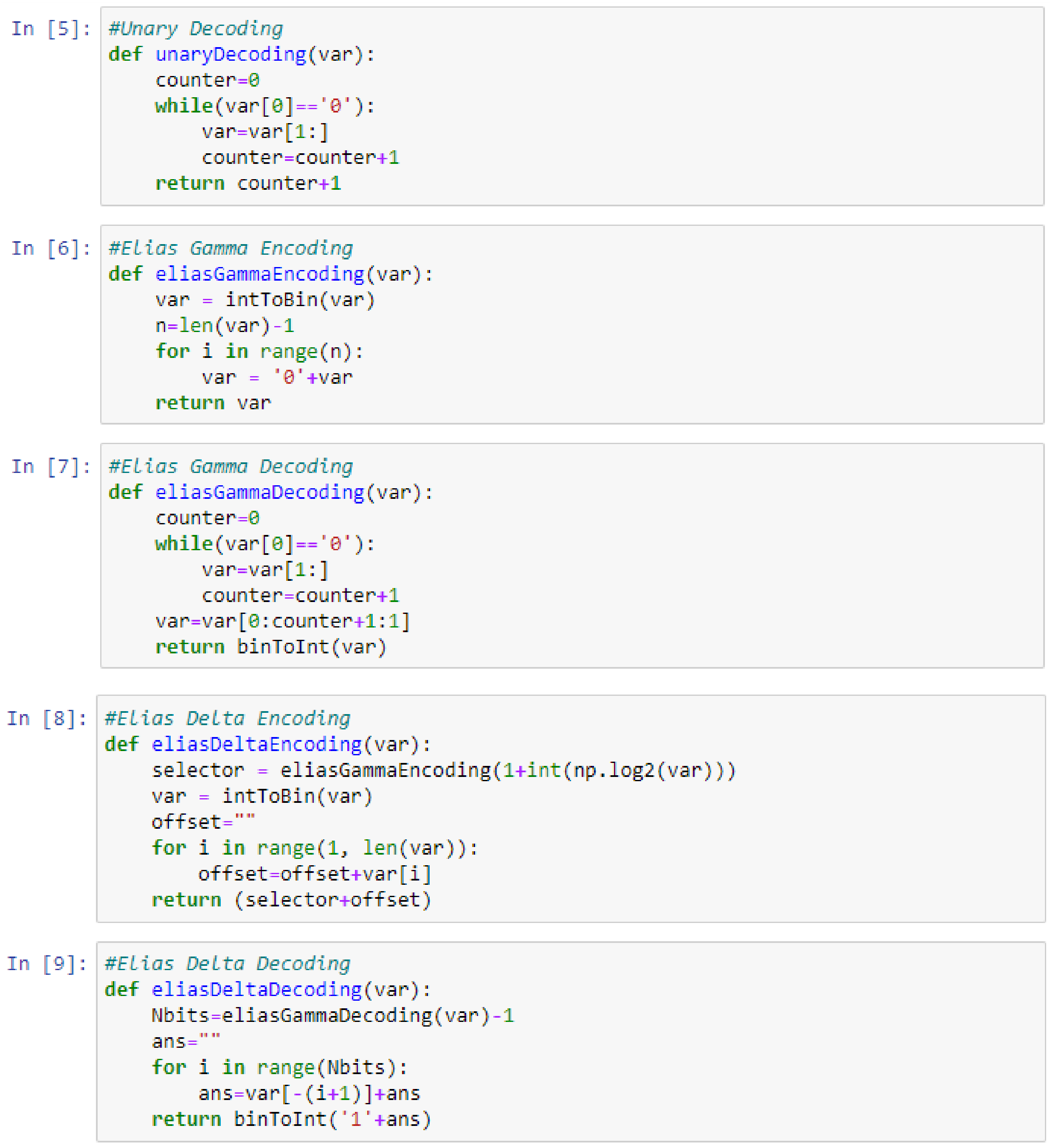


```
In [1]: #19BCE2555
        #Importing Library
        import numpy as np
```

```
In [2]: #Converting Integer to Binary
        def intToBin(var):
            return bin(var).split("0b")[1]
```

```
In [3]: #Converting Binary to Integer
        def binToInt(var):
            return int(var, 2)
```

```
In [4]: #Unary Encoding
        def unaryEncoding(var):
            unary = ""
            for i in range(var-1):
                unary='0'+unary
            unary=unary+'1'
            return unary
```



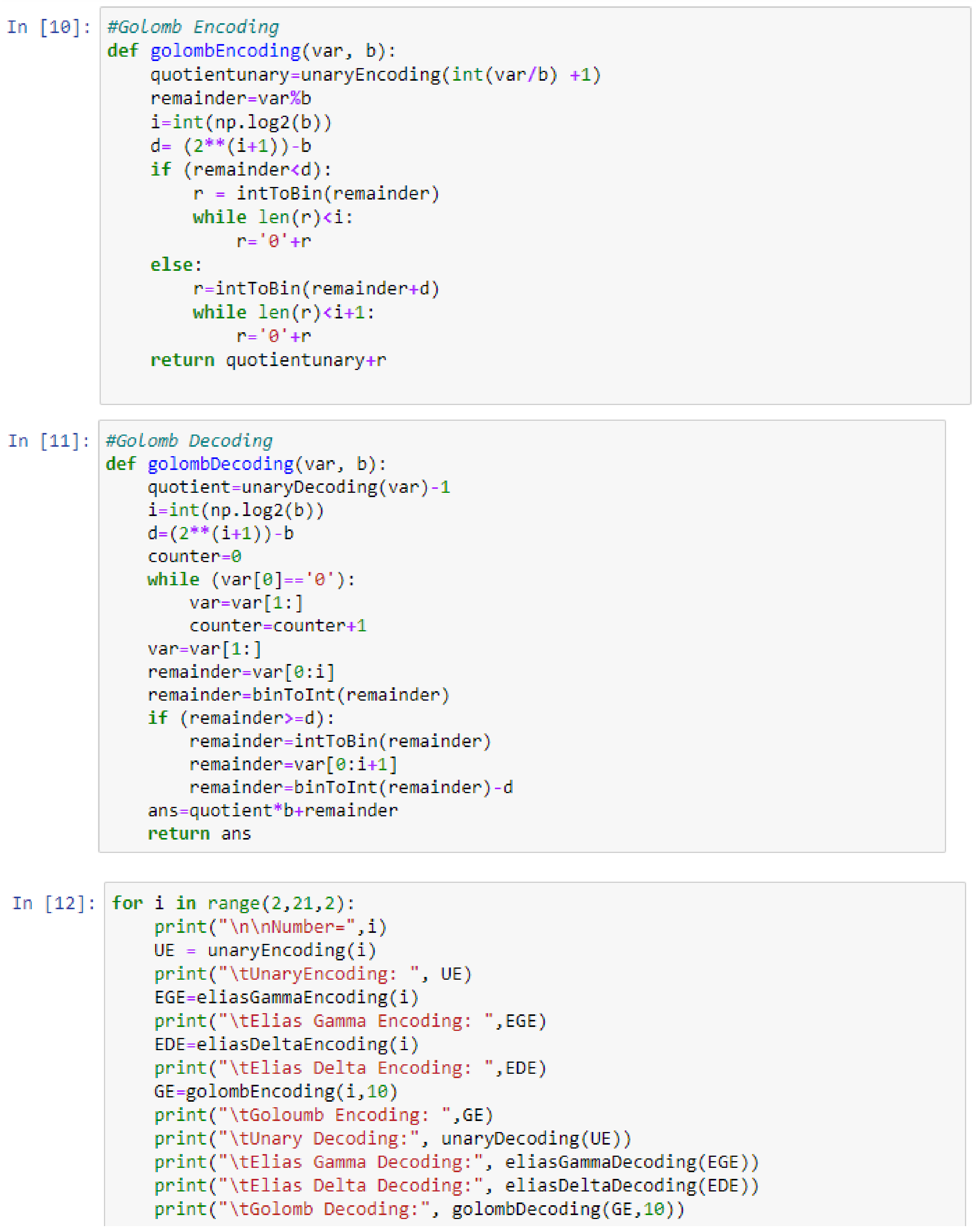
```
In [5]: #Unary Decoding
        def unaryDecoding(var):
            counter=0
            while(var[0]=='0'):
                var=var[1:]
                counter=counter+1
            return counter+1
```

```
In [6]: #Elias Gamma Encoding
        def eliasGammaEncoding(var):
            var = intToBin(var)
            n=len(var)-1
            for i in range(n):
                var = '0'+var
            return var
```

```
In [7]: #Elias Gamma Decoding
        def eliasGammaDecoding(var):
            counter=0
            while(var[0]=='0'):
                var=var[1:]
                counter=counter+1
            var=var[0:counter+1:1]
            return binToInt(var)
```

```
In [8]: #Elias Delta Encoding
        def eliasDeltaEncoding(var):
            selector = eliasGammaEncoding(1+int(np.log2(var)))
            var = intToBin(var)
            offset=""
            for i in range(1, len(var)):
                offset=offset+var[i]
            return (selector+offset)
```

```
In [9]: #Elias Delta Decoding
        def eliasDeltaDecoding(var):
            Nbits=eliasGammaDecoding(var)-1
            ans=""
            for i in range(Nbits):
                ans=var[-(i+1)]+ans
            return binToInt('1'+ans)
```

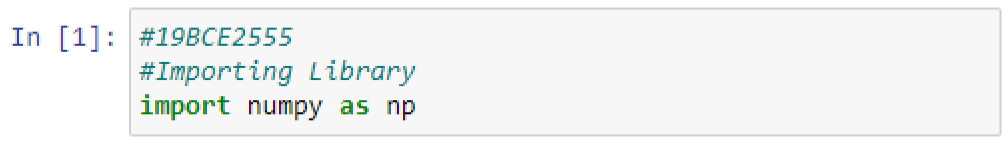


```
In [10]: #Golomb Encoding
         def golombEncoding(var, b):
             quotientunary=unaryEncoding(int(var/b) +1)
             remainder=var%b
             i=int(np.log2(b))
             d= (2**(i+1))-b
             if (remainder<d):
                 r = intToBin(remainder)
                 while len(r)<i:
                     r='0'+r
             else:
                 r=intToBin(remainder+d)
                 while len(r)<i+1:
                     r='0'+r
             return quotientunary+r
```

```
In [11]: #Golomb Decoding
         def golombDecoding(var, b):
             quotient=unaryDecoding(var)-1
             i=int(np.log2(b))
             d=(2**(i+1))-b
             counter=0
             while (var[0]=='0'):
                 var=var[1:]
                 counter=counter+1
             var=var[1:]
             remainder=var[0:i]
             remainder=binToInt(remainder)
             if (remainder>=d):
                 remainder=intToBin(remainder)
                 remainder=var[0:i+1]
                 remainder=binToInt(remainder)-d
             ans=quotient*b+remainder
             return ans
```

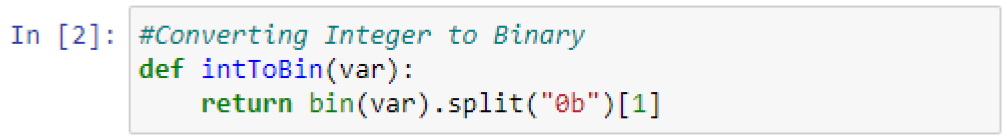
```
In [12]: for i in range(2,21,2):
             print("\n\nNumber=",i)
             UE = unaryEncoding(i)
             print("\tUnaryEncoding: ", UE)
             EGE=eliasGammaEncoding(i)
             print("\tElias Gamma Encoding: ",EGE)
             EDE=eliasDeltaEncoding(i)
             print("\tElias Delta Encoding: ",EDE)
             GE=golombEncoding(i,10)
             print("\tGoloumb Encoding: ",GE)
             print("\tUnary Decoding:", unaryDecoding(UE))
             print("\tElias Gamma Decoding:", eliasGammaDecoding(EGE))
             print("\tElias Delta Decoding:", eliasDeltaDecoding(EDE))
             print("\tGolomb Decoding:", golombDecoding(GE,10))
```

Code Snippets and Outputs:

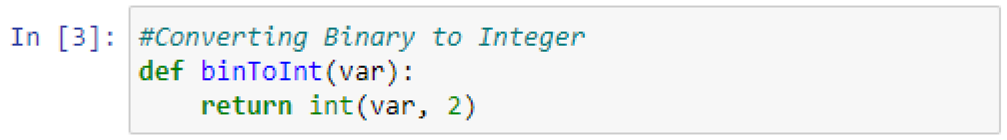


```
In [1]: #19BCE2555
        #Importing Library
        import numpy as np
```

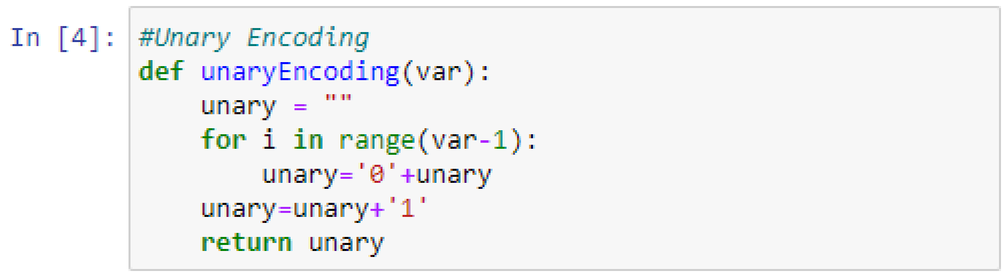
Here we are importing the libraries that are required.



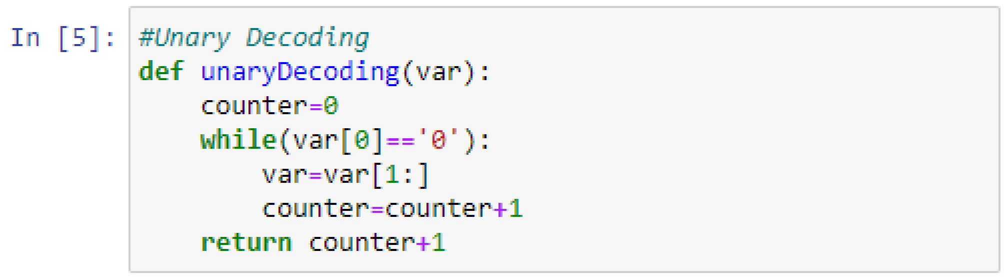
```
In [2]: #Converting Integer to Binary
        def intToBin(var):
            return bin(var).split("0b")[1]
```



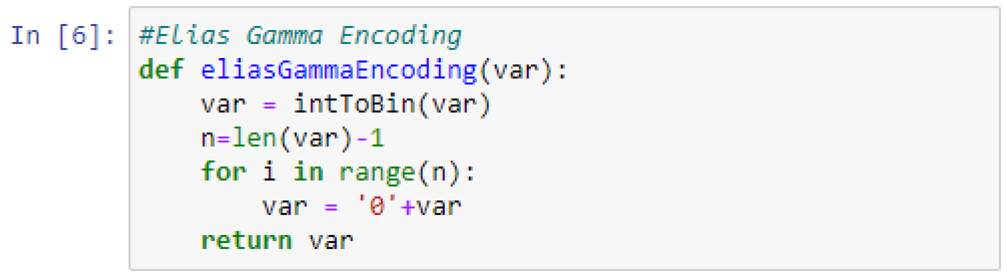
```
In [3]: #Converting Binary to Integer
        def binToInt(var):
            return int(var, 2)
```



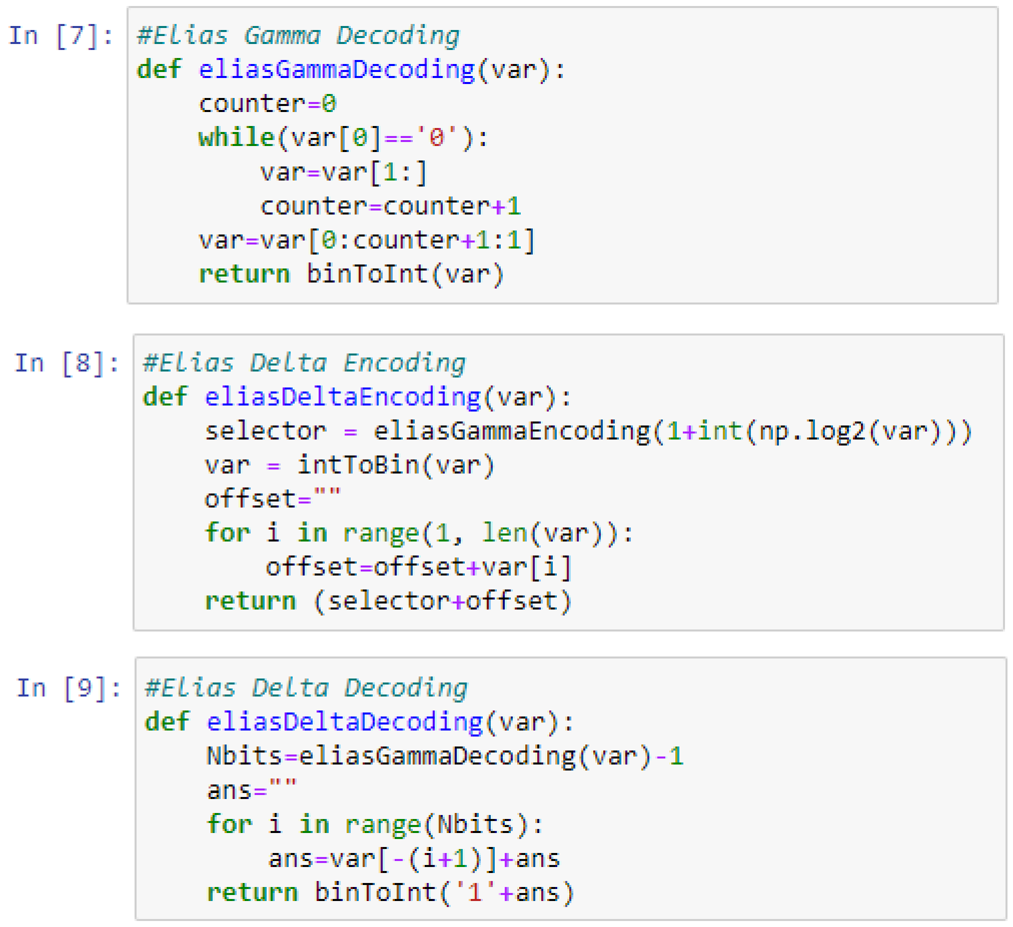
```
In [4]: #Unary Encoding
        def unaryEncoding(var):
            unary = ""
            for i in range(var-1):
                unary='0'+unary
            unary=unary+'1'
            return unary
```



```
In [5]: #Unary Decoding
        def unaryDecoding(var):
            counter=0
            while(var[0]=='0'):
                var=var[1:]
                counter=counter+1
            return counter+1
```



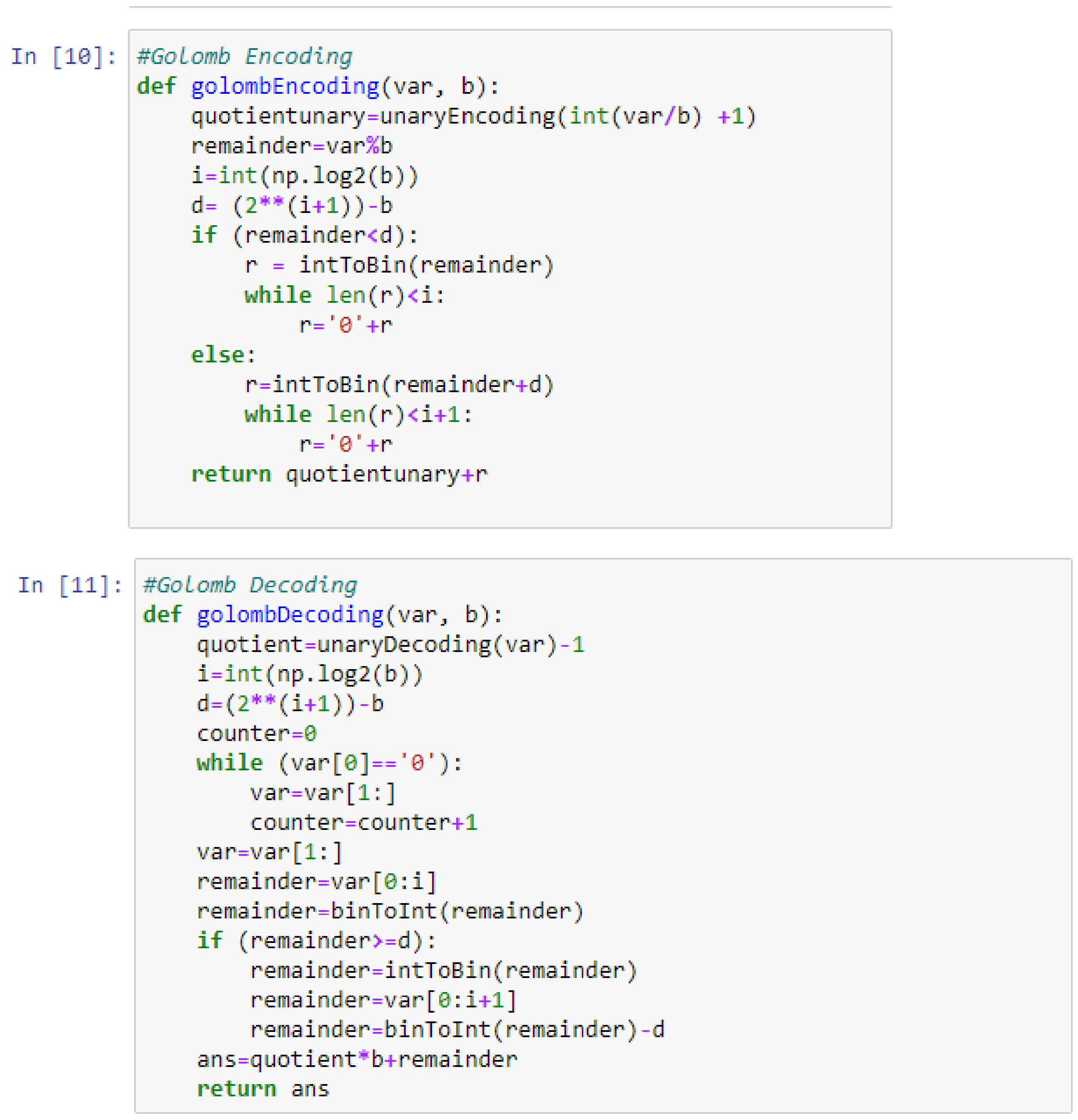
```
In [6]: #Elias Gamma Encoding
        def eliasGammaEncoding(var):
            var = intToBin(var)
            n=len(var)-1
            for i in range(n):
                var = '0'+var
            return var
```



In [7]:
```python
#Elias Gamma Decoding
def eliasGammaDecoding(var):
    counter=0
    while(var[0]=='0'):
        var=var[1:]
        counter=counter+1
    var=var[0:counter+1:1]
    return binToInt(var)
```

In [8]:
```python
#Elias Delta Encoding
def eliasDeltaEncoding(var):
    selector = eliasGammaEncoding(1+int(np.log2(var)))
    var = intToBin(var)
    offset=""
    for i in range(1, len(var)):
        offset=offset+var[i]
    return (selector+offset)
```

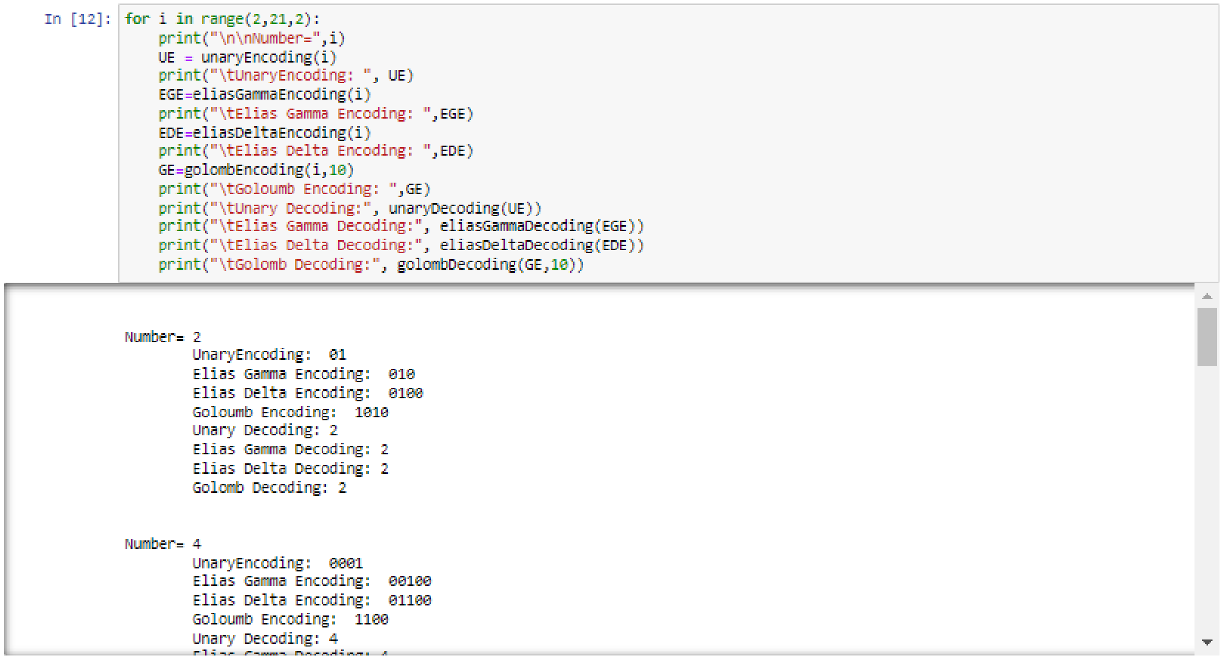
In [9]:
```python
#Elias Delta Decoding
def eliasDeltaDecoding(var):
    Nbits=eliasGammaDecoding(var)-1
    ans=""
    for i in range(Nbits):
        ans=var[-(i+1)]+ans
    return binToInt('1'+ans)
```



```
In [10]: #Golomb Encoding
         def golombEncoding(var, b):
             quotientunary=unaryEncoding(int(var/b) +1)
             remainder=var%b
             i=int(np.log2(b))
             d= (2**(i+1))-b
             if (remainder<d):
                 r = intToBin(remainder)
                 while len(r)<i:
                     r='0'+r
             else:
                 r=intToBin(remainder+d)
                 while len(r)<i+1:
                     r='0'+r
             return quotientunary+r
```

```
In [11]: #Golomb Decoding
         def golombDecoding(var, b):
             quotient=unaryDecoding(var)-1
             i=int(np.log2(b))
             d=(2**(i+1))-b
             counter=0
             while (var[0]=='0'):
                 var=var[1:]
                 counter=counter+1
             var=var[1:]
             remainder=var[0:i]
             remainder=binToInt(remainder)
             if (remainder>=d):
                 remainder=intToBin(remainder)
                 remainder=var[0:i+1]
                 remainder=binToInt(remainder)-d
             ans=quotient*b+remainder
             return ans
```

Here, we had defined all the ten functions that are described in procedure.

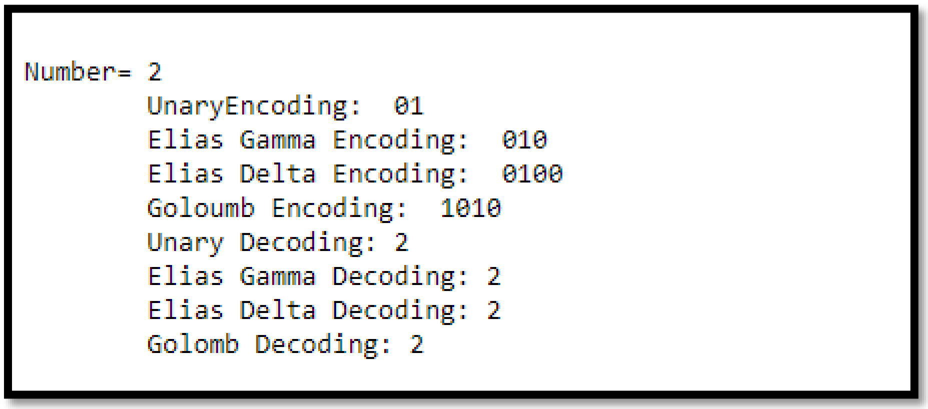


```
In [12]: for i in range(2,21,2):
             print("\n\nNumber=",i)
             UE = unaryEncoding(i)
             print("\tUnaryEncoding: ", UE)
             EGE=eliasGammaEncoding(i)
             print("\tElias Gamma Encoding: ",EGE)
             EDE=eliasDeltaEncoding(i)
             print("\tElias Delta Encoding: ",EDE)
             GE=golombEncoding(i,10)
             print("\tGoloumb Encoding: ",GE)
             print("\tUnary Decoding:", unaryDecoding(UE))
             print("\tElias Gamma Decoding:", eliasGammaDecoding(EGE))
             print("\tElias Delta Decoding:", eliasDeltaDecoding(EDE))
             print("\tGolomb Decoding:", golombDecoding(GE,10))
```

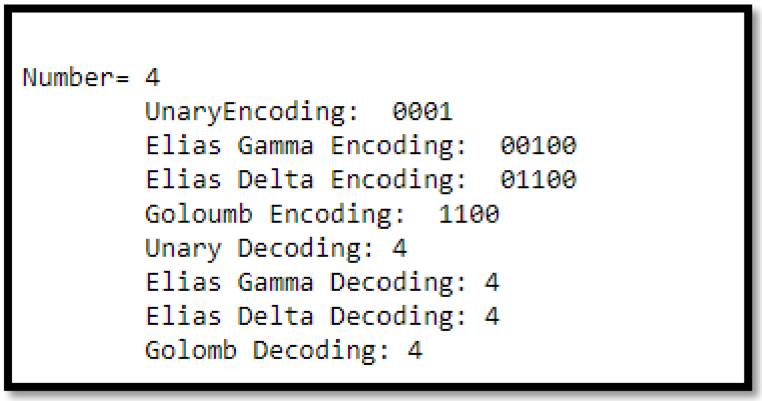
```


Number= 2
        UnaryEncoding:  01
        Elias Gamma Encoding:  010
        Elias Delta Encoding:  0100
        Goloumb Encoding:  1010
        Unary Decoding: 2
        Elias Gamma Decoding: 2
        Elias Delta Decoding: 2
        Golomb Decoding: 2


Number= 4
        UnaryEncoding:  0001
        Elias Gamma Encoding:  00100
        Elias Delta Encoding:  01100
        Goloumb Encoding:  1100
        Unary Decoding: 4
```

Here we are running a loop in order to iterate the even numbers in range 1-20 and then use the above functions to get our results.

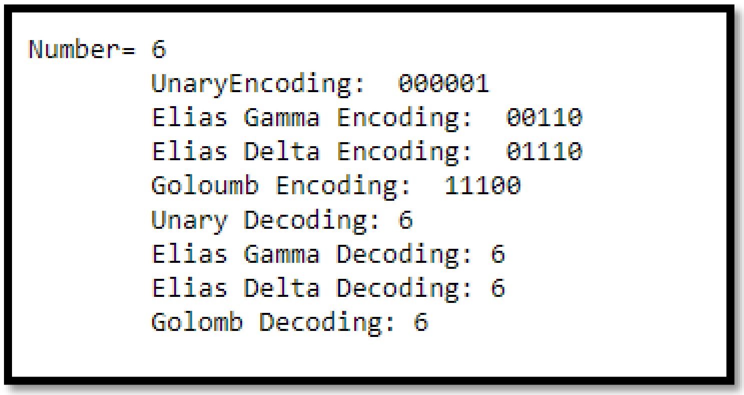
Results and Output



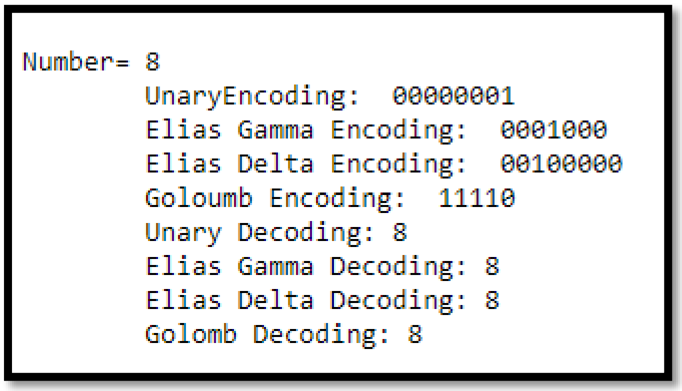
```
Number= 2
        UnaryEncoding:  01
        Elias Gamma Encoding:  010
        Elias Delta Encoding:  0100
        Goloumb Encoding:  1010
        Unary Decoding: 2
        Elias Gamma Decoding: 2
        Elias Delta Decoding: 2
        Golomb Decoding: 2
```



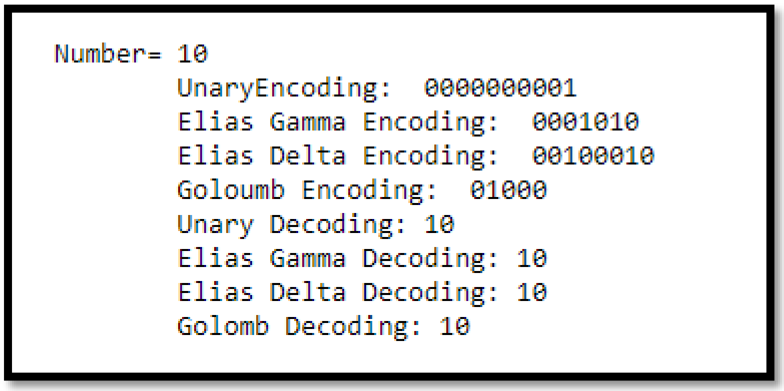
```
Number= 4
        UnaryEncoding:  0001
        Elias Gamma Encoding:  00100
        Elias Delta Encoding:  01100
        Goloumb Encoding:  1100
        Unary Decoding: 4
        Elias Gamma Decoding: 4
        Elias Delta Decoding: 4
        Golomb Decoding: 4
```



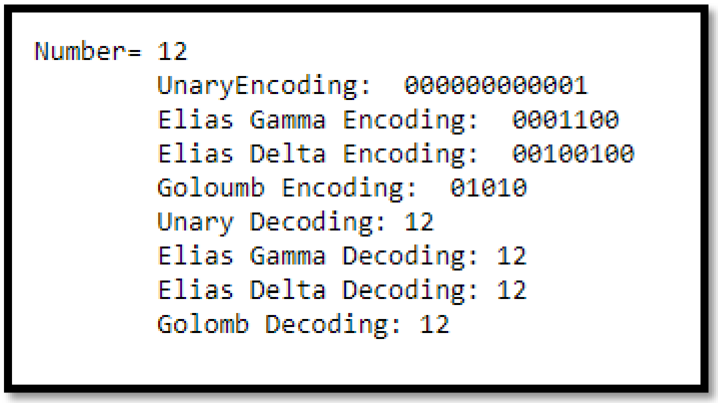
```
Number= 6
        UnaryEncoding:  000001
        Elias Gamma Encoding:  00110
        Elias Delta Encoding:  01110
        Goloumb Encoding:  11100
        Unary Decoding: 6
        Elias Gamma Decoding: 6
        Elias Delta Decoding: 6
        Golomb Decoding: 6
```



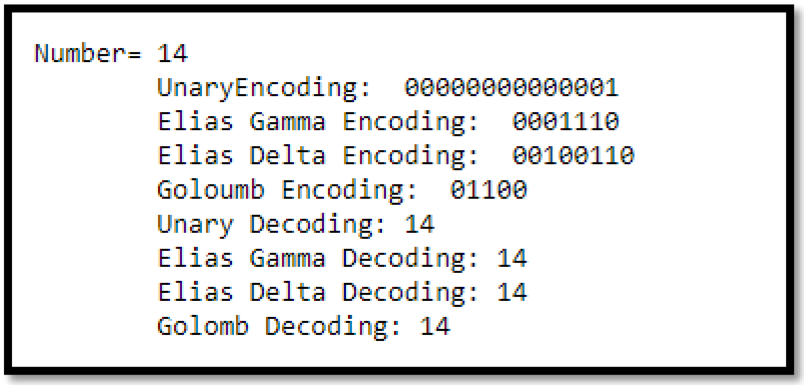
```
Number= 8
        UnaryEncoding:  00000001
        Elias Gamma Encoding:  0001000
        Elias Delta Encoding:  00100000
        Goloumb Encoding:  11110
        Unary Decoding: 8
        Elias Gamma Decoding: 8
        Elias Delta Decoding: 8
        Golomb Decoding: 8
```



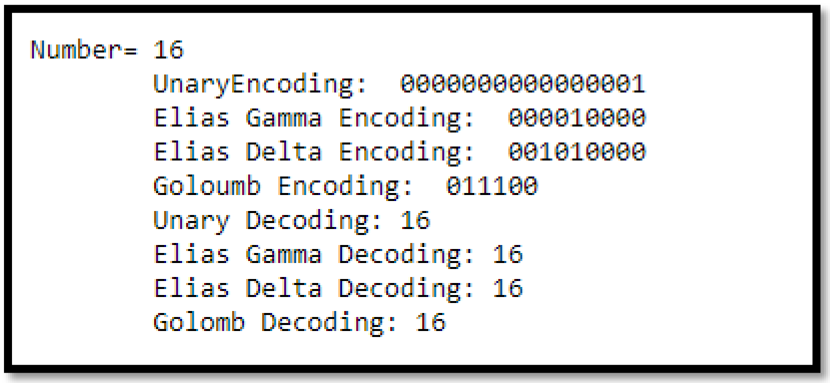
```
Number= 10
        UnaryEncoding:  0000000001
        Elias Gamma Encoding:  0001010
        Elias Delta Encoding:  00100010
        Goloumb Encoding:  01000
        Unary Decoding: 10
        Elias Gamma Decoding: 10
        Elias Delta Decoding: 10
        Golomb Decoding: 10
```



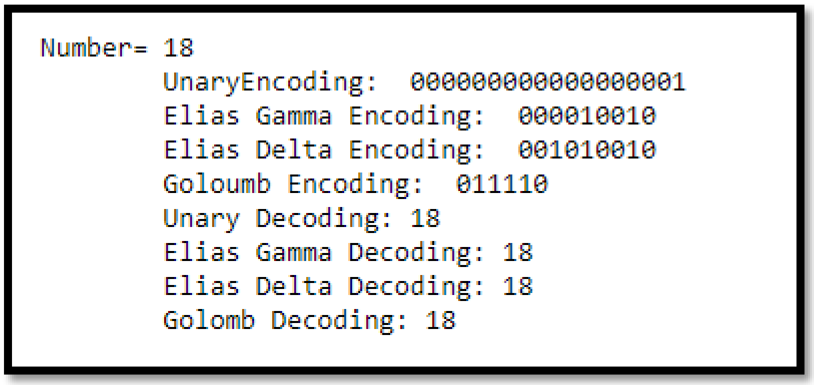
```
Number= 12
        UnaryEncoding:  000000000001
        Elias Gamma Encoding:  0001100
        Elias Delta Encoding:  00100100
        Goloumb Encoding:  01010
        Unary Decoding: 12
        Elias Gamma Decoding: 12
        Elias Delta Decoding: 12
        Golomb Decoding: 12
```



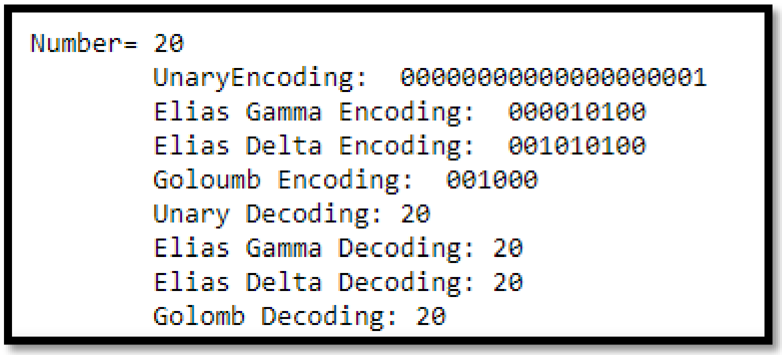
```
Number= 14
        UnaryEncoding:  00000000000001
        Elias Gamma Encoding:  0001110
        Elias Delta Encoding:  00100110
        Goloumb Encoding:  01100
        Unary Decoding: 14
        Elias Gamma Decoding: 14
        Elias Delta Decoding: 14
        Golomb Decoding: 14
```



```
Number= 16
        UnaryEncoding:  0000000000000001
        Elias Gamma Encoding:  000010000
        Elias Delta Encoding:  001010000
        Goloumb Encoding:  011100
        Unary Decoding: 16
        Elias Gamma Decoding: 16
        Elias Delta Decoding: 16
        Golomb Decoding: 16
```



```
Number= 18
        UnaryEncoding:  000000000000000001
        Elias Gamma Encoding:  000010010
        Elias Delta Encoding:  001010010
        Goloumb Encoding:  011110
        Unary Decoding: 18
        Elias Gamma Decoding: 18
        Elias Delta Decoding: 18
        Golomb Decoding: 18
```



```
Number= 20
        UnaryEncoding:  00000000000000000001
        Elias Gamma Encoding:  000010100
        Elias Delta Encoding:  001010100
        Goloumb Encoding:  001000
        Unary Decoding: 20
        Elias Gamma Decoding: 20
        Elias Delta Decoding: 20
        Golomb Decoding: 20
```