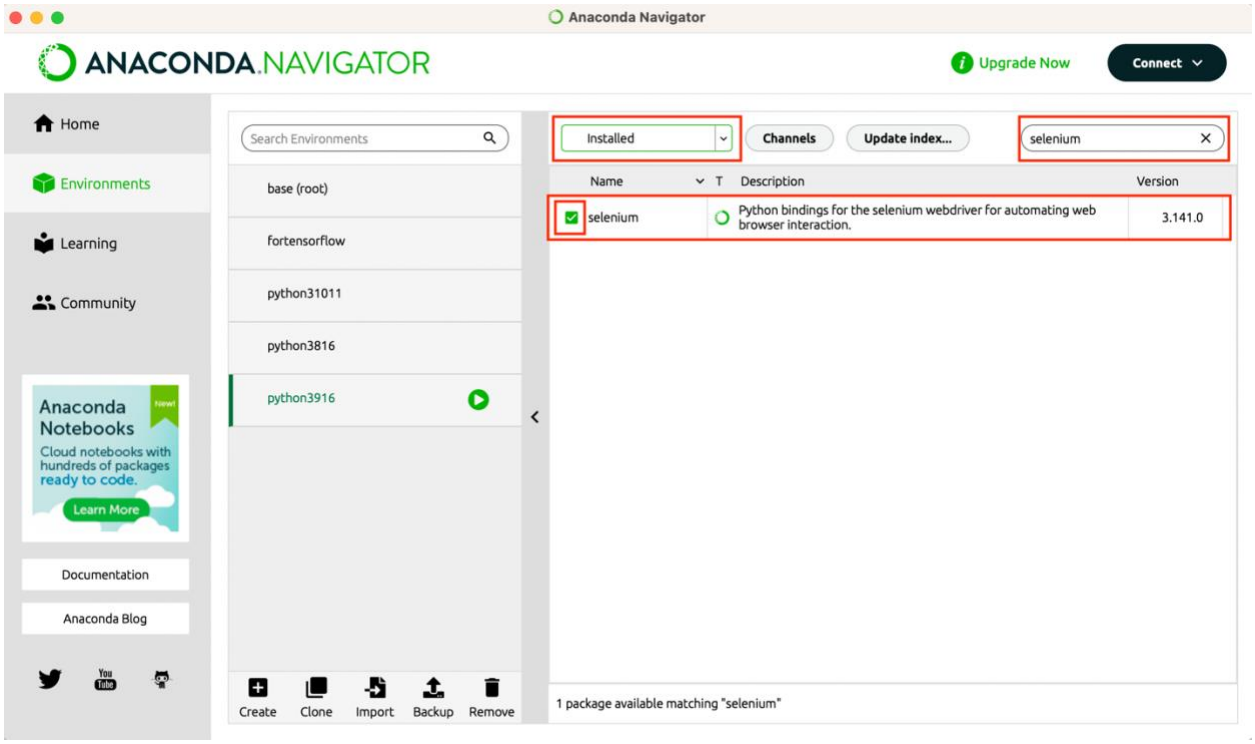
Handout_Data Collection I

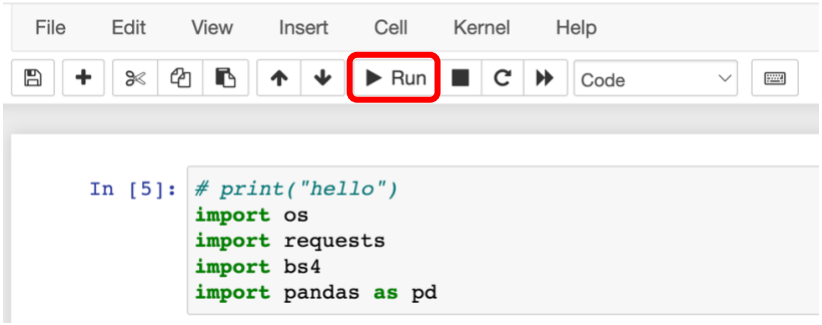
We are going to learn to use Python to collect social data from websites. For this purpose, you need to install the program first.

1. *Install Anaconda, Jupyter Notebook, and Python packages*
   1) Please follow the instruction to install the applications and necessary packages for data collection via the link below
      https://chunshengj.github.io/579-class/data_analysis/Install_Anaconda_Jupyter_Package.html

   2) Ensure that you installed all the required packages: "bs4", "requests", "selenium", and "pandas"
      a. If the packages are successfully installed, the box next to the package name is checked.



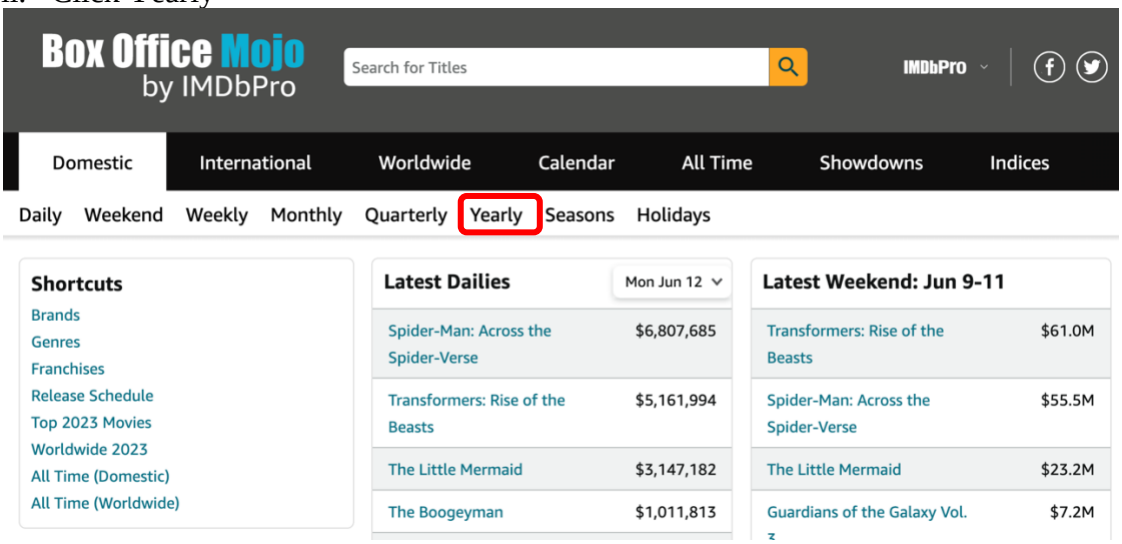
2. *Simple web scraping with Beautiful Soup and Requests*
   1. Import the packages we need for web scraping
      a. Import four packages: os, requests, bs4, and pandas
      b. Click on "Run" to run the code
         i. Deactivate the first print command by putting # in front of the command. This deactivation is called commenting.



2. Set the working directory
   a. The directory notation is different according to your operating system. Please check the commands below and deactivate the unused command with "#".
   b. The working directory shown below is the directory on my computer. You should use your directory which is different from this example.

```
os.chdir("/Users/sheng/Jupyter/AESHM_579")  # for Mac
# os.chdir("C:\\Users\\sheng\\Jupyter\\AESHM_579")  # for Windows
```

3. Select the website for scraping
   a. Let's scrape the movie information we have been using for this course.
   b. This time, let's collect movies released in the most recent year.
      i. Go to Box Office Mojo: https://www.boxofficemojo.com
      ii. Click Yearly



iii. Click the most recent year completed in the Year column

Overview | Calendar grosses

Data as of Jun 13, 19:30 PDT

Year	Total Gross	%± LY	Releases	Average	#1 Release
2023	$3,885,988,821	-	284	$13,683,059	The Super Mario Bros. Movie
2022	$7,369,357,270	+64.4%	496	$14,857,575	Top Gun: Maverick
2021	$4,482,808,453	+112.1%	440	$10,188,201	Spider-Man: No Way Home
2020	$2,113,846,800	-81.4%	456	$4,635,628	Bad Boys for Life
2019	$11,363,360,889	-4.4%	910	$12,487,209	Avengers: Endgame
2018	$11,892,160,011	+7.4%	993	$11,975,991	Black Panther
2017	$11,075,387,520	-2.6%	854	$12,968,837	Star Wars: Episode VIII - The Last Jedi

iv. We can see all the movies released in the most recent year completed.

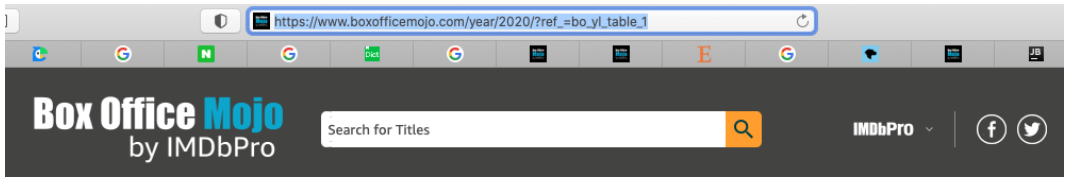
Domestic Box Office For 2023

2023 | Calendar grosses

Rank	Release	Gross	Theaters	Total Gross	Release Date	Distributor
1	The Super Mario Bros. Movie	$570,554,370	4,371	$570,554,370	Apr 5	Universal Pictures
2	Guardians of the Galaxy Vol. 3	$336,521,512	4,450	$336,521,512	May 5	Walt Disney Studios Motion Pictures
3	Avatar: The Way of Water	$283,067,859	4,340	$684,075,767	Dec 16	20th Century Studios
4	Spider-Man: Across the Spider-Verse	$232,339,759	4,332	$232,339,759	Jun 2	Columbia Pictures
5	The Little Mermaid	$232,317,534	4,320	$232,317,534	May 26	Walt Disney Studios Motion Pictures
6	Ant-Man and the Wasp: Quantumania	$214,503,921	4,345	$214,503,921	Feb 17	Walt Disney Studios Motion Pictures
7	John Wick: Chapter 4	$187,091,533	3,855	$187,091,533	Mar 24	Lionsgate
8	Creed III	$156,248,615	4,007	$156,248,615	Mar 3	United Artists Releasing
9	Fast X	$138,760,790	4,088	$138,760,790	May 19	Universal Pictures

v. Copy the URL of the webpage

> URL stands for Uniform Resource



4. Collecting the content in the webpage
   a. Assign the URL to a variable name "df"
   b. Request the content of the webpage and assign them to a variable "page"
   c. Parse the HTML document in the content with a Python package called BeautifulSoup (we are using bs4) which is the most common web scraping package in Python.
      i. HTML is HyperText Markup Language, which is "a language that webpages are created in. HTML isn't a programming language, like

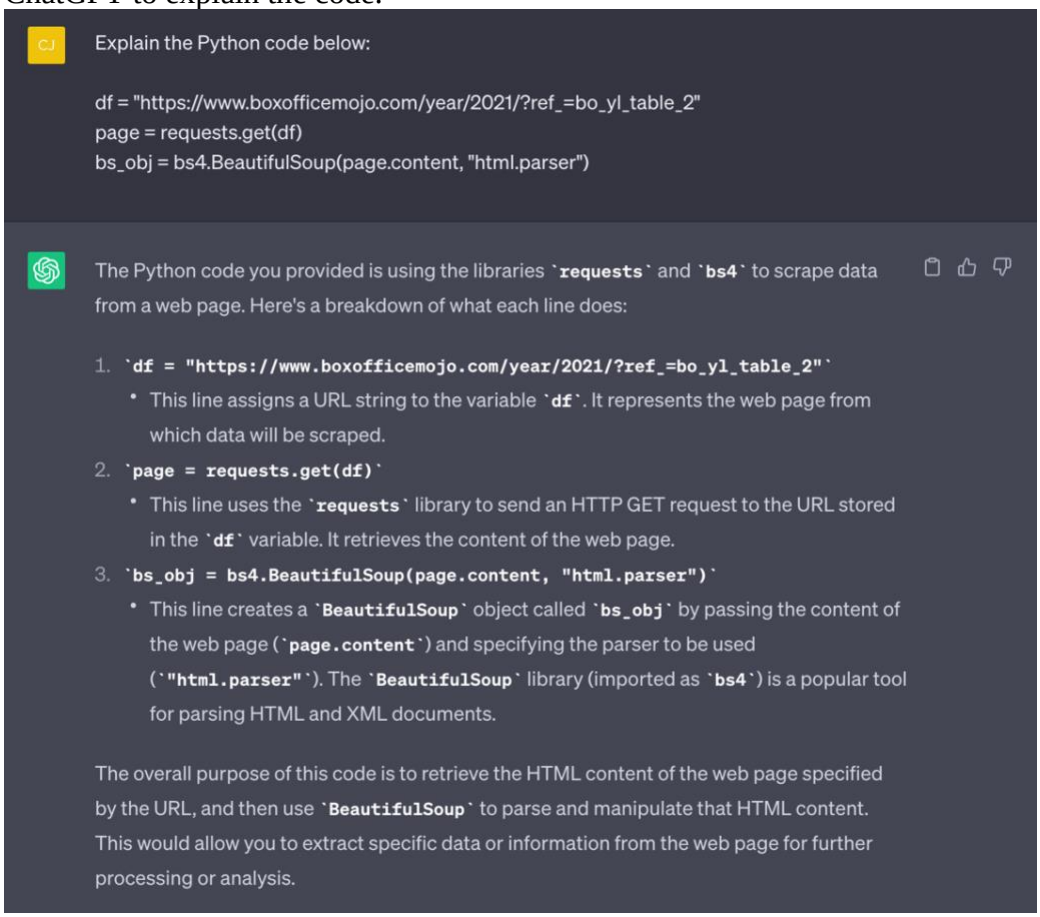
Python — instead, it's a markup language that tells a browser how to layout content" (Data Request, 2020).

ii. Assign the HTML document to a variable "bs_obj"

```
df='https://www.boxofficemojo.com/year/2023/?ref_=bo_yl_table_1'
page = requests.get(df)

bs_obj = bs4.BeautifulSoup(page.content, "html.parser")
```

iii. If you want to explore the meaning of each command line, you can ask ChatGPT to explain the code.



CJ

Explain the Python code below:

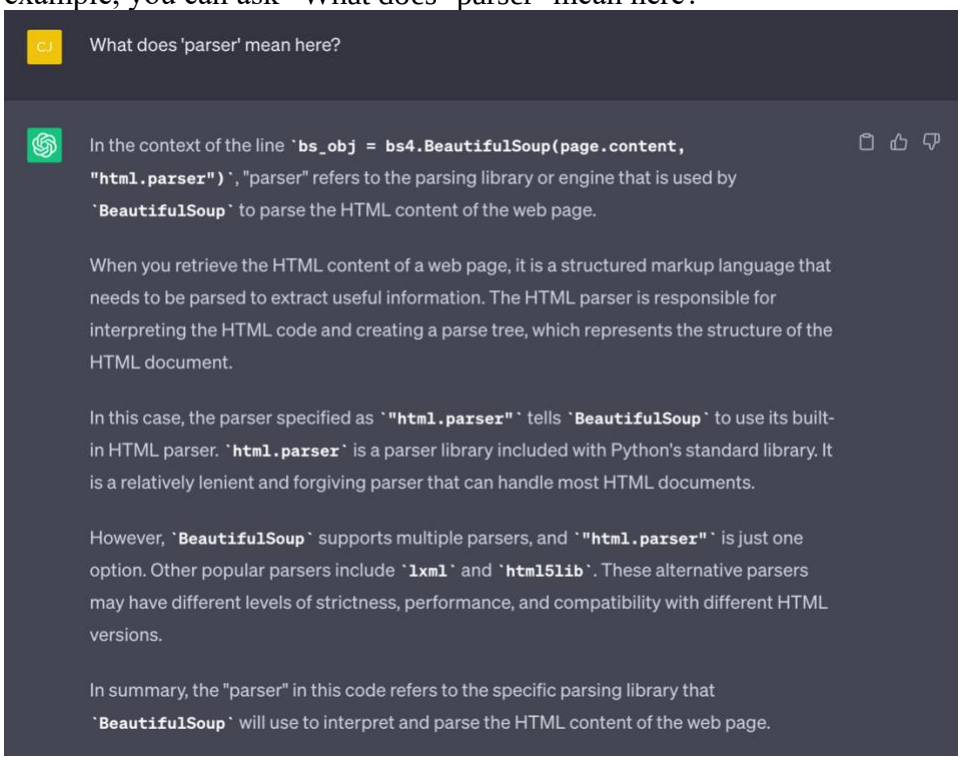
df = "https://www.boxofficemojo.com/year/2021/?ref_=bo_yl_table_2"
page = requests.get(df)
bs_obj = bs4.BeautifulSoup(page.content, "html.parser")

The Python code you provided is using the libraries `requests` and `bs4` to scrape data from a web page. Here's a breakdown of what each line does:

1. `df = "https://www.boxofficemojo.com/year/2021/?ref_=bo_yl_table_2"`
   - This line assigns a URL string to the variable `df`. It represents the web page from which data will be scraped.
2. `page = requests.get(df)`
   - This line uses the `requests` library to send an HTTP GET request to the URL stored in the `df` variable. It retrieves the content of the web page.
3. `bs_obj = bs4.BeautifulSoup(page.content, "html.parser")`
   - This line creates a `BeautifulSoup` object called `bs_obj` by passing the content of the web page (`page.content`) and specifying the parser to be used (`"html.parser"`). The `BeautifulSoup` library (imported as `bs4`) is a popular tool for parsing HTML and XML documents.

The overall purpose of this code is to retrieve the HTML content of the web page specified by the URL, and then use `BeautifulSoup` to parse and manipulate that HTML content. This would allow you to extract specific data or information from the web page for further processing or analysis.

iv. You can further interact with ChatGPT to get detailed answers. For example, you can ask “What does ‘parser’ mean here?”



What does 'parser' mean here?

In the context of the line `bs_obj = bs4.BeautifulSoup(page.content, "html.parser")`, "parser" refers to the parsing library or engine that is used by `BeautifulSoup` to parse the HTML content of the web page.

When you retrieve the HTML content of a web page, it is a structured markup language that needs to be parsed to extract useful information. The HTML parser is responsible for interpreting the HTML code and creating a parse tree, which represents the structure of the HTML document.

In this case, the parser specified as `"html.parser"` tells `BeautifulSoup` to use its built-in HTML parser. `html.parser` is a parser library included with Python's standard library. It is a relatively lenient and forgiving parser that can handle most HTML documents.

However, `BeautifulSoup` supports multiple parsers, and `"html.parser"` is just one option. Other popular parsers include `lxml` and `html5lib`. These alternative parsers may have different levels of strictness, performance, and compatibility with different HTML versions.

In summary, the "parser" in this code refers to the specific parsing library that `BeautifulSoup` will use to interpret and parse the HTML content of the web page.

5. Structure of HTML document
   To collect specific contents we are interested in from the parsed HTML document, we need to understand the structure of the HTML document.

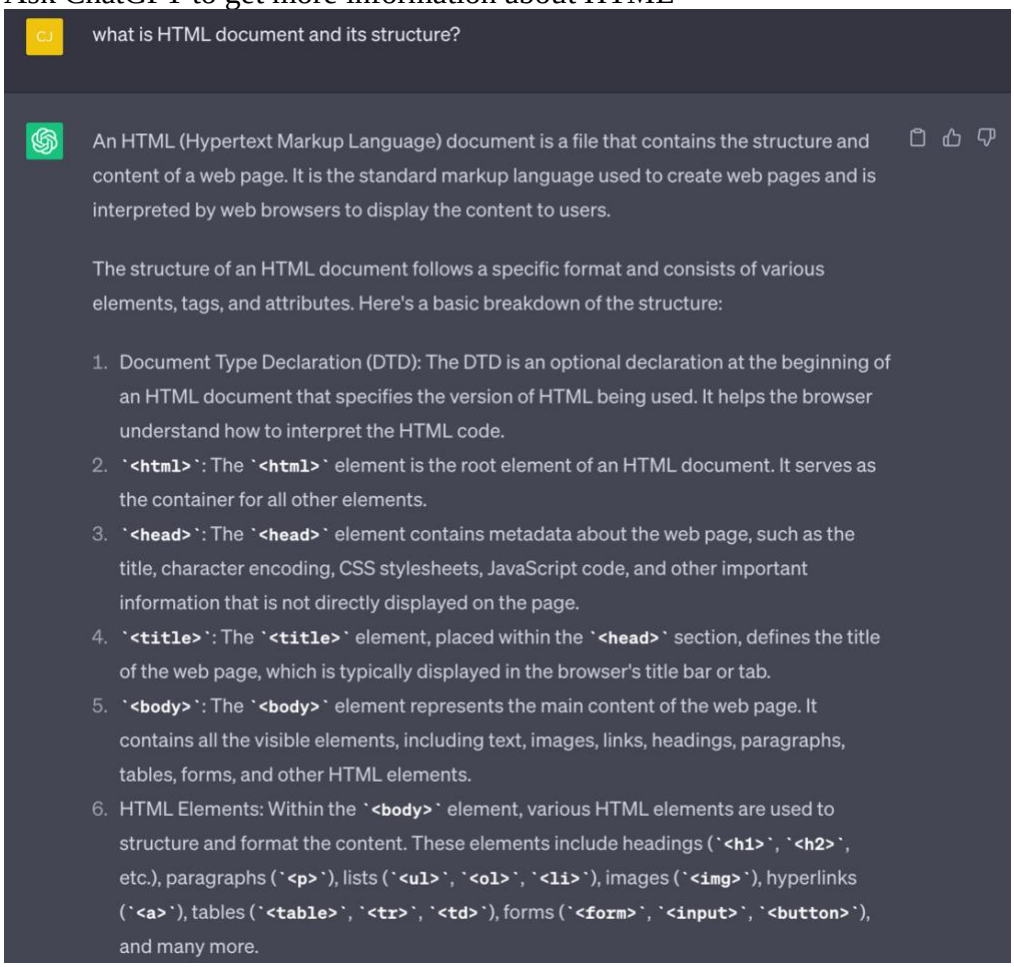
   a. Basic structure

```
<!DOCTYPE html>
        <html>
                <head>
                        head contents (mainly title)
                </head>

                <body>
                        body contents (webpage contents)
                </body>
        </html>
```

   i. HTML document consists of 1) head and 2) body.
   ii. The head element mainly contains the title of the document while the body element contains the main contents of a webpage.

iii. Ask ChatGPT to get more information about HTML



what is HTML document and its structure?

An HTML (Hypertext Markup Language) document is a file that contains the structure and content of a web page. It is the standard markup language used to create web pages and is interpreted by web browsers to display the content to users.

The structure of an HTML document follows a specific format and consists of various elements, tags, and attributes. Here's a basic breakdown of the structure:

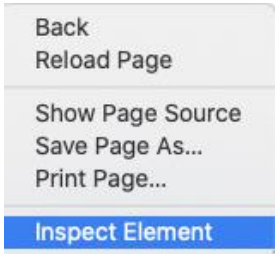
1. Document Type Declaration (DTD): The DTD is an optional declaration at the beginning of an HTML document that specifies the version of HTML being used. It helps the browser understand how to interpret the HTML code.
2. `<html>`: The `<html>` element is the root element of an HTML document. It serves as the container for all other elements.
3. `<head>`: The `<head>` element contains metadata about the web page, such as the title, character encoding, CSS stylesheets, JavaScript code, and other important information that is not directly displayed on the page.
4. `<title>`: The `<title>` element, placed within the `<head>` section, defines the title of the web page, which is typically displayed in the browser's title bar or tab.
5. `<body>`: The `<body>` element represents the main content of the web page. It contains all the visible elements, including text, images, links, headings, paragraphs, tables, forms, and other HTML elements.
6. HTML Elements: Within the `<body>` element, various HTML elements are used to structure and format the content. These elements include headings (`<h1>`, `<h2>`, etc.), paragraphs (`<p>`), lists (`<ul>`, `<ol>`, `<li>`), images (`<img>`), hyperlinks (`<a>`), tables (`<table>`, `<tr>`, `<td>`), forms (`<form>`, `<input>`, `<button>`), and many more.

b. Example of the HTML document

i. Check the element of the webpage of Box Office Mojo

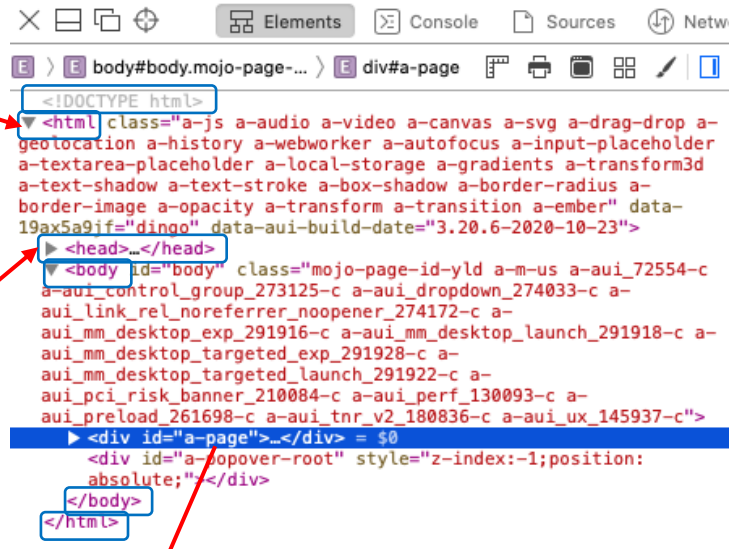
> Go to the webpage: https://www.boxofficemojo.com/year/2023/?ref_=bo_yl_table_1

> Right mouse click => Inspect Element

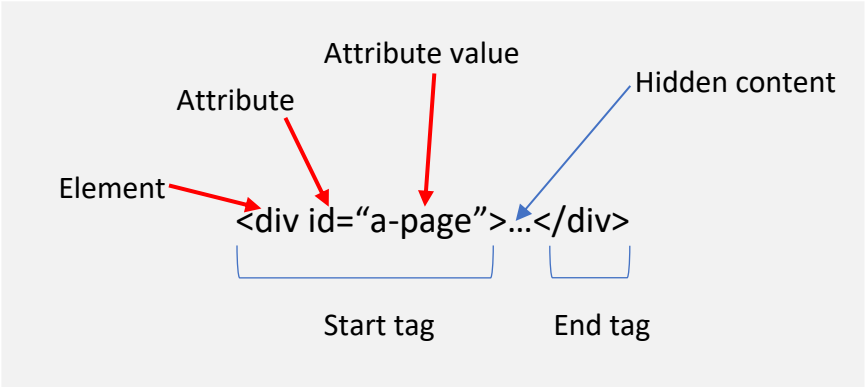


This downward arrow indicates all the content in this tag is spread out.

On the other hand, the right arrow indicates this tag is folded. If you can click this arrow, you can unfold the content in the tag.

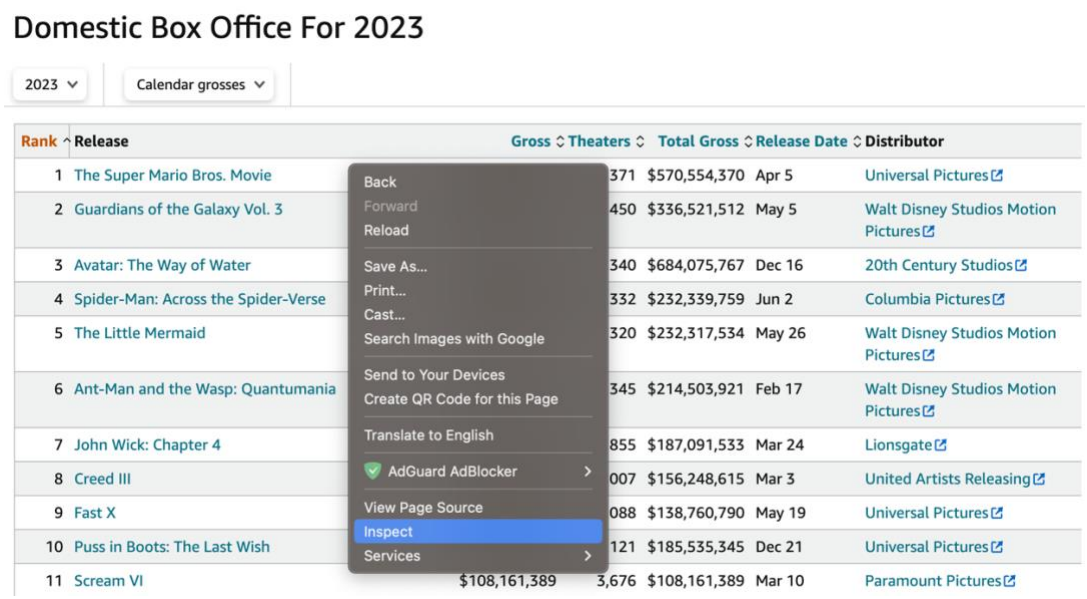


c. Structure of elements

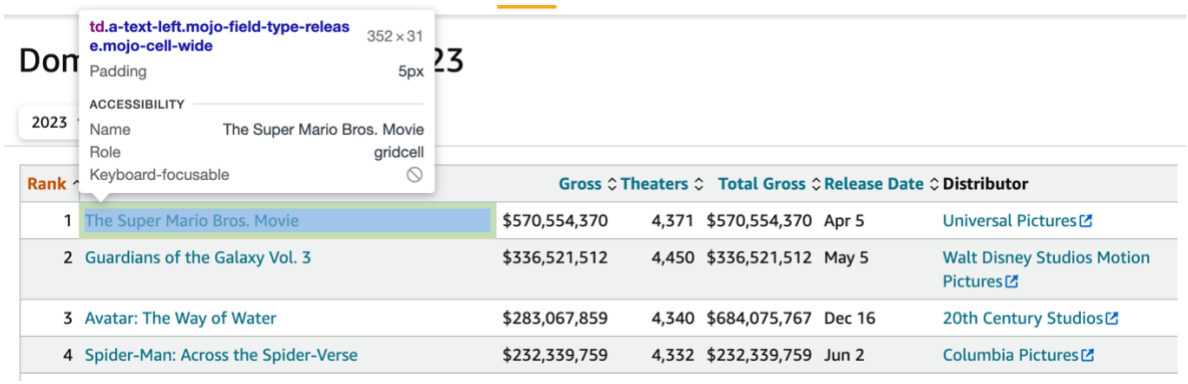


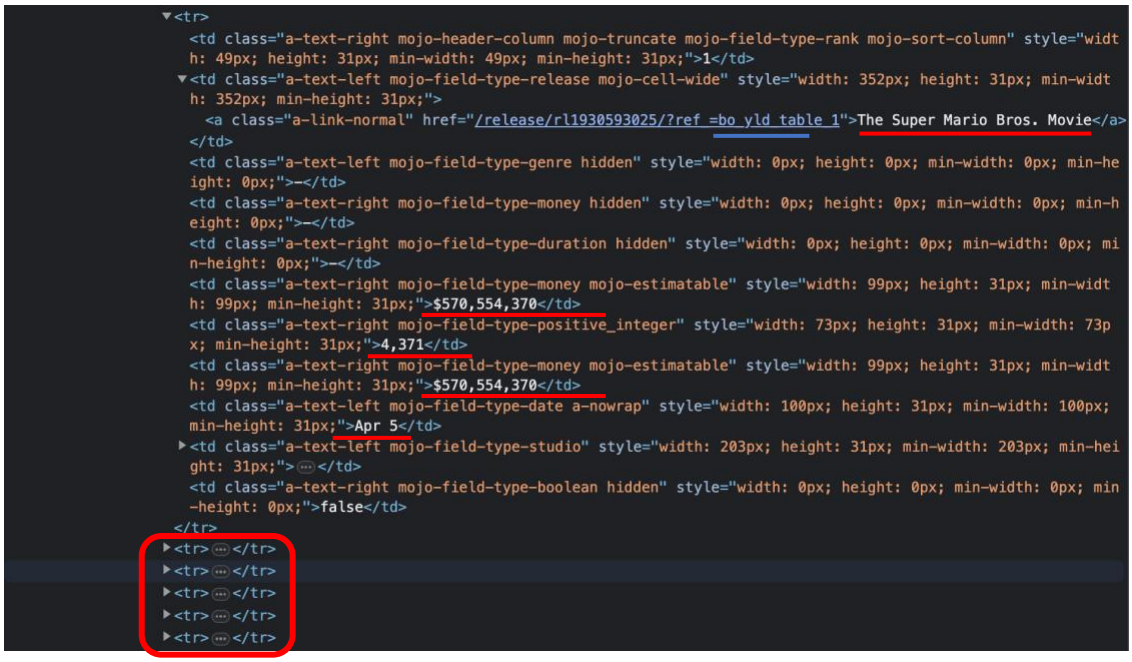
6. Find the code patterns for movies
   To collect the information of movies from the webpage, we need to find repeated elements for movies.

   a. Go back to the movie webpage
   b. Inspect the element of movies
      i. Click the table for movies
      ii. Right mouse click => Inspect element



c. Movies are contained in the "tr" element





i. We also find that domestic gross, total gross, number of theaters, release date, and studio are contained as texts in black.

7. Collect all the content in the “tr” element
   a. Put the content in a variable “movie”
   b. Run the commands

```
df='https://www.boxofficemojo.com/year/2023/?ref_=bo_yl_table_1'
page = requests.get(df)

bs_obj = bs4.BeautifulSoup(page.content, "html.parser")
movie = bs_obj.findAll("tr")
```

   c. Ask ChatGPT to explain the code

CJ

Explain the last code line:

bs_obj = bs4.BeautifulSoup(page.content, "html.parser")
movie = bs_obj.findAll("tr")

The last code line `movie = bs_obj.findAll("tr")` uses the BeautifulSoup object `bs_obj` to find and extract all the HTML elements that are `<tr>` tags from the parsed HTML content.

Here's a breakdown of what each part of the line does:

1. `bs_obj`: This is the BeautifulSoup object created earlier, representing the parsed HTML content of the web page.
2. `findAll("tr")`: This method of the BeautifulSoup object finds all instances of HTML elements that match the specified tag name, in this case, `"tr"`, which represents the table row elements in HTML.

By calling `findAll("tr")`, the code is searching for all the table row elements (`<tr>`) in the HTML content. The result is stored in the variable `movie`, which would be a list-like object containing all the `<tr>` elements found in the HTML.

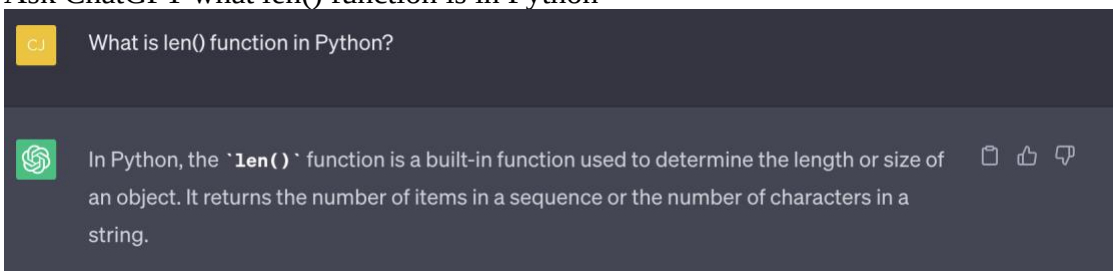
After this line of code, you can further process the `movie` object to extract specific data or information from the table rows, such as movie titles, box office numbers, or other relevant details.

   d. Need to check if there is any other content except “movie” which is contained using the “tr” element
      i. Check the length of the movie variable

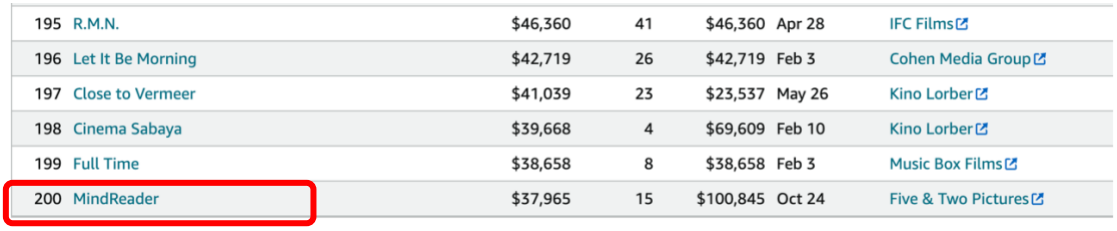
```
print(len(movie))
```

```
201
```

i. Ask ChatGPT what len() function is in Python



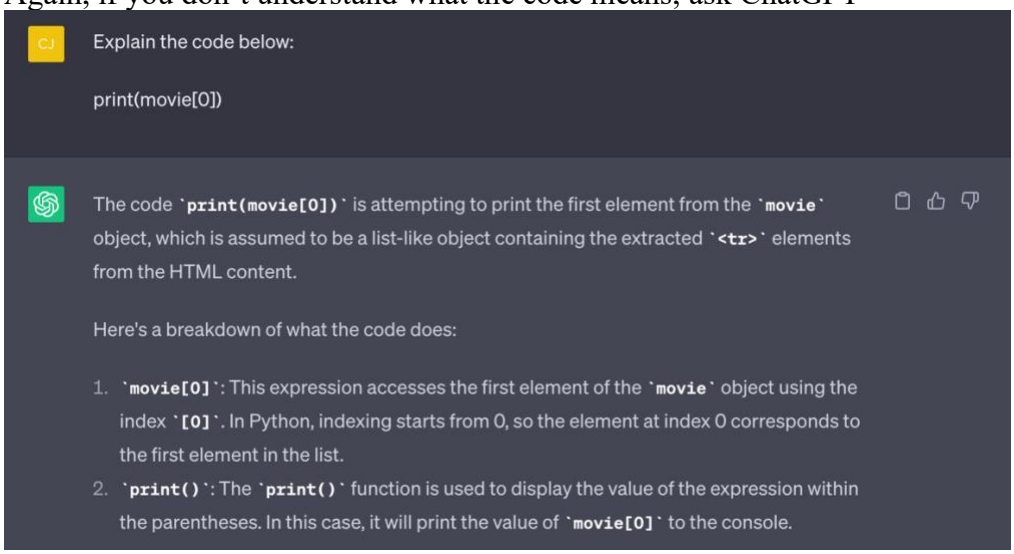
ii. The length of the movie variable is 201, but the total number of movies in the table is 200 (This is the maximum number of movies per year provided by this website). So, we need check which "tr" element has non-movie information.



195	R.M.N.	$46,360	41	$46,360	Apr 28	IFC Films
196	Let It Be Morning	$42,719	26	$42,719	Feb 3	Cohen Media Group
197	Close to Vermeer	$41,039	23	$23,537	May 26	Kino Lorber
198	Cinema Sabaya	$39,668	4	$69,609	Feb 10	Kino Lorber
199	Full Time	$38,658	8	$38,658	Feb 3	Music Box Films
200	MindReader	$37,965	15	$100,845	Oct 24	Five & Two Pictures

iii. For this purpose, we should check the first and last "tr" element. When you print out those "tr" element, you should know that every number in Python commands starts from 0, rather than 1. So, to navigate 201 contents in the movie variable, we need to investigate from movie[0] to movie[200]. This time, we pull up the contents in movie[0] and movie[200].

iv. Again, if you don't understand what the code means, ask ChatGPT



v. This is the content in the first “tr” element. It starts with <tr> and ends with </tr>. However, there is no information of a specific movie.

```
print(movie[0])
```

```
<tr><th class="a-text-right mojo-field-type-rank mojo-sort-column mojo-sortable-column a-nowrap"><a class="a-link-nor
mal a-nowrap" href="?sort=rank&amp;ref_=bo_yld__resort#table" title="Rank"><span class="a-color-state">Rank</span><sp
an class="a-letter-space"></span><span class="icon aok-relative"><i class="a-icon a-icon-collapse" role="presentatio
n"></i></span></a></th><th class="a-text-left mojo-field-type-release mojo-cell-wide mojo-sortable-column a-nowrap"><
span title="Release">Release</span>
</th><th class="a-text-left mojo-field-type-genre hidden mojo-sortable-column hidden a-nowrap"><span title="Genre">Ge
nre</span>
</th><th class="a-text-right mojo-field-type-money hidden mojo-sortable-column hidden a-nowrap"><span title="Budget">
Budget</span>
</th><th class="a-text-right mojo-field-type-duration hidden mojo-sortable-column hidden a-nowrap"><span title="Runni
ng Time">Running Time</span>
</th><th class="a-text-right mojo-field-type-money mojo-sortable-column mojo-estimatable a-nowrap"><a class="a-link-n
ormal a-nowrap" href="?sort=gross&amp;ref_=bo_yld__resort#table" title="Gross">Gross<span class="a-letter-space"></sp
an><span class="icon aok-relative"><i class="a-icon a-icon-expand table-sort-desc-placeholder" role="presentation"></
i><i class="a-icon a-icon-collapse table-sort-asc-placeholder" role="presentation"></i></span></a></th><th class="a-t
ext-right mojo-field-type-positive_integer mojo-sortable-column a-nowrap"><a class="a-link-normal a-nowrap" href="?so
rt=maxNumTheaters&amp;ref_=bo_yld__resort#table" title="Maximum Number of Theaters">Theaters<span class="a-letter-spa
ce"></span><span class="icon aok-relative"><i class="a-icon a-icon-expand table-sort-desc-placeholder" role="presenta
tion"></i><i class="a-icon a-icon-collapse table-sort-asc-placeholder" role="presentation"></i></span></a></th><th cl
ass="a-text-right mojo-field-type-money mojo-sortable-column mojo-estimatable a-nowrap"><a class="a-link-normal a-now
rap" href="?sort=grossToDate&amp;ref_=bo_yld__resort#table" title="Total Gross">Total Gross<span class="a-letter-spac
e"></span><span class="icon aok-relative"><i class="a-icon a-icon-expand table-sort-desc-placeholder" role="presentat
ion"></i><i class="a-icon a-icon-collapse table-sort-asc-placeholder" role="presentation"></i></span></a></th><th cla
ss="a-text-left mojo-field-type-date mojo-sortable-column a-nowrap"><a class="a-link-normal a-nowrap" href="?sort=rel
easeDate&amp;ref_=bo_yld__resort#table" title="Release Date">Release Date<span class="a-letter-space"></span><span cl
ass="icon aok-relative"><i class="a-icon a-icon-expand table-sort-desc-placeholder" role="presentation"></i><i class
="a-icon a-icon-collapse table-sort-asc-placeholder" role="presentation"></i></span></a></th><th class="a-text-left m
ojo-field-type-studio mojo-sortable-column a-nowrap"><span title="Distributor">Distributor</span>
</th><th class="a-text-right mojo-field-type-boolean hidden mojo-sortable-column hidden a-nowrap"><span title="Estima
ted">Estimated</span>
</th></tr>
```

vi. On the other hand, the content in the last “tr” element includes the information of the last movie in the table titled as “MindReader.” Also, we can find the information on gross, theater, release date, and studio.

```
print(movie[200])
```

```
<tr><td class="a-text-right mojo-header-column mojo-truncate mojo-field-type-rank mojo-sort-column">200</td><td class
="a-text-left mojo-field-type-release mojo-cell-wide"><a class="a-link-normal" href="/release/rl3571811073/?ref_=bo_y
ld_table_200">MindReader</a></td><td class="a-text-left mojo-field-type-genre hidden">-</td><td class="a-text-right m
ojo-field-type-money hidden">-</td><td class="a-text-right mojo-field-type-duration hidden">-</td><td class="a-text-r
ight mojo-field-type-money mojo-estimatable">$37,965</td><td class="a-text-right mojo-field-type-positive_integer">15
</td><td class="a-text-right mojo-field-type-money mojo-estimatable">$100,845</td><td class="a-text-left mojo-field-t
ype-date a-nowrap">Oct 24</td><td class="a-text-left mojo-field-type-studio"><a class="a-link-normal" href="https://p
ro.imdb.com/company/co0073962/boxoffice/?view=releases&amp;ref_=mojo_yld_table_200&amp;rf=mojo_yld_table_200" rel="no
opener" target="_blank">Five &amp; Two Pictures<svg class="mojo-new-window-svg" viewbox="0 0 32 32" xmlns="http://ww
w.w3.org/2000/svg">
<path d="M24,15.5725113,3V23.5A3.50424,3.50424,0,0,1,23.5,27H8.5A3.50424,3.50424,0,0,1,5,23.5V8.5A3.50424,3.50424,0,
0,1,8.5,5h4.9275513,3H8.5a.50641.50641,0,0,0-.5.5v15a.50641.50641,0,0,0,.5.5h15a.50641.50641,0,0,0,.5-.5ZM19.81952,8.
56372,12.8844,17.75a.49989.49989,0,0,0,.04547.654791.66534.66528a.49983.49983,0,0,0,.65479.0455319.18628-6.93518,2.12
579,2.12585a.5.5,0,0,0,.84741-.2752611.48273-9.35108a.50006.50006,0,0,0-.57214-.57214L17.969,5.59058a.5.5,0,0,0-.2752
6.84741Z"></path>
</svg></a></td><td class="a-text-right mojo-field-type-boolean hidden">false</td></tr>
```

vii. Based on these results, we can see that the first “tr” element (movie[0]) has the non-movie content and the movie information is contained from movie[1] to movie[200]. To confirm this, let’s check movie[1], the second “tr” element.

viii. As expected, the first movie “The Super Mario Bros. Movie” is found here.

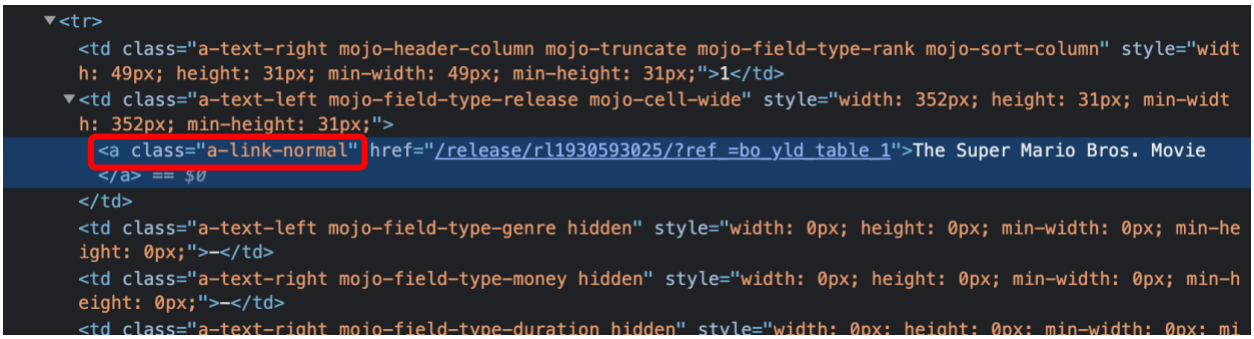
```
print(movie[1])
```

```
<tr><td class="a-text-right mojo-header-column mojo-truncate mojo-field-type-rank mojo-sort-column">1</td><td class
="a-text-left mojo-field-type-release mojo-cell-wide"><a class="a-link-normal" href="/release/rl1930593025/?ref_=bo_y
ld_table_1">The Super Mario Bros. Movie</a></td><td class="a-text-left mojo-field-type-genre hidden">-</td><td class
="a-text-right mojo-field-type-money hidden">-</td><td class="a-text-right mojo-field-type-duration hidden">-</td><td
class="a-text-right mojo-field-type-money mojo-estimatable">$570,554,370</td><td class="a-text-right mojo-field-type-
positive_integer">4,371</td><td class="a-text-right mojo-field-type-money mojo-estimatable">$570,554,370</td><td clas
s="a-text-left mojo-field-type-date a-nowrap">Apr 5</td><td class="a-text-left mojo-field-type-studio"><a class="a-li
nk-normal" href="https://pro.imdb.com/company/co0005073/boxoffice/?view=releases&amp;ref_=mojo_yld_table_1&amp;rf=moj
o_yld_table_1" rel="noopener" target="_blank">Universal Pictures<svg class="mojo-new-window-svg" viewbox="0 0 32 32"
xmlns="http://www.w3.org/2000/svg">
<path d="M24,15.5725113,3V23.5A3.50424,3.50424,0,0,1,23.5,27H8.5A3.50424,3.50424,0,0,1,5,23.5V8.5A3.50424,3.50424,0,
0,1,8.5,5h4.9275513,3H8.5a.50641.50641,0,0,0-.5.5v15a.50641.50641,0,0,0,.5.5h15a.50641.50641,0,0,0,.5-.5ZM19.81952,8.
56372,12.8844,17.75a.49989.49989,0,0,0,.04547.654791.66534.66528a.49983.49983,0,0,0,.65479.0455319.18628-6.93518,2.12
579,2.12585a.5.5,0,0,0,.84741-.2752611.48273-9.35108a.50006.50006,0,0,0-.57214-.57214L17.969,5.59058a.5.5,0,0,0-.2752
6.84741Z"></path>
</svg></a></td><td class="a-text-right mojo-field-type-boolean hidden">false</td></tr>
```

3. *Extract specific information from the parsed HTML document*

Thus far, we have collected all the content in HTML documents in a webpage. The content is now in your computer. Then, you may want to extract specific information of a movie provided in the webpage such as its title and gross.

1) Go to the webpage for 2023 movies and check the element of specific information of interest. Start with the movie title.



2) The title information is contained as a text in the “a” element with the “class” attribute and the attribute value of “a-link-normal”
   a. Use the “find” and “text” functions with the element information to extract the movie title

```python
title = movie[1].find("a", {"class": "a-link-normal"}).text
print(title)
```

```
The Super Mario Bros. Movie
```

b. Ask ChatGPT to explain the code

CJ

Explain the second Python code line below:

movie = bs_obj.findAll("tr")
title = movie[1].find("a", {"class": "a-link-normal"}).text

The second Python code line `title = movie[1].find("a", {"class": "a-link-normal"}).text` is extracting the text content of a specific HTML element within the `movie` object, which is assumed to be a list-like object containing the extracted `<tr>` elements from the HTML content.

Here's a breakdown of what the code does:

1. `movie[1]`: This expression retrieves the second element from the `movie` list by using indexing `[1]`. In Python, indexing starts from 0, so `movie[1]` corresponds to the second element in the list.
2. `.find("a", {"class": "a-link-normal"})`: This method is called on the `movie[1]` element to search for an `<a>` element (anchor tag) with a specific class attribute. The `find()` method in BeautifulSoup is used to locate the first element that matches the specified parameters. In this case, it searches for an anchor tag with the class attribute `"a-link-normal"`.
3. `.text`: The `.text` attribute is accessed to retrieve the text content of the selected HTML element. It returns the inner text of the element, excluding any HTML tags.

3) Let's try this extraction process for gross

td.a-text-right.mojo-field-type-money.mojo-estimatable 99 × 31

Rank	Release				ease Date	Distributor
1	The Super Mario Bros. Movie	$570,554,370	4,371	$570,554,370	Apr 5	Universal Pictures
2	Guardians of the Galaxy Vol. 3	$336,521,512	4,450	$336,521,512	May 5	Walt Disney Studios Motion Pictures
3	Avatar: The Way of Water	$283,067,859	4,340	$684,075,767	Dec 16	20th Century Studios
4	Spider-Man: Across the Spider-Verse	$232,339,759	4,332	$232,339,759	Jun 2	Columbia Pictures

```
</td>
<td class="a-text-left mojo-field-type-genre hidden" style="width: 0px; height: 0px; min-width: 0px; min-he
ight: 0px;">–</td>
<td class="a-text-right mojo-field-type-money hidden" style="width: 0px; height: 0px; min-width: 0px; min-h
eight: 0px;">–</td>
<td class="a-text-right mojo-field-type-duration hidden" style="width: 0px; height: 0px; min-width: 0px; mi
n-height: 0px;">–</td>
<td class="a-text-right mojo-field-type-money mojo-estimatable" style="width: 99px; height: 31px; min-widt
h: 99px; min-height: 31px;">$570,554,370</td> == $0
<td class="a-text-right mojo-field-type-positive_integer" style="width: 73px; height: 31px; min-width: 73p
x; min-height: 31px;">4,371</td>
<td class="a-text-right mojo-field-type-money mojo-estimatable" style="width: 99px; height: 31px; min-widt
h: 99px; min-height: 31px;">$570,554,370</td>
```

a. Element="td"; Attribute="class"; Attribute value="a-text-right mojo-field-type-money mojo-estimatable"

```
gross = movie[1].find("td", {"class": "a-text-right mojo-field-type-money mojo-estimatable"}).text
print(gross)
```

```
$570,554,370
```

4. *Make a table for the extracted information*
   1) Use the pandas package to create a table
   2) Ask ChatGPT to know more about pandas package in Python

CJ

What is pandas package in Python?

The pandas package in Python is a widely used open-source library that provides high-performance data manipulation and analysis tools. It is built on top of the NumPy library and offers data structures and functions for efficiently working with structured data.

The primary data structures provided by pandas are:

1. Series: A one-dimensional labeled array that can hold any data type. It is similar to a column in a spreadsheet or a single-column table in a database.
2. DataFrame: A two-dimensional labeled data structure that consists of rows and columns. It is similar to a spreadsheet or a table in a relational database. DataFrames are the most commonly used pandas data structure and are highly versatile for data manipulation and analysis tasks.

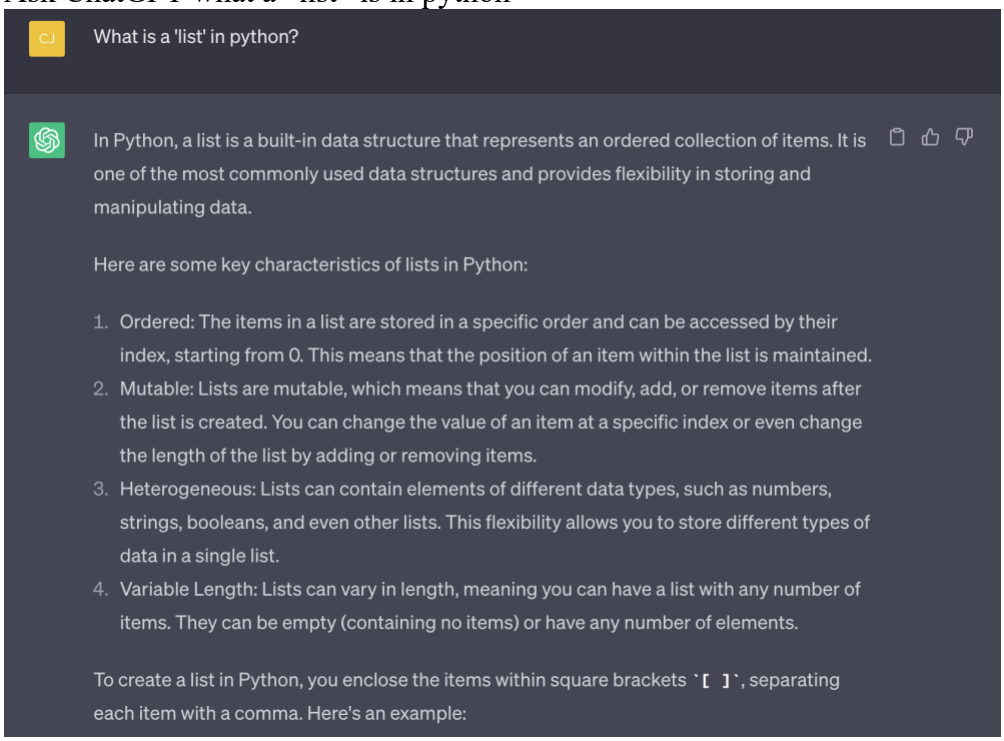
Pandas provides a wide range of functionalities and operations for working with data, including:

- Loading and writing data in various formats, such as CSV, Excel, SQL databases, and more.
- Cleaning, transforming, and manipulating data, including filtering, sorting, merging, and reshaping.
- Handling missing data and handling duplicates.
- Performing descriptive statistics and data aggregation.
- Applying mathematical and statistical operations to data.
- Visualizing data through built-in plotting capabilities.

3) Create containers that store values for columns
   a. The containers should be empty in a list form.

```
movie_title = []
movie_gross = []
```

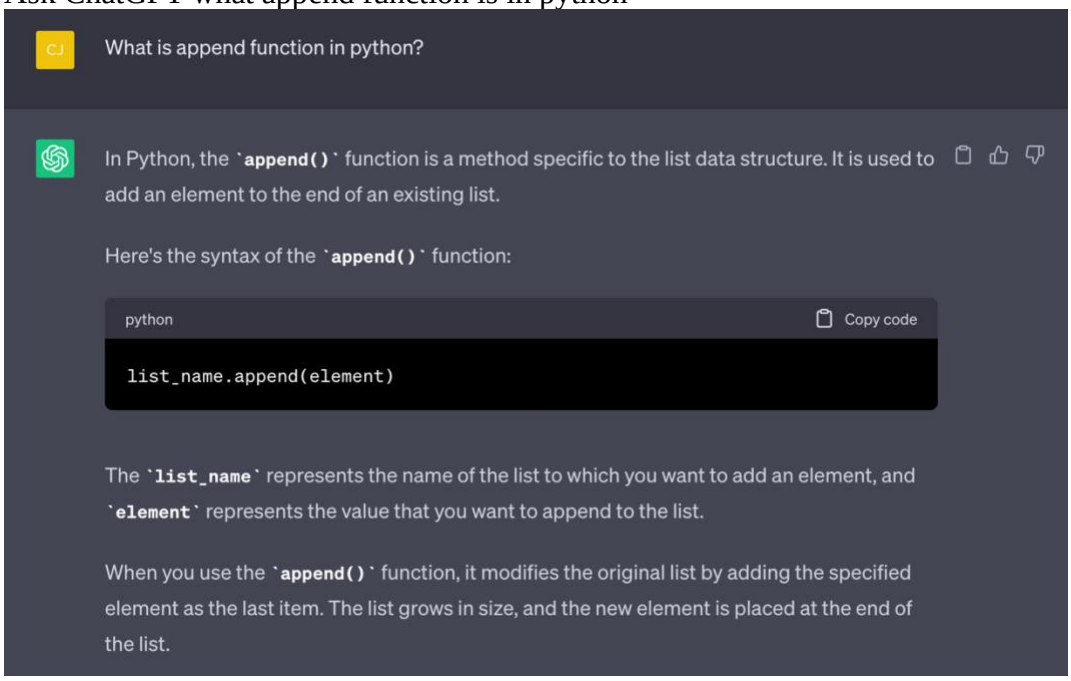
      i. List is a type of information
      Ask ChatGPT what a “list” is in python



4) Set the containers to accumulate the values extracted from the parsed HTML document.
   a. Use the append function

```
movie_title.append(title)
movie_gross.append(gross)
```

b. Ask ChatGPT what append function is in python



5) Put the values into the table and check the table

```
iter_df = pd.DataFrame({"Title": movie_title, "Gross": movie_gross})
print(iter_df)
```

```
                         Title         Gross
0  The Super Mario Bros. Movie  $570,554,370
```

5. *Looping the process to collect all the movie information in the webpage*

The next step is to develop the loop that extracts all 200 movies.

1) We need to see each movie in the movie variable and then extract the movie's title and gross sales.
   a. However, we should consider that the first "tr" element in the variable should be excluded from the loop.
   b. The loop function starts with "for"

i. Ask ChatGPT what a “for” loop is in Python

That is "for" loop in python?

In Python, a "for" loop is a control flow statement that allows you to iterate over a sequence (such as a list, tuple, string, or range) or any other iterable object. It repeatedly executes a block of code for each item in the sequence, until all items have been processed.

The basic syntax of a "for" loop in Python is as follows:

python Copy code

```python
for item in sequence:
    # code block to be executed
```

Here's a breakdown of how the "for" loop works:

1. `item`: It represents a variable name that you choose to use inside the loop. For each iteration, this variable takes on the value of the current item in the sequence.
2. `sequence`: It can be any iterable object or sequence that you want to iterate over. This can be a list, tuple, string, range, or any other object that provides iteration capabilities.
3. Code block: The indented code block following the "for" statement is the body of the loop. It contains the code that will be executed repeatedly for each item in the sequence.

During each iteration of the loop, the value of `item` changes, taking on the value of the next item in the sequence. The loop continues until all items in the sequence have been processed.

Here's an example that demonstrates the usage of a "for" loop to iterate over a list:

python Copy code

```python
fruits = ["apple", "banana", "cherry"]

for fruit in fruits:
    print(fruit)
```

Output:

Copy code

```
apple
banana
cherry
```

In this example, the "for" loop iterates over each item in the `fruits` list. For each iteration, the current item is assigned to the `fruit` variable, and the code block inside the loop (in this case, `print(fruit)`) is executed. The loop continues until all items in the list have been processed.

ChatGPT also gives you an example and the output from the example “for” loop.

c. Then, any indicator of each component in a variable should follow
   i. The example uses “item”
   ii. The indicator can be any words such as “abc” and “car” unless they are commands of Python

d. Then, the variable should be provided with its range
   i. If we use all components in the movie variable, we simply put “movie” but, we should exclude the first component with non-movie information. So, we need to set the range starting from 1 (The number in commands always start from 0 in Python as mentioned before and so 1 indicates the second component).
   ii. The end of the range is set as 5 for the practice this time

```
for item in movie[1:5]:
    title = item.find("a", {"class": "a-link-normal"}).text
    gross = item.find("td", {"class": "a-text-right mojo-field-type-money mojo-estimatable"}).text

    movie_title.append(title)
    movie_gross.append(gross)
```

The commands for the loop should be indented

Movie[1] should be replaced by “item”, the component indicator in the loop.

The return of the process is seen below. As you can see, the return has only four movies. It means that although the range is set as 1 through 5 (movie[1:5]), the last number of 5 is not counted as the range.

```
movie_title = []
movie_gross = []

for item in movie[1:5]:
    title = item.find("a", {"class": "a-link-normal"}).text
    gross = item.find("td", {"class": "a-text-right mojo-field-type-money mojo-estimatable"}).text

    movie_title.append(title)
    movie_gross.append(gross)

iter_df = pd.DataFrame({"Title": movie_title, "Gross": movie_gross})
print(iter_df)
```

```
                                Title         Gross
0          The Super Mario Bros. Movie  $570,554,370
1       Guardians of the Galaxy Vol. 3  $336,521,512
2             Avatar: The Way of Water  $283,067,859
3  Spider-Man: Across the Spider-Verse  $232,339,759
```

2) So, when expanding the process to the entire movie variable (200 movies), we need to set the range to end with 201 in order to collect all 200 movies.

```
for item in movie[1:201]:
    title = item.find("a", {"class": "a-link-normal"}).text
    gross = item.find("td", {"class": "a-text-right mojo-field-type-money mojo-estimatable"}).text

    movie_title.append(title)
    movie_gross.append(gross)
```

3) Check the return with a command "print iter_df"

```
df = "https://www.boxofficemojo.com/year/2021/?ref_=bo_yl_table_2"
page = requests.get(df)
bs_obj = bs4.BeautifulSoup(page.content, "html.parser")
movie = bs_obj.findAll("tr")

movie_title = []
movie_gross = []

for item in movie[1:201]:
    title = item.find("a", {"class": "a-link-normal"}).text
    gross = item.find("td", {"class": "a-text-right mojo-field-type-money mojo-estimatable"}).text

    movie_title.append(title)
    movie_gross.append(gross)

iter_df = pd.DataFrame({"Title": movie_title, "Gross": movie_gross})
print(iter_df)
```

```
                                      Title         Gross
0                       Spider-Man: No Way Home  $572,984,769
1     Shang-Chi and the Legend of the Ten Rings  $224,543,292
2                  Venom: Let There Be Carnage  $212,609,036
3                                  Black Widow  $183,651,655
4                            F9: The Fast Saga  $173,005,945
..                                         ...           ...
195                                      Ailey      $204,513
196                                Groundhog Day      $200,989
197                                   The Resort      $191,996
198                 Kaamelott: First Installment      $188,000
199                         Earwig and the Witch      $173,704

[200 rows x 2 columns]
```

a. All 200 movies and their gross are collected (from 0 to 199).