Ans1: requests module lets you easily download files from the web without having to worry about complicated issues such as network errors, connection problems, and data compression. **Webbrowser** Comes with Python and opens a browser to a specific page. **bs4** Parses HTML, the format that web pages are written in. S**elenium** Launches and controls a web browser. The selenium module is able to fill in forms and simulate mouse clicks in this browser.

Ans2: requests.get()’s return value, we can see that it returns a Response object, which contains the response that the web server gave for your request. The requests.get() function takes a string of a URL to download, To access we can use the .text attribute of the Response object.

Ans3: If the request for this web page succeeded by checking the status\_code attribute of the Response object. If it is equal to the value of requests.codes.ok, then everything went fine

Ans4: The status code for “OK” in the HTTP protocol is 200. We can get by response.status\_code == 200 in if else conditional statements

Ans 5: We save the web page to a file on our hard drive with the standard open() function and write() method. must open the file in write binary mode by passing the string 'wb' as the second argument to open()and then we Use a for loop with iter\_content(chunk\_size) to download and write the content in manageable chunks

Ans6: In Firefox, you can bring up the Web Developer Tools Inspector by pressing CTRL-SHIFT-C on Windows and Linux or by pressing image-OPTION-C on macOS. Pressing F12 again will make the developer tools disappear. In Chrome, you can also bring up the developer tools by selecting **View** ▸ **Developer** ▸ **Developer Tools**. In macOS, pressing image-OPTION-I will open Chrome’s Developer Tools.

Ans7 : We can right-click any part of the web page and select **Inspect Element** from the context menu to bring up the HTML responsible for that part of the page.

Ans8: The # symbol is used in CSS selectors to target an element by its id. Example: #main. Targets the element with id="main"

Ans9: The . (dot) prefix in a CSS selector is used to target elements by their class attribute. Example: .main .Selects all elements with the class="main".

Ans10

Ans11

Ans12

Ans13 Tag values also have an attrs attribute that shows all the HTML attributes of the tag as a dictionary.

**import bs4**

**exampleFile = open('example.html')**  
**exampleSoup = bs4.BeautifulSoup(exampleFile.read(), 'html.parser')**  
Link**elem = exampleSoup.select('#author').attrs**

**print(linkelem)**

**Ans14** Install selenium by running pip install --user selenium from a command line terminal. Instead of import selenium, we need to run from selenium import webdriver

ANs15 The find\_element\_\* methods return a single WebElement object, representing the first element on the page that matches your query. The find\_elements\_\* methods return a list of WebElement\_\* objects for every matching element on the page.

Ans16 WebElement objects returned from the find\_element\_\* and find\_elements\_\* methods have a click() method that simulates a mouse click on that element.

The selenium module has a module for keyboard keys. These values are stored in attributes in the selenium.webdriver.common.keys module. Run from selenium.webdriver.common.keys import Keys at the top of your program

Ans17 An easier way to submit a form with Selenium is to call the .submit() method directly on the form element. For example :

search\_box.send\_keys("Selenium Python")

search\_box.submit()

Ans18 The selenium module can simulate clicks on various browser buttons as well through the following methods:

**browser.back()** Clicks the Back button.

**browser.forward()** Clicks the Forward button.

**browser.refresh()** Clicks the Refresh/Reload button.

**browser.quit()** Clicks the Close Window button.