## Selenium Lab: Automating the 113 Attendance Form

Every class in 113, the TAs make a TinyURL that will link you to the daily attendance form. The only modification necessary for different forms is to change the TinyURL link.

Using Selenium, automate filling out today's attendance form (and future ones, too). Here are your tasks:

1. Open the link to the form
2. Find the element corresponding to the "AndrewID" text input box element and store it in a variable
3. Input your AndrewID into the text input box
4. Find the element corresponding to the submit button
5. Click the submit button

Running the code successfully will give you the attendance for the February 9th, 2023 lecture of 113! Ask for help during lecture if you need to, and if you aren't able to finish the lab before the end of lecture talk to the TA's and we can check you off. We'll also be available to help you outside of lecture through Piazza or Zoom, if requested. If you finish before the lecture ends, feel free to play around with Selenium!

The daily phrase is "selenium-test". Here's a hint for free. If you can't access the text input box, put in a time.sleep() after you call driver.get(). This will let the page load completely, and include all elements you're looking for.

Your starter code is on the next page. Useful syntax and tips are at the end of the lab handout. Have fun!

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

PATH = "<<INSERT PATH HERE>>"
driver = webdriver.Chrome(PATH)

def attendanceFiller(dailyPhrase):

    tinyURL = 'https://tinyurl.com/'

    driver.get(_____)

    andrewID = "<<INSERT ANDREWID HERE>>"

    ### COMPLETE YOUR TASKS ###

    driver.quit()

attendanceFiller("selenium-test")
```

Here's some syntax you might find useful:

```
driver.get("<link>")
```

This command takes a link in the form of a string as an argument, and opens the link in the current tab that Selenium is controlling.

```
driver.find_element(By.<method>, "<string argument>")
```

This command will return **the first** HTML element that has the attributes specified on the current page Selenium has pulled up. <method> can be filled by any of the options discussed in the HTML overview: ID (By.ID), name (By.NAME), class name (By.CLASS_NAME), XPath (By.XPATH), CSS selector (By.CSS_SELECTOR), link text (By.LINK_TEXT), partial link text (By.PARTIAL_LINK_TEXT), and tag name (By.TAG_NAME). The way to find the string argument is defined in the HTML overview in the beginning of this writeup, and is also described in the demos. **Selenium will return an error if it does not find any element that matches the specifications.**

```
driver.find_elements(By.<method>, <string argument>)
```

This command is the same as driver.find_element except it returns a list of **all elements** on the current page Selenium has pulled up that contain the attributes specified. **Selenium will return an empty list if it does not find any element that matches the specifications.**

```
<element name>.click()
```

This command will attempt to click the current element stored in <element name>. **If it is not a clickable/interactable element, Selenium will raise an error message.**

```
<element name>.send_keys(<keys>)
```

This command will attempt to send the keys in the argument to the current element stored in <element name>. **If the element cannot have keys sent to it, Selenium will raise an error message.** <keys> can be a string with the text that the user wants to put into the element, or keys using the object Keys must be imported.

## HINTS

(Highlight the text in the pdf to see them)

HINT 1:

HINT 2:

HINT 3:

HINT 4:

HINT 6:

If you've tried all these hints and are still stuck, let a TA know! Selenium is difficult to figure out and can be very very finicky… we know how it feels.