Documentation:

Functions of our code:

- Given a root website (University's Web Page URL) and its name (e.g. UIUC, UCB ....), the program will automatically find its Faculty directory page.
- After getting the Faculty directory page, the program will automatically find all of its faculty homepages.
- Then according to the user's instructions, the program will output a given number of Faculty Information including their Phone, Email, past College, personal bio, Research Area, and earned Awards.
- This program can be used for CS students to search desired colleges' professors, and get their information in a convenient way (the program will filter useful information for them).

How code is implemented:

- For the first part, we use Selenium to find and interact with College's search bar in order to find candidates for the Faculty directory page url (directory.py). Since some college's search system is not perfect, we combine search results from google to make sure our candidate list includes the true Faculty directory page url. Then we build Random Forest Classifier (rfclassifier1.py, one model - rfclassifier1) with manually collected training data to classify the correct directory page url.(getfeature.py)
- Second part, we keep using Selenium and Requests/Bs4 to find candidates' url list of Faculty Homepage. Since there is too much candidate url, we use some conditions to filter out as much as possible url. Then building another Random Forest Classifier (rfclassifier2.py, one model - rfclassifier2) to find corrected Faculty Homepages. (getfeature.py)
- Inside each Faculty Homepage, we first get the html file and filter out useless information (dataprocess.py), and make all useful information into a list. Then we build a tfidf vectorizer as the feature to train our text classifier (Random Forest.py, two models - Vectorizer and text_classifier). Then we use the classifier to classify our desired information (bio,education,awards,research interests).

How to run the code:

- All the main code is inside directory.py. In order to run it, just need to run ./directory.py and make sure the computer has the correct version of browser to fit with Selenium. And then input College Url, College Name, and desired number of output.
- Environment requirement: beautifulsoup4, selenium, numpy, pandas, seaborn, scikit-learn, matplotlib

Contribution of Team member:

- Guanhua Li: Implementing the Selenium/Web crawler part and building models for classifiers.
- Ruoyu Zhu: Collect dataset and find features for models.
- Ziqi Li: Find features for models , scrape features from webpages and function1 coding.