**BERKE ÖZDEMİR / 2016400246**

**İBRAHİM ÖZGÜRCAN ÖZTAŞ / 2016400198**

# CMPE 230 PROJECT 2

**April 28, 2019**

## Overview

In our project, we are responsible to scrape throughout Bogazici University Registration Page (http://registration.boun.edu.tr) to retrieve course informations based on several traits. Then, print out whole courses via comma separated value form (.csv)

## Goals

1. **Scrape the site to fetch data:** By using beautifulsoup4 library in python, data of the courses served by the university is fetched and transformed into viable form.
2. **Print out via comma separated value:** After data transformation, processed data is ready to be served via comma separated value (.csv) format in terminal.

## Specifications

Code is based on functions to format data or retrieve data from current url. Several functions are created to aid the essential functions. Comments of those functions are deliberately explained and formed.

Constructing the table:

Delimiter used is ‘;’. Method first constructs the header with department name and lengths of U, G , I sets for the departments and the individual semesters. Then it constructs the string for each line. Given each course corresponds to a line, each individual element of couse_dict is used to create the string. Relevant data needed for this is all grouped together in course_dict. It adds all lines to a list of strings, then sorts it by course code(the order to print) and returns the list for printing.

## Essential Functions

1. main():

   Program starts with taking the given semesters and constructing a list of each semester included in range of the two, that is a string pluggable to the url for the fetching operation. The departments are hard coded in a list containing their department code, keyword to plug to the url, and their department name. Then, with the help of nested loops, we fetch the data for each department, by each year. In the end, we have the data of courses for each department, grouped together in a list. Then, we construct the string for the header table in the given format, we use the semesters list to construct portion of the header that has the name each semester. Then, with the parse_list method  process the data for each department, and get the strings to construct a comma separated value table (.csv) line by line for the respective departments. We then print the data line by line.

2. get_function(department, semester)

   This method takes the department, semester string and constructs an url from the url portion of department and semesters. Then, sends a request to web page of the constructed url and gets the html data. We process the html data with the help of BeautifulSoup4 and construct a list of [course_code,semester, course_name, instructor] from each of the rows of the html file. Puts them all in one list and returns the whole data for the given department, semester.

3. parse_list(data, semesters, department)

   This method takes the data of courses for a whole department and constructs a csv formatted table line by line from that data, as a string. We have variables U_set(), G_set(), and instructor_set() to determine numbers of unique undergrad, grad courses and instructors for the department, then we have a semesters_dict, mapping each semester to a list of its [number_of_undergrad_courses, number_of_grad_courses, distinct_instructors_set_for semester]. Then, course_dict map each course code to the list of [full_name_for_the_course, set_of_semesters_the_course_has_been_given, distinct_instructors_giving_the_said_course]. Then, for each element of data, it checks if the course is a grad or undergrad course, increases the number of undergrad (or grad) courses given for the department and semester, also adds it to the set of distinct courses for the department if it wasn't added.  Add the instructor to the list of distinct instructors for the department and the relevant semester. Then, it checks if course_dict already has data for the course. If it is method updates the sets of instructors giving the course and semester the course is given in , if it is not in the dictionary, method adds it instead. After iteration of each data element is completed it starts constructing the comma separated value string.