**HOMEWORK 2 - CMPE230**

DOCUMENTATION

Ömer Faruk Özdemir – 2016400048

Yusuf Başpınar - 2014400042

In this project, we developed a Python program called **bucourses.py** that crawls Bogazici University’s OBIKAS registration pages and extract course offering information.

Our Python program crawls course offerings of all departments/programs for all semesters given in a range by the user as:

**./bucourses.py StartYear-Term EndYear-Term**

**deps array:**

We stored all the Departments like “ASIA&bolum=ASIAN+STUDIES” to fill the url part accordingly.

After filling deps array we parse the input given by the user into start-term and end-term, we converted into proper url form.

We treated as different the courses though their shortcodes are the same:

Shortcode = ASIA CourseName = ASIAN+STUDIES+WITH+THESIS

Shortcode = ASIA CourseName = ASIAN+STUDIES

**ConvertYear(year):** We deconverted 2017/2018-2 to 2018-Spring etc. while parsing given input.

**IsGrad(courseCode):** Returns True if given couseCode is greater than 4xx

**ChangeComa(myString):** Changes “,” s to “;” in courseCodes to maintain csv format.

**GetCourseCode(myString):** Changes couseCode with section code to only couseCode

**GetDepartmentname(myString):** Returns departmentname from url with “%26” to “&” and “+” to “ “

**DepMethod(depName):**

Our main method. Takes depname as parameter from deps array and fill the url part and begins crawling into urls. First parses <tr> field with class ‘schtd2’ as gray zone and <tr> field with class ‘schtd2’ as white zone. Appends each row of the page into bodyArray then from bodyArray to bodiesTemp. From bodiesTemp to bodies as pages. From bodies to allBodies as a whole department with given intermediate from input. We use bodyArray fields as follows(It is 13 fields in pages but we use 3):

bodiesTemp.append(bodyArray[0]) # CourseCode

bodiesTemp.append(bodyArray[2]) # CourseName

bodiesTemp.append(bodyArray[5]) # CourseInstructor

bodiesTemp.append(whichTerm) # CourseTermNumber

Then we sort courses according to coursecode and we begin to mark with Xs to courses whether it is in the given term. Discard ‘STAFF’ as instructor and do this operation to all pages then save it to xList array. Calculate TotalOfferings as #/# at the end of the row.

After that we started to create the table as titles beginningRow string and for the follow ups departmentTable.

**CalculateCourseGivenInTermsNumber(xList):** Parse xList to count Courses in terms.

**CountUGI(allBodies, allBodies2):**

Traverse allBodies(gray zones) and allBodies2(white zones) and concatenate findings about Undergraduate(U), Graduate(G) and Instructor(I). Return numbers of U, G, I to show in table. At the end sum all findings and calculate Total.

allBodies -> bodies -> bodyArray -> Fields(CourseCode, CourseInstructor)

From CourseCode count Graduates and Undergraduates checking number 4xx or lower and vice versa.