|  |
| --- |
|  |
| **HOMEWORK 2 REPORT** |
| **CmpE 230** |

|  |
| --- |
| **Bekir Yıldırım - 201400054**  **Alkım Ece Toprak - 2017400294** |

1. **BASICS**

In this project, we were required to develop a Python that extracts information from Bogazici University’s OBIKAS registration pages and output the required data in CSV table format. In order to implement this, we started out by learning about HTML and CSV formats and how to analyze them through given links.

After some research, we learned about Beautiful Soup which is a Python library. This library is used to pull data out of HTML files. We decided to use this library to manipulate the HTML format we have and obtain the information we want from it.

We learned the basics of Beautiful Soup and started looking at the OBIKAS registration pages’ HTML files. Analyzing the general format, we understood which parts we should pull to get the course names, codes, hours, instructors and semesters.

1. **IMPROVING**

After grasping the main idea, we started implementing different functions to help obtain and sort the data. Our code consists of 13 functions that work together to manipulate the input and create the output

1. removeSpaces(string)
2. toString(List)
3. removeAll(string)
4. get\_data(year, semester, departmentshort="CMPE",departmentlong="COMPUTER+ENGINEERING")
5. boolean(List)
6. recursive(year1, semester1, year2, semester2, deps)
7. constructFirst(year1, year2, semester1, semester2, semesters)
8. construct\_table(courseList, year1, year2, semester1, semester2 ,semesters, deps, dictList)
9. fillspaces(listTemp, stringCourse, length)
10. coursecode(listTemp, course\_temp, temp, control)
11. xsign(listTemp, a, allList, semesterList)
12. totalofferings(listTemp, a, allList)
13. sortedfunc(listTemp, control)

First 3 functions are general basic functions that alter a given list or string to make them follow a format.

Get\_data, boolean and recursive are functions to pull data from URL’s and store them in different dictionaries to be used later.

Rest of the functions are used to create the output table by manipulating the data we have in different dictionaries.

1. **PROBLEMS**

After implementing the first version of the code, we realized that there are many exceptions. Even though HTML formats are similar in general, courses that opened differ from year to year and semester to semester. Some courses have lab sessions and there are courses with undefined instructors.

We handled these exceptions using different ways. To take the data in the given range, we created variables to store the starting and ending year and semester: year1, year2, semester1, semester2.

Additional boolean variables checkSemester1 and checkSemester2 are used to keep pulling data. If checkSemester1 becomes true, it means that the starting year and semester has passed and we are in the required range. If checkSemester2 becomes true, it means that the ending year and semester has passed and we are out of the required range. If checkSemester2 is true, we increase a var called “a” and stop pulling more data.

Another problem is that some courses do not open in certain semesters. To handle this, we added try and except parts:

try:

data = get\_data(year, semester, dep[0], dep[1])

exist = True

existList.extend([exist])

except:

exist = False

existList.extend([exist])

continue

Existlist is a boolean list which is used to store courses that did not open. If a course did not open, it is not written in the output dataframe.

When “staff staff” is seen, chackStaff becomes false and it is stored in the semesterL dictionary. This is checked when forming the output data and acted accordingly.

To solve the problem on “ Staff staff” we decreased the length of the lists when “staff staff” is seen and ignore them:

# If there is "Staff Staff" in the instructors, decrease the length by 1

if courseList[dep][1]['I'][1] == False :

length\_I = len(courseList[dep][1]['I'][0]) - 1

#else, keep it the same

else :

length\_I = len(courseList[dep][1]['I'][0])

Another problem was that there are 2 departments with the same short name ASIA: Asian studies with thesis and Asian studies. To solve this, we considered the same course twice and counted them for both of the departments. So a course with the code ASIA is written twice in the output table.

Another issue is that for some courses, different course names are seen with the same code such as AD409,BUSINESS ETHICS and AD409,BUSINESS ETHICS(G-Z). We considered them to be different courses and wrote them separately in the output CSV.