Documentation

Function #1: def csv_func(course_num)

This function takes a course number as an argument and reads in a csv file into a pandas dataframe containing information on faculty course evaluations. Using the course number that is passed into the function, the rows of the faculty course evaluations dataframe that have the specified course number as an index are returned to the user, after being passed back to the menu_execution function.

Function #2: def getHeinzCourseCatalog(course_num):

This function takes a course number as an argument and then appends that course number string to a url string for the Heinz Course Catalog. The contents of the course page related to the specified course number are returned and then parsed according to the section headers on the course page, namely course units, class name, description, learning outcomes, prerequisites, and syllabi. If there is no associated course page, then the user is returned to the menu_execution() function. If the course page does exist for the course, then a list is returned with the course number, course name, number of course units, description, learning outcomes, prerequisites, and the syllabi.

Function #3: def menu():

This function displays a list of menu options to the user and prompts them to input the number associated with the menu option that they would like. This input is then converted to an integer value. If this integer is not in the list of allowable menu options, then the user is continually prompted until they enter a valid menu option. The integer value for the menu option is returned from this function to wherever it was called from.

Function #4: def menu_execution(course_num , request_code)

This function takes two values as arguments -- a course number and the menu option as integer for the action that the user wants to see for the specified course number. This function then performs the desired action for the user, based on the request code (or menu option) that they previously selected. After executing the action that the user wanted, the user is then prompted again for a new menu option.

Function #5: def request1(course_num):

This function takes a course number as an argument and, as long as the course number exists in the Schedule of Classes dictionary for Fall 2019, prints information about the professor, date and time, location, and campus for each instance of the course for the Fall 2019 semester.

Function #6: def request2(course_num):

This function takes a course number as an argument, and as long as this course exists in the Schedule of Classes dictionary for Fall 2019, the department that the course belongs to is printed out for the user.

Function #7: def request3(course_num):

This function takes a course number as an argument and converts the format of the course number to one where the dash between the first two and last three numbers is removed. Then, this newly formatted course number is passed as an argument to the “csv_func” function to get the faculty course evaluation information associated with that course number, if it exists.

Function #8: syllabi_pull(course_num):

This function takes a course number as an argument and calls the “getHeinzCourseCatalog” function. From the list that is returned from this function call, the last element of the list, which contains a dictionary of all of the syllabi for a course, is isolated from the list. Then, a print statement is executed to iterate over the keys in the dictionary and print all of the available syllabi.

Function #9: def main():
This function asks the user to input a course number (in XX-XXX format) that they want more information about. This input is then stored in the “course_num” variable. Then, the function uses a regular expression to check that the inputting course number is in the appropriate format. If the inputted string is not in the correct format, then the user is continually prompted until the course number is in the correct format. Then the “menu_execution” function is called to return the desired information to the user about a course of interest to them.

The only code that exists outside of any function is code that queries the ScottyLabs API. This code is enclosed in an if-else statement. The if statement checks if the user already has a file name called “filtered_courses.json” in their file path. If they don’t, then the api is queried to get information from the Carnegie Mellon University Schedule of Classes. This information is returned as a json file. The data is filtered to just include information from Heinz, whose courses start with the numbers 90-95. A few of the columns returned are also dropped. This if-else statement exists because there are a limited number of times in which the ScottyLabs API can be queried. As a result of this, we wanted to create a control within our program so that the API could not be queried every single time the program was run.