|  |
| --- |
| **def** display\_timetable**(**selection**:** **str,** selection\_type**:** **str,** timetable**:** **list):**  """  Takes timetable and displays information based on selection.  :param selection: name of selection - E.g. Ada Log, ABC101  :param selection\_type: type of selection - Campus, Lecturer, Subject, Room  :param timetable: list created from csv  """  **print(**f'Timetable for {selection\_type} "{selection}":'**)**  **print(**'=' **\*** 80**)**  **print(**'Subject Activity Day' **+** **(**' ' **\*** 7**)** **+** 'Start' **+** **(**' ' **\*** 5**)** **+** 'End' **+**  **(**' ' **\*** 7**)** **+** 'Campus' **+** **(**' ' **\*** 4**)** **+** 'Room' **+** **(**' ' **\*** 6**)** **+** 'Lecturer '**)**  **print(**'-' **\*** 80**)**  # Iterates through timetable and  **for** line **in** timetable**:**  **if** selection **in** line**:**  **for** i **in** **range(**8**):**  **if** i **!=** 7**:**  **print(**line**[**i**]** **+** **(**' ' **\*** **(**10 **-** **len(**line**[**i**]))),** end**=**''**)**  **else:**  **print(**line**[**i**]** **+** **(**' ' **\*** **(**10 **-** **len(**line**[**i**]))),** end**=**'\n'**)**  **print(**'=' **\*** 80**)**  **def** sort\_data**(**file**:** **str)** **->** **tuple[list,** **list]:**  """  Opens file and sorts data into subjects, campuses, rooms and lecturers.  It also puts each line of the csv into a list.  :param file: string that contains file name  :return: 2 lists, first is the full timetable, second is the selections list  """  file **=** **open(**file**,** 'r+t'**)**  # creating 5 separate empty lists  timetable**,** lecturers**,** rooms**,** subjects**,** campuses **=** **([]** **for** i **in** **range(**5**))**  **for** line **in** file**:**  row **=** line**.**strip**().**split**(**','**)**  timetable**.**append**(**row**)**  # Conditional statements used to sort data into lists of each option without any duplication  **if** row**[**0**]** **not** **in** subjects**:**  subjects**.**append**(**row**[**0**])**  **if** row**[**5**]** **not** **in** campuses**:**  campuses**.**append**(**row**[**5**])**  **if** row**[**6**]** **not** **in** rooms**:**  rooms**.**append**(**row**[**6**])**  **if** row**[**7**]** **not** **in** lecturers**:**  lecturers**.**append**(**row**[**7**])**  selection\_list **=** **[**campuses**,** lecturers**,** rooms**,** subjects**]**  file**.**close**()**  **return** timetable**,** selection\_list  **def** sub\_menu**(**selection**:** **str,** file**:** **str):**  """  Takes the selection made in the main menu and displays it's sub menu.  :param selection: string from main menu based on the selections made  :param file: string that contains file name  """  timetable**,** selection\_list **=** sort\_data**(**file**)**  # Conditional statements process selection from main menu  **if** selection **==** 'c'**:**  **print(**'\nCampuses\n' **+** **(**'-' **\*** 8**))**  selection\_name **=** 'campus'  options **=** selection\_list**[**0**]**  **elif** selection **==** 'l'**:**  **print(**'\nLecturers\n' **+** **(**'-' **\*** 9**))**  selection\_name **=** 'lecturer'  options **=** selection\_list**[**1**]**  **elif** selection **==** 'r'**:**  **print(**'\nRooms\n' **+** **(**'-' **\*** 5**))**  selection\_name **=** 'room'  options **=** selection\_list**[**2**]**  **elif** selection **==** 's'**:**  **print(**'\nSubjects\n' **+** **(**'-' **\*** 8**))**  selection\_name **=** 'subject'  options **=** selection\_list**[**3**]**  # selection\_list is passed to display\_selection function  valid\_input**,** user\_input **=** display\_selection**(**options**)**  **while** **not** valid\_input**:**  **print(**'\nInvalid selection. Please try again.\n'**)**  valid\_input**,** user\_input **=** display\_selection**(**options**)**  display\_timetable**(**options**[**user\_input **-** 1**],** selection\_name**,** timetable**)**  **def** display\_selection**(**options**:** **list)** **->** **bool** **and** **int:**  """  Takes the list of options and then uses a loop to print out each option  :param options: list of campuses or lecturers or rooms or subjects  :return: boolean value and an int - int will be 0 if boolean is False  """  count **=** 1  **for** option **in** options**:**  **print(**f'[{count}] {option}'**)**  count **+=** 1  # Using a try except statement with a conditional statement after to handle incorrect inputs from user  **try:**  user\_input **=** **int(input(**'\n> Enter selection: '**))**  **except** **ValueError:**  **return** **False,** 0  **if** **not** 0 **>=** user\_input **<** **len(**options**):**  **return** **True,** user\_input  **else:**  **return** **False,** 0  **def** main\_menu**(**file**:** **str):**  """  Displays main selection menu.  :param file: string that contains file name  """  selection\_list **=** **[**'c'**,** 'r'**,** 's'**,** 'l'**,** 'q'**]**  **print(**'\nTimetable menu\n' **+** **(**'-' **\*** 14**)** **+** '\n[C]ampus\n[R]oom\n[S]ubject\n[L]ecturer\n'**)**  selection **=** **input(**'> Enter selection or [Q]uit: '**).**lower**()**  **while** selection **not** **in** selection\_list**:**  **print(**'\nInvalid selection. Please try again.'**)**  **print(**'\nTimetable menu\n' **+** **(**'-' **\*** 14**)** **+** '\n[C]ampus\n[R]oom\n[S]ubject\n[L]ecturer\n'**)**  selection **=** **input(**'> Enter selection or [Q]uit: '**).**lower**()**  **if** selection **==** selection\_list**[**4**]:**  **print(**'\nHave a nice day!'**)**  **quit(**0**)**  **else:**  sub\_menu**(**selection**,** file**)**  **def** file\_exists**(**file**:** **str)** **->** **bool:**  """  Tries to open the file name parsed through.  If it cannot open a file it raises an IOError and returns false.  :param file: string that contains file name  :return: boolean value  """  **try:**  file **=** **open(**file**.**strip**(),** 'r+t'**)**  **except** **IOError:**  **return** **False**  **else:**  **return** **True**  **def** correct\_format**(**file**:** **str)** **->** **bool:**  """  Splits the file name, checks if the last element in the list is csv.  If it is a csv, it checks if each element in the file is formatted correctly.  :param file: string that contains file name  :return: boolean value  """  **if** file**.**split**(**'.'**)[-**1**]** **!=** 'csv'**:**  **return** **False**  **else:**  file **=** **open(**file**,** 'r+t'**)**  **for** line **in** file**:**  check\_line **=** line**.**strip**().**split**(**','**)**  **if** **len(**check\_line**)** **!=** 8**:**  **return** **False**  # Subject and Room checks  **if** **not** check\_line**[**0**].**isalnum**()** **or** **not** check\_line**[**6**].**isalnum**():**  # checks if subject or room aren't alphanumeric  **return** **False**  **else:**  **if** **not** check\_line**[**0**][**0**].**isalpha**()** **or** **not** check\_line**[**6**][**0**].**isalpha**():**  # checks if the first character in subject or room is a letter  **return** **False**  # Activity, Day, Campus and Lecturer checks  **if** **not** check\_line**[**1**].**isalpha**()** **or** **not** check\_line**[**2**].**isalpha**()** **or** \  **not** check\_line**[**5**].**isalpha**()** **or** **not** check\_line**[**7**].**replace**(**' '**,** ''**).**isalpha**():**  # check\_line[7] and checkline[5] has any whitespaces removed for the condition check  **return** **False**  **return** **True**  **def** main\_loop**():**  **print(**'Timetable Organiser\n' **+** **(**'-' **\*** 19**))**  file **=** **input(**'\nEnter the timetable data file name: '**).**lower**()**  **while** **not** file\_exists**(**file**):**  **print(**"File doesn't exist. Please try again."**)**  file **=** **input(**'Enter the timetable data file name: '**).**lower**()**  **print(**'\*\* Reading timetable data ...'**)**  **if** **not** correct\_format**(**file**):**  **print(**'\*\* Error! File is not correctly formatted.\n\n\nHave a nice day!'**)**  **exit(**0**)**  **print(**'\*\* Success!'**)**  main\_menu**(**file**)**  **if** \_\_name\_\_ **==** '\_\_main\_\_'**:**  main\_loop**()** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Test Data** | **Reason it was selected** | **Expected output** | **Screenshot of actual output** |
| Normal | ITC106,Lecture,Tuesday,18:00,  19:00,Distance,N00000,Mostafa Dahshan  ITC132,Tutorial,Thursday,14:00,  15:00,Distance,D28969,Linus Torvalds | To test lecturer selection in timetable menu | Two options will appear, Mostafa Dahshan and Linus Torvalds. After selecting Mostafa Dahshan, dialog should show “Timetable for **lecturer “Mostafa Dahshan”**”. |  |
| Normal | ITC105,Lecture,Monday,17:00,  18:00,Distance,A10000,Anthony Chan  ITC162,Tutorial,Wednesday,12:00,  13:00,Distance,A91968,David Plummer | To test campus selection in timetable menu. | Only option should be “Distance”. After the selection is made, dialog should show “Timetable for **campus “Distance”**”. |  |
| Normal | ITC161,Tutorial,Monday,18:00,  19:00,Distance,X10232,Quazi Mamun  ITC101,Lecture,Tuesday,18:00,  19:00,Bathurst,X10645,Margaret Hamilton | To test subject selection in timetable menu | Two options will appear, ITC161 and ITC101. After selecting ITC161, dialog should show “Timetable for **subject “ITC161”**”. |  |
| Abnormal | ITC161,Tutorial,Monday,18:00,19:00,Distance,  ITC101,Lecture,Tuesday,18:00,19:00,Bathurst, | To test when the file has missing data fields | Program will display error message and exit |  |
| Abnormal | ITC106,Lecture,Tuesday,18:00,  19:00,Distance,N00000,Mostafa Dahshan  ITC132,Tutorial,Thursday,14:00,  15:00,Distance,D28969,Linus Torvalds  ITC105,Lecture,Monday,17:00,  18:00,Distance,A10000,Anthony Chan  ITC162,Tutorial,Wednesday,12:00,  13:00,Distance,A91968,David Plummer  ITC161,Tutorial,Monday,18:00,  19:00,Distance,X10232,Quazi Mamun  ITC101,Lecture,Tuesday,18:00,  19:00,Bathurst,X10645,Margaret Hamilton  ITC161,Tutorial,Monday,18:00,  19:00,Distance,X10232,Quazi Mamun  ITC101,Lecture,Tuesday,18:00,  19:00,Bathurst,X10645,Margaret Hamilton | To test incorrect input from user after selecting lecturer from timetable menu | Program will display “Invalid selection” and then you will be prompted again to make selection. |  |
|  |  |  |  |  |