**[All] Section**

* File: [All]\_Check\_Validity.cpp: This file contains functions to check whether the value is valid or not
  + The "CapitalClassName" function changes the class name string into the correct form (22clc02 -> 22CLC02)
  + "isValidStudentID" function checks if the student ID has 8 characters and the first 2 characters fit the year
  + "chechYear" check whether the newly created year existed or not
  + "checkClass" check whether a newly created class existed or not
  + "Check\_Student" check whether a newly created student existed or not
  + "isLeap" is to check if the year is a leap
  + "isValidDate" to check the validity of an inputting date
  + "isDay" checks the string if it is in MON, TUE, WED, THU, FRI, SAT, SUN
  + "isSession" to check if it is in S1, S2, S3, S4
  + "isValidGender" check input is M or F
  + "isValidYear" checks the format of the school year is yyyy\_yyyy
  + "SeparateName" separates the full name into first and last name
* File: [All]\_Del\_LL.cpp: This file consists of functions that delete all the created linked list
  + "DeleteYear" function deletes every node of the linked list and the class with the student node inside it.
  + "DeleteStudent" erase all instance of students in the linked list inside courses in the semester.
  + "DeleteCourse" erase all instance of courses in the linked list inside the semester.
  + "DeleteSMT" delete all semester in a current year and is put inside the "DeleteYear" function
  + "DelAccount" cleans up the space that the account linked list used.
* File: [All]\_Draw.cpp: contains all functions with the usage of drawing out the information that users can use the arrow button to move around.
  + The "Draw\_XY" takes in the list of options in order to print out the list of actions for the user to choose.
  + The "Draw\_table" takes in the 2d array to draw onto the console a table with its data passed in.
  + "Draw\_Horizontal\_XY" is another version of Draw\_XY but it is used to print out the table in the horizontal ways.
  + "Draw\_ShortVer" is another version of "Draw\_XY" but the printout list only covers 1 block instead of 3.
* File: [All]\_Output\_File.cpp: contains all methods of exporting files for saving purposes
  + The "OutCourse" exports the course information in a semester to the file where store the semester information. The information output is the name, courseID, credits, max students, and a current number of students, ...
  + "Output" exports the years, the semesters in the year, and the course information to the file 'in4smt.txt' in the corresponding folder.
  + "Write account" save the account and the user information to a file named 'account.csv'
  + "New account" is to create a new account for a new student and link it to the account linked list.
  + "Outyear" saves the number of years in the 'years.txt' alongside classes and students in classes.
  + "outScoreboard\_StudentCourse" and "outStudentCourse" export the student's score to the corresponding csv files.
  + "exportListStudentCourse" is used to create a folder and put allow those above functions to put files in it.
* File: [All]\_Readin\_File.cpp: contains all methods of loading information from saved files into linked lists
  + "ReadAccount" loads back the user's accounts into the account linked list
  + "loadingFile" starts the loading process by calling the 'importYear' function
  + "importYear" loads the year from the file into the year linked list
  + "importClass" loads the classes and "addClass" creates the class linked list
  + "Read\_Sem", "Read\_All\_Semester" takes the 'in4smt.txt' file to trace back all the saved information of courses and semesters
  + "readScoreStudentCourse" and "readStudentCourse" takes the score of students and save them to the list
  + "readAllFileCourses" and "importScoreBoardCourse" finds the files and call the above functions to read them.
* File: [All]\_Render\_Text.cpp: storing all functions to print out the artistry of words, which functions take in value x and y to get the coordinate to print
* File: [All]\_Role\_Menu.cpp: The menu of actions an account has
  + "Staff\_Main" is the function, the menu of action for a staff showing when enter, which is account options, view, add, modify information, import or export data.
  + "Student\_Main" is the menu options for student accounts that students can manage the account, view the course, and score.
  + "Main\_Menu" is called when the user enters the system after login successfully. It will decide based on the account role and show the corresponding menu.
* File: [All]\_Sort.cpp: has the function to swap the year
  + "SwapYear" is used to swap the inside information that stores in a year linked list
  + "SortYear\_Descending" swaps the year into the correct order.
* File: [All]\_Sync.cpp: has functions that allow the program to sync information across linked lists
  + "SyncInForStudent" traverses the linked list of years and finds the correct account node to link to the student.
  + "SyncCourse" gets all the course that student has put inside a linked list for better management.
  + "SyncFullName" traverse the year and account linked list to get the name of the student corresponding to the student ID.
* File: [All]\_Utilities.cpp: stores all the functions that use across files for managing the cursor and color purpose
  + "ShowConsoleCursor" is the function that shows or hides the cursor.
  + "Pause": use the ‘conio.h’ style to replicate the 'pause' command line function.
  + "Message\_Warning" and "Message\_YesNo" are functions to show a message box with the inputting text and a button for users to click.
  + "TextColor" changes the color of the console text.
  + "goToXY" moves the cursor to the inputting position.
  + "Limit\_Input" replicate the std::cin by using ‘conio.h’ library and limiting the inputting length.
  + "resizeWindow" resizes the console window to the fixed size.
  + "hideScrollBar" hides the console's scroll bar.
  + "initWindow" initializes the console window by setting the size, and hiding the scroll bar,...

**[Sem] Section**

* File: [Sem]\_Add\_Course\_Student.cpp: stores function to create a new course in semester and put in enrolled students
  + "AddNewCourse" takes in the current chosen semester and the year linked list, then it prints to the console the necessary information for the user to input including the course ID, its name, teacher, its credits, the maximum number of students, the room and its time table. Finally, the function creates a new node at the end of the course inside the selected semester and returns back the node itself.
  + "ImportStudentFromFile" makes the user input the file name that the user put in the specific folder, then it reads the file and imports the student information into the student linked list.
  + "AddStudentByHand" allows the user to input each student's information, then creates a new student node at the end of the linked list.
  + "AddingCourse" gives the user options to choose the way they want to add a student to the course, either by importing from a file or by hand.
* File: [Sem]\_Add\_Semester.cpp: holds functions that allow users to create new semesters or courses with students
  + “compareDate”: the function takes in 2 date strings and compare them, return -1, 0, 1 accordingly.
  + "AddSemester" is a function that takes in the year linked list, creates a new semester node at the end of the selected year, then allows the user to input the starting and ending date of the semester. Finally, it returns the semester that just has been created.
  + “isStringDigits” takes an inputted string and check whether is contains only number of not.
* File: [Sem]\_Initialize\_Modify.cpp has the 2 functions which call the corresponding functions
  + "initModify" function shows the options for the user to modify the selected item, which is semesters, courses, students, and scoreboards.
  + "RunModify" takes in the selected option and calls the corresponding function to modify the selected item.
* File: [Sem]\_Interface\_New\_Stuff.cpp: contains functions to relocation the program to the called one
  + "New\_Stuff" shows the menu that the program allows users to choose to add to the program, which is to add a new year, a new semester, or a new course.
  + "Interface\_New\_Sem" gives the option for the user to use the follow-up function after making the addition of a new semester.
* File: [Sem]\_Mod\_Course.cpp: has the functions to enable modifying course information, or remove it
  + "modifyCourse": first it finds the chosen year node, then allows the user to input the course ID, then perform the modification in "ChangeCourseInfo" function and notices to the user whether the modification is successful or the course is not found.
  + "ChangeCourseInfo" shows the current course's information and has the user to choose which information they want to modify and then update to the course node.
  + "removeCourse" first find the chosen year node, then gets the inputting course ID and checks whether it exists or not, if it exists, remove the course node and notify the user whether the course is removed successfully or not found.
* File: [Sem]\_Mod\_Sem\_Student.cpp contains functions that make modifications to semester and enrolled students in a course.
  + "modifySemester" takes in the full year linked list and traverses through it to get the needed year and performs modification of the semester's start and end date.
  + "addStudent" takes in the current course and the course ID to begin adding new students to the course by entering their student ID.
  + "removeStudent" other hand gets the course linked list and the course ID itself to remove a specific student from that chosen course.
  + "addRemoveStudent" is the function that allows users to choose whether they want to add or remove students from the course.
* File: [Sem]\_View\_Course.cpp: holds the option to print out courses in a year.
  + "Tablize" function takes in 0 for the head, 1 for the body, and 2 for the tail to draw the horizontal line of the table using ASCII characters.
  + "Part\_Of\_Course" prints out the list of courses inside a table with the year. Users can use the arrow keys to navigate through the table in semesters and years.
  + "ViewCourse" finds the number of pages that the list of courses fits in and handles the work of getting arrow keys to navigate the course for the 'Part\_Of\_Course' to print the course out.

**[Sem] Year**

* File: [Year]\_Add\_New\_Year\_Classes.cpp: holds functions that create new year node and classes addition operations
  + "createSchoolYear" first checks the year, then if not create a new year node at the end and create a new year path in the saving folder.
  + "Create\_New\_Classes" allows users to create new classes according to the newly created year and put them in the class linked list.
  + "ChooseClassToAdd" prints out the classes for users to choose to begin adding students.
  + "Method" contains the style the users want to add students either by hand or from the file.
  + "inputStudent" begins to print out the list of information that needs to be filled in.
  + "importStudent" has the users choose the file they want to import new students to classes and perform operations for the data to fit in the linked list.
  + "add1stYearStudents" uses the new student information and adds them to the new node of students in class linked list.
  + "Draw\_In\_Stud" prints out the boxes the user enters the student's information and checks for their validity.
* File: [Year]\_New\_Year\_Interface.cpp: stores functions with the purpose of getting back the file and having the user choose the year they want to create
  + "Show\_Year\_List" prints out the lists of existing years.
  + "Get\_CheckFormat\_Existed\_Year" first calls 'Show\_Year\_List" function and allows the user to input the school year.
  + "RecoverFile" is called to begin loading all files that saved in files and create the first linked list of years.
  + "Interface\_New\_Year" allows users to enter the year they want to create and call 'create\_new\_year' function to create the new year.

**Login.cpp**

* "LoggingIn" prints out the boxes for users to enter the username and password and checks through the account linked list to verify it to allow users to access in the program.
* "ChangePass" makes users re-enter their password to change into the new one and save it into the data bank.
* "Profiling" displays the user's profile including username, password, role, full name, birthday, gender, and social ID.
* "AccountAlteration" prints out the menu of managing accounts and calls the corresponding functions to do the tasks.
* "LoggingMain" stores the two functions to minimize the calling function in the other functions to login and manage accounts.

**Main.cpp**

* "EradicateLL" is managed to call all functions that delete the linked list and free up the memory.
* "Merge\_year\_sem" merges all the tasks including recover back the files, reading the accounts and years, linking the information around the linked lists, and calling the main menu that fits the user's role.