CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

SOFT 6017 PROJECT

- Due: 26th April @ 23:59
- Submit your code & evidence of using **Git** through **Canvas**
- Oral **assessment** & **demonstration** may take place in week 12
- Worth: **25%** SOFT6017's overall mark
- Code will be tested for **plagiarism** (refer http://www.cit.ie/aboutcit/reports_plansandpolicies/academic for CIT policies)

Module Attendance Records

The Module Attendance Records programme will allow a lecturer do the following tasks for any module that they teach.

- record the attendance of each student for a class session (as present/absent/excused)
- get statistical information based on the attendance data stored

The lecturer will logon to the system using a username and password. Once the lecturer has successfully logged on they will choose which task they wish to undertake. Following this they will then be shown a list of modules that they teach. They will then select a module from this list.

If the lecturer has chosen to record attendance they will be given an opportunity to mark each student enrolled as either present, absent or excused. The system will then update the relevant file with this information.

If the lecturer chooses to generate statistics the generated data will be displayed on the screen and stored to a file. The name of the file will use the module code and the current date. The following information is displayed and written to the statistics file:

- the code for the module
- the total number of students enrolled
- the total number of classes to date
- the average attendance rate
- the names of any non-attender(s)
- The names of student(s) with an attendance rate of under 70%
- the names of the best attender(s)

The following files will be required by the program to store login information for one lecturer, the module codes and module names for that lecturer along with the attendance information for each module.

1. Login_data.txt

This stores the username and password for the lecturer that can use the system.
Here is a sample file for a lecturer named ‘Anna’ with a password ‘12345’. The password may be stored in clear text.

```
Anna
12345
```

2. Modules.txt

This holds the details of the modules taught by one lecturer. Each line in the file has a module code and module name. A snippet of this file might be:

3. Module Specific Files

Each module that is listed in modules.txt will have an associated file (e.g. SOFT6017.txt). This holds the attendance records for a module. Each line in the file has a student's name, number of days present, number of absences, number of days excused. A snippet of one such file might be:

```
Mary Martin,10,0,0
Alan Wilson,0,9,1
Alan Lowe,5,6,0
```

Details of Screen Shots

1. Login Screens

Asks the user to login and verify the login details. On successful login the user can view the main menu as shown.

Sample Run for Successful Login and Exiting Immediately.

```
Name:Anna
Password:12345

Welcome Anna


Module Record System - Options
-------------------------------
1. Record Attendance
2. Generate Statistics
3. Exit
>3
```

On entering a wrong user or password details, display a login failure message.

Sample Run for Failed Login

```
Name: Anna
Password: 1
Module Record System – Login Failed
```

2. Main Menu Screen

On successful login display the list of operations the user can choose from. These options are recording attendance, generating statistics or exiting.

Sample Run for Main Menu

```
Name: Anna
Password: 12345
Welcome Helen
Module Record System - Options
------------------------------
```

```
1. Record Attendance
2. Generate Statistics
3. Exit
>
```

3. Generate Statistics Screen

If the user chooses option 2. *Generate Statistics*, then the user is provided with a list of the modules that they teach. The list of modules is taken from *modules.txt*. The program will then read the relevant module file (i.e. *SOFT_6017.txt*) and generate the statistics for this module. The statistics should be displayed on the screen but should also be stored to a file. Note the file name uses the module name and the current date.

Sample Run for Generating Statistics

```
Name: Anna
Password: 12345
Welcome Helen
Module Record System - Options
------------------------------
1. Record Attendance
2. Generate Statistics
3. Exit
>
2
Module Record System(Statistics) - Choose a Module
------------------------------------------------
1. SOFT_6017
2. COMP_1234
>1
Module: SOFT_6017
Number of students: 3
Number of Classes: 10
Average Attendance: 5.0 days
Low Attender(s): under 70.0 %
       Alan Lowe
Non Attender(s):
       Alan Wilson
Best Attender(s):
       Attended 10/10 days
       Mary Martin
```

4. Record Attendance Screen

If the user chooses option *1. Record Attendance* they are then prompted to select a module. The attendance data for that module must be loaded from the relevant file (ie. SOFT_6017.txt).
The user is prompted then to record the attendance status (present/absent/excused) for each student taking this module. The updated attendance data is then written back to the same file, overwriting the original content of the file.

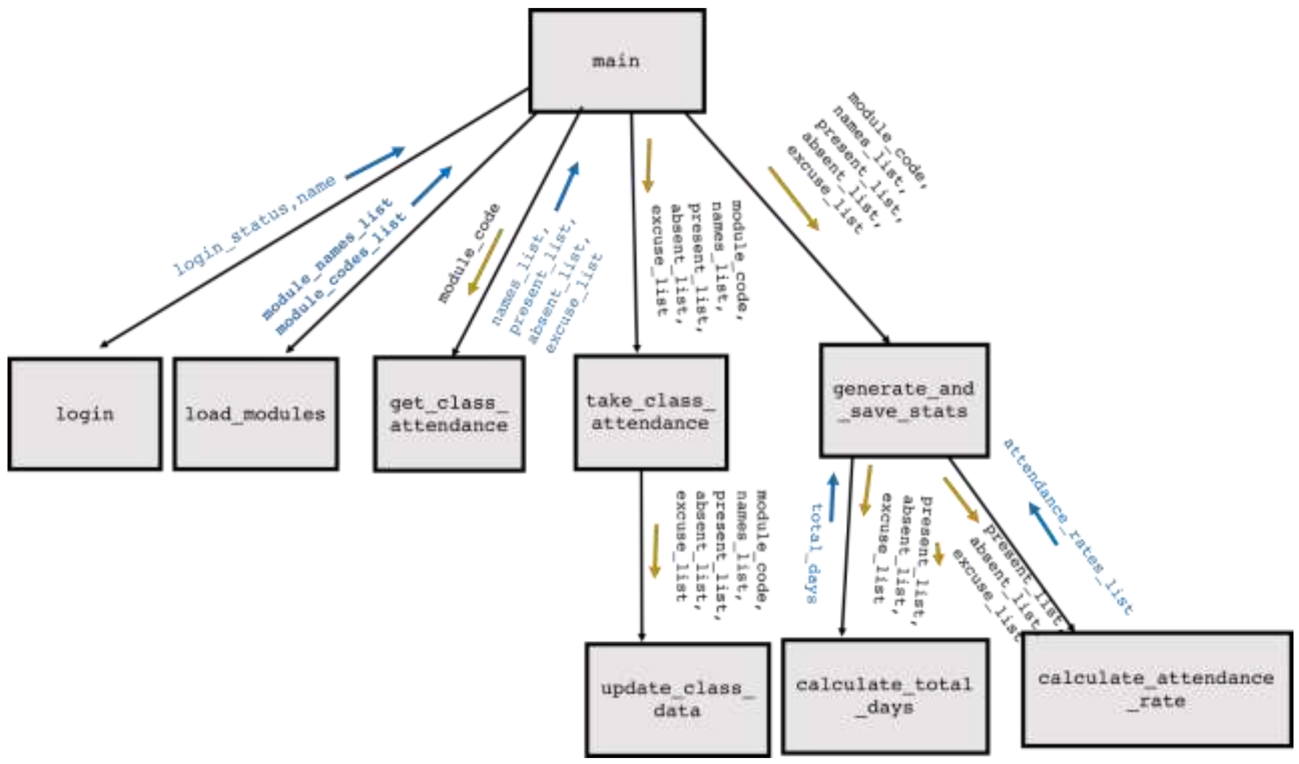
Sample Run for Record Attendance

```
Name: Anna
Password: 12345
Welcome Helen
Module Record System - Options
------------------------------
```

```
1. Record Attendance
2. Generate Statistics
3. Exit
> 1
Module Record System(Attendance) - Choose a Module
-------------------------------------------------
1. SOFT_6017
2. COMP_1234
>1
Module Record System(Attendance) SOFT_6017
--------------------------------------
There are 3 students enrolled.
Student #1: Mary Martin
1. Present
2. Absent
3. Excused
>2
Student #2: Alan Wilson
1. Present
2. Absent
3. Excused
>3
Student #3: Alan Lowe
1. Present
2. Absent
3. Excused
>1
SOFT_6017.txt updated with latest attendance records
```

Notes:
The total number of students listed is given at the start.
The students are numbered as they are listed .
The file `SOFT_6017.txt` is overwritten with the new attendance numbers.

Structure


main
login_status, name
module_names_list, module_codes_list
module_code
names_list, present_list, absent_list, excuse_list
module_code, names_list, present_list, absent_list, excuse_list
module_code, names_list, present_list, absent_list, excuse_list
login
load_modules
get_class_attendance
take_class_attendance
generate_and_save_stats
module_code, names_list, present_list, absent_list, excuse_list
total_days
absent_list, excuse_list
present_list, absent_list, excuse_list
attendance_rates_list
update_class_data
calculate_total_days
calculate_attendance_rate

Advice

Coding

When using lists you may make use of the `max`(), `index`() and `sum`() functions for this application.

Version Control System (VCS)

As part of programming a large application it is important to use a version control system to keep track of the changes you make to a program. VCS allow you to revert to an old version if you need to do that. We have chosen **Git**, which is an industry standard and is in-built to PyCharm.
Each time you add functionality, commit that change to Git. At the end of the project, take a **screenshot** of the git log commands – this will list all the changes in full and brief format, as evidence of how you applied version control. Marking Scheme

Quality of code	• naming conventions • comments • use of whitespace • use and re-use of functions/modules	10%
Version Control	• implementation of git for project	10%
Functionality	• functionality	80%