De La Salle University
College of Computer Studies
Department of Software Technology

Programming with Structured Data Types (CCPROG2)
Machine Project Specification
3rd Trimester, SY 2022 - 2023

Project Name : Room Reservation System

Project Rationale : A system for managing advanced room bookings is called a room reservation system. Users can rent rooms that will be utilized for events like symposiums, workshops, meetings, thesis presentations, and similar activities.

For this project, you [and a classmate] will develop a room reservation system for the College of Computer Studies.

Project Premises :

1. Rooms that can be reserved will be based on the available rooms that are entered into the system via the admin module.

   a. In the admin module, room name, type & capacity are encoded.
   b. The types of rooms are as follows: classroom, seminar room, Auditorium, and training room,
   c. The room capacity may vary based on the room and location.
   d. Room names follow a pattern of XYYYY, where X refers to an alphabetic character followed by 4 digits. If the room is 607 and is located in Andrew, the room name will be A0607..

2. Allowed timeslots are the following: (Take note that these are fixed)

Timeslots for MTHF	Timeslots for WS
09:15 – 10:45 11:00 – 12:30 12:45 – 14:15 14:30 – 16:00 16:15 – 17:45 18:00 – 19:00	09:00 – 12:00 13:00 – 16:00 16:15 – 19:15

Mechanics :

1. The system will be accessed and utilized by the department secretary.
2. Those who would like to reserve a room will need to fill out a form that will be submitted to the secretary.

   The data that needs to be filled out are as follows:

- ID Number
- Full Name (First and Last Name)
- Year & Program
- Date and Time for Reservation
- Number of Participants
- Room to Reserve (preferred)
- Description of Activity

3. In case the activity will use more than 1.5 hours on specific days, they can reserve an additional timeslot (i.e., must be consecutive timeslot)
4. A person can reserve a maximum of 3 room reservations. Beyond this, s/he can reserve once an activity is completed (i.e., reservation has been marked as completed). Marking of an activity as completed will be done on the admin module.
5. A person could make changes to the details of the room reservation if other rooms requested are still free.
6. Moreover, a person can also cancel a room reservation if the cancellation is done before the actual date of the reservation.
7. The program must be able to detect the following scenarios:
    a. Conflict with date, time, and timeslot for reservation
    b. Maximum reservations per reserver
    c. Maximum capacity for each room to be used.
8. When the program is terminated, all reservation transactions must be saved into a text file. Moreover, when the program starts, reservations from the text file must be loaded to memory.

It is expected that the program you will be developing will require the use of the following concepts:

- All programming concepts discussed in CCPROG1
- Arrays or Pointers
- Structures
- String Manipulation
- File Structures*

Bonus / Extra Credits: A maximum of 10 points may be given for features over & above the requirements, such as: (1) Making the room search more flexible (e.g., searching for rooms that have a specific capacity, searching for all rooms that are vacant; (2) Report generation for all vacant and reserved rooms for a specific date (i.e., generating a text file report (e.g., the report includes a header / title, report details are in column form.)

Milestones: There will be features in your project that needs to be checked on certain dates. This is to show your progress regarding your project. The following dates will be set for assessment / progress reports:

June 16, 2023 (F): You will need to record a video of what you have already completed as part of Milestone #1 which covers the following: basic screens for user inputs and menus, creation of the structures and arrays that will be used.

July 1, 2023 (S): Submit a video recording of your progress in the following items for Milestone #2, namely: Adding, Editing, Viewing and Deleting of data entries, Data Validations, and Displaying of Records.

Submission & Demo: Final MP Deadline: July 29, 2023 (S), 1200 noon via Canvas. No projects will be accepted once the submission link is locked.

It should be noted that during the MP demo, it is expected that the program can be compiled successfully and will run. If the program does not run, the grade for the project is automatically 0. However, a running program with complete features may not necessarily get full credit, as implementation (i.e., code) and other submissions (e.g., non-violation of restrictions evident in code, test script, and internal documentation) will still be checked.

Requirements: The following files will be required when you submit your projects.

1. Complete Program (Source Code, .c file)

Make sure that your implementation has considerable and proper use of arrays, strings, structures, files, and user-defined functions, as appropriate, even if it is not strictly indicated.

Internal documentation / comments must also be present in your code. It should describe variables, functions, and sections of the program.

It is expected that each feature is supported by at least one function that you implemented. Some of the functions may be reused (meaning you can just call functions you already implemented [in support] for other features. There can be more than one function to perform tasks in a required feature.

Debugging and testing was performed exhaustively. The program submitted has:

1. NO syntax errors.
2. NO warnings - make sure to activate -Wall (show all warnings compiler option) and that C99 standard is used in the codes.
3. NO logical errors -- based on the test cases that the program was subjected to.

2. Function Specifications (included in the source code, should be placed below the main program)

This document includes all the user-defined functions. The function's name, description, input parameter/s, and the return values should be written in the document. A document template is provided in the next page.

```
/* funcA returns the number of capital letters that are changed to small letters
@param strWord - string containing only 1 word
@param pCount - the address where the number of modifications from
capital to small are placed
@return 1 if there is at least 1 modification and returns 0 if no modifications
Pre-condition: strWord only contains letters in the alphabet
*/
```

3. Test Script (Test Cases encoded on a spreadsheet program and saved as a pdf file)

This document will contain the test cases for functions that feature computation or processing of data. Functions which are only used for screen designs may not be included in this document. Script document templates are provided in the next page.

Additional Requirements:

Honesty Clause / Declaration of Academic Honesty

In the first part of your source code, you will need to include a Code of Honesty clause indicating that you did not plagiarize your project. A sample of the clause is provided below.

```
/*************************************************************************
     This is to certify that this project is my own & partners work, based on
     my/our personal efforts in studying and applying the concepts learned. I/we
     have constructed the functions and their respective algorithms and
     corresponding code by me (and my partner). The program was run, tested, and
     debugged by my own efforts. I further certify that I have not copied in
     part or whole or otherwise plagiarized the work of other students and/or
     persons.

     <Your Full Name> - <ID Number> - <Section>
     <Your Full Name> - <ID Number> - <Section>
 *************************************************************************/
```

User Acceptance Test

The Test Script should be in a table format. There should be 3 kinds of test cases (each described by the 'Description' column) per function. There is no need to test functions which are only for screen design (i.e., no computations/processing; just printf). The last column, 'P / F', states whether the program's actual output matches the test cases' expected output. Sample is shown below.

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
sortIncreasing	1	Integers in array is in increasing order already	aData contains: 1 37 8 10 15 32 33 37 53	aData contains: 1 3 7 8 10 15 32 33 37 53	aData contains: 1 3 7 8 10 15 32 33 37 53	P
	2	Integers in array are in decreasing order	aData contains: 53 37 33 32 15 10 8 7 3 1	aData contains: 1 3 7 8 10 15 32 33 37 53	aData contains: 1 3 7 8 10 15 32 33 37 53	P
	3	Integers in array is combination of positive and negative numbers and in no particular sequence	aData contains: 57 30 -4 6 -5 -33 -96 0 82 -1	aData contains: - 96 -33 -5 -4 -1 0 6 30 57 82	aData contains: 96 -33 -5 -4 1 0 6 30 57 82	P

Other Important Instructions:

1. Use **`gcc -Wall`** or DevC++ with compiler option -Wall to compile your C program. Make sure you test your program completely (compiling & running).

2. **Do not use brute force**. Use conditional statements, loops, and functions whenever/wherever appropriate.

   You may use topics outside the scope of CCPROG2 but this will be self-study. However, do expect to be questioned on these constructs. **Goto** label, **exit()**, **break** (except in switch), **continue**, **global variables**, calling **main()** are not allowed.

3. Include internal documentation (comments) in your program.

4. Upload the softcopies via Submit Assignment in Canvas. You can submit multiple times prior to the deadline. However, only the last submission will be checked. Send also to your DLSU Google account a copy of all deliverables to serve as a backup for yourself.

5. Filename conventions are as follows:

Deliverable	File Name Convention	Example
.c Program	room_lastname1_lastname2.c	Room_Lumagui_Ang.c

.txt file	Use the default name records.txt	records.txt
Function files	Use the name "helper"	helper.c and/or helper.h

6. During the demonstration, the students are expected to appear on time, to answer questions in relation to the output and the implementation (source code), and/or to revise the program based on a given demo problem. Failure to meet these requirements could result in a grade of 0 for the project.

7. It should be noted that during the MP demo, it is expected that the program can be compiled successfully and will run. If the program does not run, the grade for the project is automatically 0. However, a running program with complete features may not necessarily get full credit, as implementation (i.e., code) and other submissions (e.g., function specs, test script, and internal documentation) will still be checked.

8. This project can be done individually or by pair. Any form of cheating (working in collaboration, asking other people's help, copying any part of other's work, etc.) can be punishable by a grade of 0.0 for the course & a disciplinary case. Thus, do not risk it; the consequences are not worth the risk and not worth the pesky reminder by your conscience.

9. Compiler warnings will merit deductions.

10. Any requirement not fully implemented, or instruction not followed will merit deductions.