BACHELOR OF COMPUTER SCIENCE
SCHOOL OF COMPUTER SCIENCE
BINA NUSANTARA UNIVERSITY
JAKARTA

ASSESSMENT FORM

Course: COMP6047001 - Algorithm and Programming

Method of Assessment: Case Study

Semester/Academic Year : 1/2024-2025

Name of Lecturer : **……**

Date : **……**

Class : **……**

Topic : **Material Review II**

Group Members :	1_____

Student Outcomes:
(SO 2) Mampu merancang solusi aplikasi piranti lunak berdasarkan analisis permasalahan yang dapat diselesaikan dengan pendekatan terstruktur dalam bidang informatika;
***Able to design software application solutions based on problems analysis which can be solved with structured approach in informatics area*;**

Learning Objectives:
(LObj 2.2) Mampu mengimplementasikan solusi berbasis komputasi untuk memenuhi serangkaian persyaratan komputasi tertentu dalam konteks ilmu komputer

Able to implement a computing-based solution to meet a given set of computing requirements in the context of computer science.

Learning Outcomes:
LO-2 : apply syntax and functions in C language in problem solving
LO-3 : construct a program using C language in problem solving

No	Related LO – Lobj - SO	Assessment criteria	Weight	Excellent (85 - 100)	Good (75-84)	Average (65-74)	Poor (0 - 64)	Score	(Score x Weight)
1	LO 2 – LObj 2.2 – SO 2	Ability to apply C syntax for problem solving	**50%**	85% - 100% of C syntax is correctly applied	75% - 84% of C syntax is correctly applied	65% - 74% of C syntax is correctly applied	0% - 64% of C syntax is correctly applied		
2	LO 3 – LObj 2.2 – SO 2	Ability to construct the algorithm into C program	**50%**	The C program is built 85% - 100% correctly	The C program is built 75% - 84% correctly	The C program is built 65-74% correctly	The C program is built 0-64% correctly		
	Total Score: ∑(Score x Weight)								

Remarks:

ASSESSMENT METHOD

Instructions

1. This case study is individual, with duration of 1 week.
2. This case study consists of 2 questions.
3. The first question is a single case study problem, while the second question is a narrative case study to **create 4 functions** with their respective goals. Your program should run correctly to get full score.
4. The second question should be combined into a single .c file. Give comments to explain which function does what.
5. All answers should be included into a single .zip file and submitted to Binusmaya.
6. Example only serves as an example. You may create the command line program as creative as you can.

Note for Lecturers:

1. This case study assignment will be held with duration of 1 week in review topic, or week 13.
2. The answer is manually checked by each lecturer (not by system).
3. The example only serves as an example to help students understand the assignment. As long as the function works as intended and the program run correctly, you may give the full score for each problem.

1. Case study 1 **(LO 2 – L.Obj 2.2 – SO 2, 30%)**:

   Using string manipulation, iteration, and selection, create a C program to handle string conversion. The string conversion accept a string input with constraint:

$$1 \leq |S| \leq 100$$

$$s \in S$$

$$s = \{UpperCaseCharacters \cup LowerCaseCharacters\}$$

   The conversion needed **is string reversal, followed by inverse capitalize character at each position**. Inverse capitalize means that lowercase will convert to uppercase, and uppercase will convert to lowercase. Example: if given string **SuniBVerse,** the string will reverse to **esreVBinuS**. After inversion, the program will convert each character with the rules above. Therefore, the final string will become: **ESREvbINUs**.

2. Case study 2 **(LO 3 – L.Obj 2.2 – SO 2, 70%)** :

   Download the file from link here: **https://1drv.ms/u/s!AhuAx03LAKWtnOM9O1wlXSAR84Z67g?e=lVmH5x**

   The file itself is a .csv file containing multiple rows and columns of data. Your task is to build several functions as utility for the data itself, **therefore you should implement function to read the .csv file into your program first.** The functions needed as follows:

   a. Display (15%)

      This function needs 1 variable: **number of rows to be displayed.** This function will display data with n rows**.** ***n* must be a positive integer number.** If n > total number of rows, display all data. Example:

```
What do you want to do?
1. Display data
2. Search Data
3. Sort Data
4. Export Data
5. Exit
Your choice: 1
Number of rows: 5

Location    City              Price       Rooms Bathroom    Carpark     Type        Furnish
Mont-Kiara  Kuala-Lumpur      1000000     2     2           0           Built-up    Partly
Cheras      Kuala-Lumpur      310000      3     2           0           Built-up    Partly
Kepong      Kuala-Lumpur      358000      3     3           0           Built-up    Partly
Taman-Desa  Kuala-Lumpur      455000      2     2           0           Built-up    Partly
Kepong      Kuala-Lumpur      358000      3     3           0           Built-up    Partly
```

   b. SelectRow (20%)

This function needs 2 variables as input: **column** and **query value**. This function will display rows that have the **exact value** with the query. If data is not found, print **Not Found**. If data is found, print data details as depicted in example below. If multiple data exist, display all data that matched the query. Example:

```
What do you want to do?
1. Display data
2. Search Data
3. Sort Data
4. Export Data
5. Exit
Your choice: 2
Choose column: Location
What data do you want to find? Jakarta
Data not found!
```

```
What do you want to do?
1. Display data
2. Search Data
3. Sort Data
4. Export Data
5. Exit
Your choice: 2
Choose column: Location
What data do you want to find? Jinjang
Data found. Detail of data:
Location    City              Price       Rooms Bathroom    Carpark     Type        Furnish
Jinjang     Kuala-Lumpur      56000       3     2           0           Built-up    Partly
Jinjang     Kuala-Lumpur      72000       5     2           0           Land-area   Unfurnished
Jinjang     Kuala-Lumpur      1200000     5     4           0           Bult-up     Partly
```

c. SortBy (20%)

This function needs 2 variables as input: **column** and **ascending or descending**. After data was sorted, display the first 10 data. Example (this example only showed 5 data to simplify):

```
What do you want to do?
1. Display data
2. Search Data
3. Sort Data
4. Export Data
5. Exit
Your choice: 3
Choose column: Rooms
Sort ascending or descending? asc
Data found. Detail of data:
Location          City              Price       Rooms Bathroom    Carpark     Type        Furnish
KLCC              Kuala-Lumpur      1450000     1     1           0           Built-up    Fully
KLCC              Kuala-Lumpur      2506500     1     1           0           Built-up    Partly
Damansara-Heights Kuala-Lumpur      1109760     1     1           0           Built-up    Fully
Bangsar           Kuala-Lumpur      1300000     1     1           0           Built-up    Fully
City-Centre       Kuala-Lumpur      1420000     1     1           0           Built-up    Fully
```

d. Export (15%)

This function needs 1 variable as input: **filename (string)**. This function will **write the data into a .csv file or comma separated value** (,) **with user specified filename in the same directory as your program**. Example:

```
What do you want to do?
1. Display data
2. Search Data
3. Sort Data
4. Export Data
5. Exit
Your choice: 4
File name: sorted_data
Data successfully written to file sorted_data.csv!
```