CCP6114 Programming Fundamentals Trimester 2430
Assignment Programming

Submission Dates
Assignment milestone 1: 25 Dec 2024, Wed, 5:00pm, Week 8
Assignment milestone 2: 22 Jan 2024, Wed, 5:00pm, Week 12

Group Assignment
This is an assignment for a group of three to four students or four students.
You may get permission from your respective tutor if your number of group members is less than four as early as possible.
No free rider, no sleeping member and all members in the group are expected to work together towards this group assignment.
STRICTLY NO COPYING from other sources except codes given in this course.
If detected, all parties involved will get 0 marks.
The codes must be your own.

Implementations
In this assignment, you are required to implement a "Light Mariadb Interpreter" using standard C++.
The interpreter interprets the file input statements.

The requirements of the light interpreter are:
- Reading from a file, outputting to screen, writing to a file
- Create one database and view one database name
- Create one table and view one table name
- Table supports two data types i.e. INT, TEXT, maximum of ten columns
- Insert rows to the table
- View table in csv mode
- Update table rows and view table
- Delete table rows and view table
- Count and output number of rows in the table

Also document all your assignment tasks with the marking table that contain cover page, table of contents, page numbering, inputs, outputs, screenshots, explanations, and others.

The given requirements in the sample inputs and sample outputs below are the basic requirements of this assignment.
You may feel free to add more features and capabilities to make the interpreter robust and efficient.

Sample inputs and outputs
The program is expected to read a program file input, execute it, display to the screen, and store the results to a file.
Step by step executing the sample programs as shown below.

Sample 1
input file
filename: fileInput1.mdb

```
CREATE fileOutput1.txt;
DATABASES;

CREATE TABLE customer(
customer_id INT,
customer_name TEXT,
customer_city TEXT,
customer_state TEXT,
customer_country TEXT,
customer_phone TEXT,
customer_email TEXT
);
TABLES;

INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (1,'name1','city1','state1','country1','phone1','email1');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (2,'name2','city2','state2','country2','phone2','email2');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (3,'name3','city3','state3','country3','phone3','email3');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (4,'name4','city4','state4','country4','phone4','email4');

SELECT * FROM customer;
```

output file and screen output
filename: fileOutput1.txt

```
> CREATE fileOutput1.txt;
> DATABASES;
C:\mariadb\fileInput1.mdb
> CREATE TABLE customer(
customer_id INT,
customer_name TEXT,
customer_city TEXT,
customer_state TEXT,
customer_country TEXT,
customer_phone TEXT,
customer_email TEXT
);
> TABLES;
customer
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (1,'name1','city1','state1','country1','phone1','email1');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (2,'name2','city2','state2','country2','phone2','email2');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (3,'name3','city3','state3','country3','phone3','email3');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (4,'name4','city4','state4','country4','phone4','email4');
> SELECT * FROM customer;
customer_id,customer_name,customer_city,customer_state,customer_country,customer_phone,
customer_email
1,name1,city1,state1,country1,phone1,email1
2,name2,city2,state2,country2,phone2,email2
3,name3,city3,state3,country3,phone3,email3
4,name4,city4,state4,country4,phone4,email4
```

Sample 2
input file
filename: fileInput2.mdb

```
CREATE fileOutput2.txt;
DATABASES;

CREATE TABLE customer(
customer_id INT,
customer_name TEXT,
customer_city TEXT,
customer_state TEXT,
customer_country TEXT,
customer_phone TEXT,
customer_email TEXT
);

INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (1,'name1','city1','state1','country1','phone1','email1');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (2,'name2','city2','state2','country2','phone2','email2');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (3,'name3','city3','state3','country3','phone3','email3');
INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (4,'name4','city4','state4','country4','phone4','email4');
SELECT * FROM customer;

TABLES;

UPDATE customer SET customer_email='email333' WHERE customer_id=3;
SELECT * FROM customer;

DELETE FROM customer WHERE customer_id=4;
SELECT * FROM customer;

SELECT COUNT(*) FROM customer;
```

output file and screen output
filename: fileOutput2.txt

```
> CREATE fileOutput2.txt;
> DATABASES;
C:\mariadb\fileInput2.mdb
> CREATE TABLE customer(
customer_id INT,
customer_name TEXT,
customer_city TEXT,
customer_state TEXT,
customer_country TEXT,
customer_phone TEXT,
customer_email TEXT
);
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (1,'name1','city1','state1','country1','phone1','email1');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (2,'name2','city2','state2','country2','phone2','email2');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (3,'name3','city3','state3','country3','phone3','email3');
> INSERT INTO
customer(customer_id,customer_name,customer_city,customer_state,customer_country,custom
er_phone,customer_email) VALUES (4,'name4','city4','state4','country4','phone4','email4');
> SELECT * FROM customer;
customer_id,customer_name,customer_city,customer_state,customer_country,customer_phone,
customer_email
1,name1,city1,state1,country1,phone1,email1
2,name2,city2,state2,country2,phone2,email2
3,name3,city3,state3,country3,phone3,email3
```

```
4,name4,city4,state4,country4,phone4,email4
> TABLES;
customer
> UPDATE customer SET customer_email='email333' WHERE customer_id=3;
> SELECT * FROM customer;
customer_id,customer_name,customer_city,customer_state,customer_country,customer_phone,
customer_email
1,name1,city1,state1,country1,phone1,email1
2,name2,city2,state2,country2,phone2,email2
3,name3,city3,state3,country3,phone3,email333
4,name4,city4,state4,country4,phone4,email4
> DELETE FROM customer WHERE customer_id=4;
> SELECT * FROM customer;
customer_id,customer_name,customer_city,customer_state,customer_country,customer_phone,
customer_email
1,name1,city1,state1,country1,phone1,email1
2,name2,city2,state2,country2,phone2,email2
3,name3,city3,state3,country3,phone3,email333
> SELECT COUNT(*) FROM customer;
3
```

Sample 3
Your own sample?

Deliverables

a) Source code in only one cpp file and/or other header files with implementations. For example, TC3L_G04_main.cpp, File_Input.h, File_Output.h)
b) Design documents such as structure diagrams, flowcharts, pseudocodes in docx and pdf format to explain your work.
c) A detailed and well-organized document that includes your inputs, outputs, figures, screenshots, explanations, code segments, pseudocodes, pseudocode explanations, and others.
d) Assignment presentation and assignment interview with your respective tutor in person in MMU. Every member in the group requires to attend and present your group assignment with your respective tutor.

Additional Info on Deliverables

a) Source codes have to be properly formatted and documented with comments and operation contracts. Do not submit executable files.

b) For ALL your source code cpp files, insert the following information at the top of the file, for example:

```
// ******************************************************
// Program: YOUR_FILENAME.cpp
// Course: CCP6114 Programming Fundamentals
// Lecture Class: TC3L
// Tutorial Class: TT5L
// Trimester: 2430
// Member_1: ID | NAME | EMAIL | PHONE
// Member_2: ID | NAME | EMAIL | PHONE
// Member_3: ID | NAME | EMAIL | PHONE
// Member_4: ID | NAME | EMAIL | PHONE
// ******************************************************
// Task Distribution
// Member_1:
// Member_2:
// Member_3:
// Member_4:
// ******************************************************
```

Soft-copy submission instruction

a) Create a folder in the following format:
LECTURESECTION_GROUPNUM_A1orA2.zip

For example, if your name is Frank Carrano, you come from TC3L section, Group 4, and you are submitting Assignment Milestone 1, then your folder name should be "TC3L_G04_A1.zip" without the double quotes.

b) Place all files for submission into the folder.

c) Zip your folder to create a zip archive.
Remember that for Assignment Milestone 2, your zip archive filename should be, for example, "TC3L_G04_A2.zip".

d) Submit your assignment through online form determined by your respective tutor before the deadline.

No mark will be awarded if you did not attend and present your group assignment together.

No use or test using CodeBlocks and C++ standards will result in no mark.

Cheating in any form will not be tolerated and will result in no mark.

Assignment programming and documentation (30%)
You are required to submit assignment milestone 1 to your respective tutor also before the submission deadline.
Also document all your assignment tasks with this marking table that contain cover page, table of contents, page numbering, inputs, outputs, screenshots, explanations, and others.

Criteria	**Max**	**A1**	**A2**	**Mark**
Q1. Create database and view database name Create table, view table name Table supports two data types i.e. INT, TEXT Insert rows to the table View table in csv mode	5	*	*	?
Q2. Reading from a file, outputting to screen, writing to a file (0 if no files used or no screen outputs)	3		*	?
Q3. Update table rows and view table Delete table rows and view table	4		*	?
Q4. Count and output number of rows in the table	2		*	?
Q5. Must use vectors or arrays, functions or classes, to store file output contents	2		*	?
Q6. Inline comments, function or class comments, indentation, following proper C++ naming and styling conventions Any violation is penalized by a reduction of 1 mark.	2		*	?
Q7. The program demonstrates error handlings. [0: Below Expectation, 1: Within Expectation, 2: Exceed Expectation]	2		*	?
Q8. Correct structured diagrams	2		*	?
Q9. Correct flowcharts or pseudocodes with explanations for all the file input statements. Any missing flowchart or pseudocode will cause you to lose 1 mark.	2		*	?
Q10. Sample file inputs at least 3, their screen outputs, their file outputs with screenshots and explanations.	3		*	?
Q11. User documentation done and is coherence with the all	3		*	?

implementations. Any missing input statement will cause you to lose 1 mark.				
Total	30			?

Additional comments

You are required to fill in your task percentage and task descriptions.
Every student is responsible for 100% (task percentage) of this group assignment work.

Student 1

Student ID	?
Student name	?
Task percentage	?
Task descriptions	?
Total score (30m)	?

Student 2

Student ID	?
Student name	?
Task percentage	?
Task descriptions	?
Total score (30m)	?

Student 3

Student ID	?
Student name	?
Task percentage	?
Task descriptions	?
Total score (30m)	?

Student 4

Student ID	?
Student name	?
Task percentage	?
Task descriptions	?
Total score (30m)	?

Each feature will be evaluated based on documentation, fulfilment of requirements, correctness, compilation without warnings and errors, error free during runtime, error handlings, quality of comments, user friendliness, good coding format and style.