Introduction to Systems Programming (System I)
<u>Lab #8</u>
Max Points: 40
<u>Due: Wednesday, October 21, 2020 before 11:59 PM</u>
<u>Email-based help Cutoff: 5:00 PM on Wednesday, October 21, 2020</u>

You should save/rename this document using the naming convention MUid.docx (example: `vendomcg.docx`).

<u>**Objective**</u>: The objective of this exercise is to gain experience with programming a database application using C++ and number representations including:
1. Binary
2. Octal
3. Hexadecimal

Fill in answers to all of the questions. You may discuss the questions with your neighbor or instructor. **Yes, you may verify your solution with reference solution available on Canvas.**

Name: Karim Sammouri

Note: It is possible to get extra-credit. There are 45 possible points on this lab.

Part #1: Binary number representation
[15 points]

The first exercise is completed for you to provide an example (you can copy-paste for other problems in this exercise)

1. Convert 54_{10} to binary. **[2 points intermediate steps, 1 point final answer]**

Step	Operation	Result	Remainder
1	54/2	27	0
2	27/2	13	1
3	13/2	6	1
4	6/2	3	0
5	3/2	1	1
6	1/2	0	1

Successively dividing by 2 until quotient is less than 2 and writing remainders in reverse order, $54_{10} = 110110_2$

2. Convert 119_{10} to 8-bit binary. **[2 points intermediate steps, 1 point final answer]**

Step	Operation	Result	Remainder
1	119/2	59	1
2	59/2	29	1
3	29/2	14	1
4	14/2	7	0
5	7/2	3	1
6	3/2	1	1
7	1/2	0	1

$119_{10} = 01110111_2$

3. Convert 128_{10} to binary. **[2 points intermediate steps, 1 point final answer]**

Step	Operation	Result	Remainder
1	128/2	64	0
2	64/2	32	0
3	32/2	16	0
4	16/2	8	0
5	8/2	4	0
6	4/2	2	0
7	2/2	1	0
8	1/2	0	1

$128_{10} = 10000000_2$

4. Convert 10100_2 to decimal.   **[2 points intermediate steps, 1 point final answer]**

$10100_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (0 \times 2^0) = 20_{10}$

5. Convert 00001101_2 to decimal. **[2 points intermediate steps, 1 point final answer]**

$00001101_2 = (0 \times 2^7) + (0 \times 2^6) + (0 \times 2^5) + (0 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) = 13_{10}$

Part #2: Octal number representation
[6 points]

6. Convert 36_{10} to octal   **[2 points intermediate steps, 1 point final answer]**

Step	Operation	Result	Remainder
1	36/8	4	4
2	4/8	0	4

$36_{10} = 44_8$

7. Convert 110_8 to decimal.   **[2 points intermediate steps, 1 point final answer]**

$110_8 = (1 \times 8^2) + (1 \times 8^1) + (0 \times 8^0) = 72_{10}$

Part #3: Hexadecimal number representation
[12 points]

8. Convert $10F_{16}$ to decimal.   **[2 points intermediate steps, 1 point final answer]**

$10F_{16} = (1 \times 16^2) + (0 \times 16^1) + (F \times 16^0) = (1 \times 16^2) + (0 \times 16^1) + (15 \times 16^0) = 271_{10}$

9. Convert ABC_{16} to binary   **[2 points intermediate steps, 1 point final answer]**

$ABC_{16} = $ "1010" + "1011" + "1100" $= 101010111100_2$

10. Convert 0110100101010011_2 to Hex. **[2 points intermediate steps, 1 point final answer]**

$0110100101010011_2 = $ "0110" + "1001" + "0101" + "0011" = "6" + "9" + "5" + "3" $= 6953_{16}$

11. Convert 10011111_2 to Hex   **[2 points intermediate steps, 1 point final answer]**

$10011111_2 = $ "1001" + "1111" = "9" + "15" = "9" + "F" = $9F_{16}$

Part #4: Programming in C++ with SQL
[12 points]

Background: As we discussed in class, we can create a program just like the mysql command-line program that we used in the previous lab. In this lab, we will be making a light-weight interactive program to interface with a MySQL RDMS.

The program will need to do the following:
1. Take an argument from the command-line to specify interactive mode (-I) or to load (-L) a file into the database. **[1 point]**
    a. Interactive mode will operate like the mysql program where it reads from standard input to get queries from a user.
    b. Loading a file takes an additional argument of the file (so it will be: -L filename) to parse and load into the database.
2. The program will need to use the argument to determine its execution path. **[1 point]**
3. The program will need to connect to the database **(INFO BELOW)**. **[1 point]**
4. Interactive mode (-I): **[5 points]**
    a. Continue to run unless the user enters "quit" or reaches logical end-of-file
    b. Take a user input and make it a query
    c. Execute the query
    d. Access and format the results
    e. Display the results to the user **(EXAMPLE ON NEXT PAGE)**
5. Load a file (-L): **[4 points]**
    a. Parse the supplied file.
        i. Note: it can be safely assumed any file give will be structured the same, but values can be different (first column is table then columns formatted as attribute:value for some arbitrary number of columns).
    b. Generate a query to insert the file's contents into the database
    c. Execute the query **(EXAMPLE ON NEXT PAGE)**

NOTE: Assuming the prior sections are completed correctly, full credit for the lab will only require _one_ of the two options to be correctly implemented. However, they will both be required for Homework 4.

Database Information (same as Lab 7):
```
Server: localhost
User: cse278
Password: S3rul3z
Database Name: cse278
```

Example Query Result:

```
select * from Product;
-----Query Result-----
| 1 | Product1 | 50 | 1 | Canon |
| 2 | Product2 | 150 | 2 | GizmoWorks |
| 3 | Product3 | 150 | 2 | Hitachi |
------End Result------
```

Example File Load Result:

```
cvendome@Dagobah:solutions$./lab7 -L test_input.csv
Data Line 1 Loaded
Data Line 2 Loaded
Data Line 3 Loaded
```

Submit to Canvas (and *NOT* CODE)

Once you successfully completed the aforementioned exercises upload the following files to Canvas.

    i.   This MS-Word document (duly filled-in) saved as a PDF document.

    ii.   The C++ source file of the program developed as part of this lab.

Ensure you actually **submit** the files after uploading them to Canvas and the files are **NOT** empty/template files.