Project #10 (50 points): Structures

Submit Your Solution Using Angel by Noon, Friday April 5, 2013

(A late submission drop box will be available on 4/5/13 from Noon to 2pm)

<Project 10 Description>

In Project 10, you will write a computer program that adds, subtracts, and multiplies 2x2 matrices. The program should input two matrices, ***A*** and ***B***, from a user-specified file and store the components of each matrix into a structure variable. After reading a matrix, your program should echo print the matrix (***A*** or ***B)*** as shown in the sample solution. If reading of the matrix values is successful, the program computes ***A+B***, ***A-B***, ***A*B***, and ***B*A*** using the equations listed below. The results of these calculations are printed to the monitor as shown in the sample solution.

- In addition to your ***main*** function, you must write at least five other supporting functions.
- **All three math operations (+, -, *) are to be written as separate functions, and these functions are to return (by reference or as the function return type of a value returning function) the result of the operation. The math functions are not to print out the result. The returned result must be contained within a structure and it is printed out by a function call in main**
- You must use ***struct*** to create a data type to represent a matrix
- ***Global Variables are not allowed.*** If necessary, global constants may be used.
- You may only use material from Chapters 1 - 10 on this assignment.
- ***Arrays (Chapter 11) are not allowed on this assignment!!***

$$A=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad B=\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

$$A+B=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}+\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}=\begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \end{bmatrix}$$

$$A-B=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}-\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}=\begin{bmatrix} a_{11}-b_{11} & a_{12}-b_{12} \\ a_{21}-b_{21} & a_{22}-b_{22} \end{bmatrix}$$

$$A*B=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}*\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}=\begin{bmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{bmatrix}$$

Input is to follow the same order as that illustrated by the provided solution executable. Output is to exactly match that shown by the provided solution.

<Sample Solution for Project 10>

Run the provided sample solution to see what information needs to be output by the program. You may **RUN** the sample solution **Project_10_solution** by typing the following at a command prompt in a terminal window that has a current working directory of **~/CPE112_SPR13/Project_10 (where ~ represents your home directory)** and that the sample input files are present in that directory. **(Note, the input file must be in the directory from which the command below is run)**

/home/work/cpe112/Executables/Project_10/Project_10_solution

Provided sample input files: P10_in1.txt through P10_in6.txt are zipped in the file P10_in.zip

<Project 10 Directions>

Open a terminal window and change into your **CPE112_SPR13/Project_10** directory. Use a text editor (i.e. gedit) to open the existing **Project_10.cpp** file (this is the header file created in Project 2) and add code to it to complete this project. Once you have finished with the program and it compiles without syntax errors, run the executable and verify that the output for your program matches the output from the provided solution executable – **Project_10_solution**.

Once you are satisfied with your solution, submit Project_10.cpp via Angel.

NOTE: make sure that you do not change the order in which the information is entered. An automatic script is used to process all lab submissions, and if the order of the input information is modified, the script will not work properly with your program.

➔ Project 10 Requirements ←

- **You must use structures in your program as indicated in the description**

- This project is to introduce the concept of structures and how to use structures with functions. Therefore, **your solution is to contain at least 5 functions (void or value returning),** and some of these functions must use a structure parameter and/or return a structure as its function value.
- **At least one function is to be a value returning function that returns a structure variable as its function value**
- **At least one function is to have a reference parameter that is a structure**
- **Possible functions to use are: open an input file, load a matrix, print a matrix, add two matrices, subtract two matrices and multiply two matrices.**
- **The program is to have a function that prints out a single matrix. This function is to have a string parameter that contains the heading to print out above the matrix printed out. See the sample solution.**
- **Functions can have at most 3 parameters.** This requirement forces you to use structures as parameters.
- **All Functions are called from within main** – they cannot be called from other functions.
- The matrix values in the input file are to be read into structure variables. When reading the values, the program is to detect problems with reading in values for the first matrix and for the second matrix. Run the sample solution with input files 2 through 6 to see examples.
- **Read the matrices one at a time. After reading one matrix, print it out, then read the second matrix and print it out. Then calculate each result and output it as shown**
- A function to open the input file is required. The function should continue to prompt the user for a valid input file name until a valid name has been entered or ctrl-c is typed to exit the program. This task requires a loop. If an open fails, the input file stream variable needs to

be reset before being used again. Use the following statement to reset input file stream variable.

```
yourInputFileStreamVariable.clear();
```

<Project 10 Input File Information>

The input file for this project will contain data for two arrays. This information is stored in the file as shown below:

```
1    2
3    4

5    6
7    8
```

The input file can have three possible errors/problems. There can be a missing value from one of the two matrices, there can be a non-digit character in Matrix A and there can be a non-digit in Matrix B. If a value is missing from Matrix A, the error will not show up until Matrix B is read. Run the sample solution with all 6 provided input files to see the results of a valid input and various versions of the three possible errors mentioned.

< Project 10 Hints >

- Reading of the information from the file can be made directly into the member of the appropriate structure. For example: **`inFile >> structVar.a11;`**
- Use a loop in the print function to print out the dashes above and below the phrase passed in. Use the length of the phrase as the limit on the loop. Can also use the string(int,char) operation to obtain the same result.
- Test the status of the input file stream after a matrix has been read. The status will indicate if the read was successful or unsuccessful.
- **Using a structure declaration, create a structured data type to represent a matrix. This structure has 4 integer members – the four values of a matrix**
- To perform the matrix operations specified, program in the calculations provided in this handout. Make each matrix operation a function.
- Program needs three variables declared as the matrix structure data type. One variable holds the values of the A matrix, one variable holds the values of the B matrix and one variable holds the values of the result matrix.
- The first value of each row of a matrix is output left justified in a field width of 6. The second value of each row of a matrix is output immediately after the first value (no setw for the second value)
- An empty input file will be treated like an input file with information missing

<Project 10 C++ Concepts Explained>

The following C++ concepts are included or introduced in this project:

struct Defines a record data type in C++. Each record can contain multiple named fields accessible by name. The syntax template appears below.

StructDeclaration syntax template

```
struct  StructDataTypeName
{
   MemberList
};
```

where

MemberList syntax template

```
DataType     MemberName;
DataType     MemberName;
```

Remember, the structure type declaration describes the new data type – it does not reserve memory for storing values. You must declare a variable of that type in order to store values into the various fields.

```
StructDataTypeName          StructVariable;

int  X;  // Note the similarity of the above variable declaration to this one
```

Member Selection Operator

The member selection operator **.** is used to access a particular member (or field) of a struct variable. The syntax template appears below.

```
StructVariable.Membername
```

Allowed Aggregate Operations on Structs

Aggregate Operation	Allowed on Structs?	
Input/Output	No	Perform on a member by member basis
Arithmetic	No	
Comparison	No	
Assignment	Yes	
Argument Passage	Yes, by value or by reference	
Return as a function's return value	Yes	