A290/A590
Tools for Computing/Topics in Programming
System Programming with C and Unix.

Homework ASSIGNMENT 2

DUE Thursday, September 25 2025 by 11:59.00.00PM (SHARP)
Submit your file to your Canvas "Homework ASSIGNMENT 2" assignment in the "Individual Homework ASSIGNMENTS" group.

Preliminary Information:

You will be creating and submitting **one (1) file** for this Assignment: **`A2.c`**.

IMPORTANT REMINDER: As with PROGRAMS 1 and 2 and ASSIGNMENT 1, You are not permitted to use any libraries, or any other programmatic strategies or structures beyond those shared in our class sessions preparing you for PROGRAM 2. That is not a typo. Nothing I share in Meeting 7, or Meeting 8 can be used on ASSIGNMENT 2. Only those things in Meeting 4, Meeting 5, and (perhaps) Meeting 6 directly related to PROGRAM 2 can be added to your possibilities when it comes time to write your code. This will continue to be true for all future PROGRAMS and ASSIGNMENTS, as previously announced.

ASSIGNMENT 2 Program Descriptions:

There are 3 primary parts to this assignment, but you need to create **functions** for each and use your "main" to call each function. To repeat, you **must** create 3 separate functions, one for each task, and then call them, in the proper order, in your main(). Read carefully to be sure everything happens in the correct order and that each function does everything it is expected/required to do.

1. Arrays. [15 points]

Write an appropriately named function that, taking user input, creates two arrays of type `int` of the same size and then prompts further for the values for every element or cell of both arrays.

Display as output the intersection of the two arrays (all elements they each have in common, i.e., with the same value), and the union of the two arrays (all the values they have combined with no duplications).

Sample Output:

```
Enter the size of the arrays:
5

Enter elements of the first array:
7 8 9 5 4

Enter elements of the second array:
8 9 2 3 6

The intersection of two arrays is: 89
The union of two arrays is: 78954236
```

[**NOTE**: No "common" values are repeated. 8899 and 7889923654 would not be acceptable. Order of the specific digits in the union and intersection is not important.]

Make sure you understand why the output for intersection and union are what they are.

Matrices [2-D arrays]. [25 points]

A matrix M with i rows and j columns can be ***transposed*** into a matrix N having j rows and i columns by simply setting the value of $N_{a,b}$ equal to the value of $M_{b,a}$ for all relevant values of *a* and *b*.

Write an appropriately named function that takes **as an argument** a 4 x 5 matrix and a 5 x 4 matrix. Have the function transpose the 4 x 5 matrix and store the results in the 5 x 4 matrix. Make sure you call this function in your `main()` to test it, **without the need for user input**.

"Takes as an argument" has a specific and important meaning here. Your function code will look something like `functionname(arg1, arg2, arg3 …)`. Arguments can be almost anything, including specific values, arrays, etc. To call the function "without the need for user input" means you will have to declare `arg1, arg2, arg3, etc.` in `main()` and then call the function with those arguments as input.

It is up to you to decide the contents of the original 4 x 5 matrix, but your output should mimic the following:

Sample Output:

```
The original matrix was:
1 2 3 4 5
6 7 8 9 10
10 9 8 7 6
5 4 3 2 1

The transposed matrix is:
1 6 10 5
2 7 9 4
3 8 8 3
4 9 7 2
5 10 6 1
```

2. Calculator [30 points]

You are writing a simple calculator, but it has some unique features. Once started, the user will enter information into the Calculator in the form:

number operator

The Calculator must first be given an initial value which with be stored in the "Accumulator" with the special operator "S". The program ends gracefully with the special operator "E".

So your Calculator will use 6 operators: +, -, *, /, S, and E.

This is a Calculator, so it needs to be as precise in its calculations as you can make it.

Once the initial value is set, the "number operator" input expression will apply the operator to the number and the value that is stored in the "Accumulator" memory location. The result will be written in that same Accumulator and displayed in a meaningful and appropriate way. **You need to check for "divide by 0" and provide an appropriate "error message" without exiting the Calculator. You also need to check for unknown operators (anything other than the 6 listed above).** You are not responsible for addressing other possible errors by the user, but you are free to do so if you wish. There will be no penalty if you do not and no special consideration if you do.

[sample output for Calculator begins on the next page.]

Sample Output: [this is for the Calculator function only]

```
[jwhitmer@silo.luddy.indiana.edu] mycalc
Begin Calculations

Initialize your Accumulator with data of the form
 "number" "S" which sets the Accumulator to the value of your number
123123 S
Value in the Accumulator = 123123.000000
234 +
Value in the Accumulator = 123357.000000
123.5234523432 *
Value in the Accumulator = 15237482.510700
123456 /
Value in the Accumulator = 123.424398
1123123213 -
Value in the Accumulator = -1123123089.575602
0 /
Can not divide by 0.
Value in the Accumulator = -1123123089.575602
123123 k
Unknown operator.
Value in the Accumulator = -1123123089.575602
3 E
Value in the Accumulator = -1123123089.575602
End of Calculations.
```

ASSIGNMENT 2 General Program Requirements:

Write your `main()` or "calling function" so the 3 functions above run in the order they are presented. The "Array" function should run first, the "Matrixes" function should run second, and the "Calculator" function should run last.

Assignment 2 **must** include the following features:

1. Proper Heading Comment Block at the beginning of the file.
2. Proper, appropriate, and thorough "in-line" or "in the code" comments.
3. Input and output as indicated in each function example.
4. Properly written `main()` or "calling function" to "call" and test all functions in the order indicated above.
5. Your program compiles for testing.

Again, BE SURE you thoroughly comment your actual code and include the expected Heading Comment Block shown in Meeting Guide 2 and after as well as in other in class examples. These comments and comment block will represent 20% of your score for this Assignment. Use appropriate "white space" to insure both your code and comments are easy to read and follow. **Since you are submitting only one file, it is fair for us to expect more thorough and detailed "in-line" or "in code" comments, making it clear how each of your functions works along with your main() and the rest.**

Scoring:

Proper Comments and Heading Comment Block: **20 points**
File compiles: **10 points** and `main()` is written to properly test all functions.
File compiles and accepts proper input and generates proper output: **70 points** distributed across 3 functions as indicated above. [1. Arrays: 15 points, 2. Matrices: 25 points, 3. Calculator: 30 points]

Handing in your Assignment.

You will have to use some form of a “Secure FTP” program. This could be the one that comes with puTTY, or it could be WinSCP, or it could be some other tool you are already familiar. Just be sure you understand you cannot simply “drag-and-drop” a file from your silo account to your Canvas Assignment. It will usually take two steps: 1: Move the file from your silo account to the computer you are using, 2. Upload the file from the computer you are using to Canvas with the usual method. If you have questions about this, ask them ASAP.