Class: CS 2640
Name: Keita Katsumi
ID: 016628924

Program Report

Section 1. Project description

The program sets up a 5 by 3 2D array filled up with values the programmer decided as a default value. Then it displays menu 1. Replace value, 2. Calculate the total Sum, 3. Print the 2D array, and 4. Exit program. The program keeps accepting an integer until the user inputs 4.

Section 2. Project specification

The project requires realizing the 2D array with pointer arithmetic of memory address. The challenging part of this project is the calculated offset of rows and columns in a virtual 2D array to access the correct memory location in RAM. Moreover, the nested loop helps to organize the code, increasing the risk of bugs. Therefore, it is essential to ensure that the inner loop and outer loop visit the correct memory location after calculating the offset accurately. Since the menu displays four options, I split this assignment into a few parts. 1. Set up and initialize a 2D array, 2. Ask the user to enter an integer with a loop that terminates when the user input "4", 3. Replace value option, 4. Calculate the total Sum option, and 5. Print the 2D array options. After achieving each task, I tested separately to ensure my sub-task worked properly. In the last phase, I combined all the sub-tasks in 1 single file and tested the file again. It let me save a lot of time and effort to finish this assignment.

Section 3. Testing methodology

As I explained above, I broke down this assignment into small portions. Testing small piece is easy to find bugs and fix them. After checking each subtask, I combined all the parts. For replace value function, I typed each row and column to check the position where I expected to put it. I was able to replace the value correct index with the intended value.

Section 4. Lessons learned.

Through the assignment, I learned how to access memory with a pointer. Especially to realize a logical 2D array, it is required to calculate the offset of rows and columns to pretend as the 2D array in memory. It helps to understand how the program behaves inside memory when I allocate a 2D array. Assembly language makes it visible what low-level language allocates memory.