CIS 200 - Lab 0202

1. Problem Statement

Create a program that uses a three-dimensional array to calculate monthly

2. Requirements

2.1 Assumptions

- User will only enter integers 1-12 for month
- User will only enter y or n for loop choice
- Command line input/output only

2.2 Specifications

- Program will initialize 3 dimensional array with given data
- 3 dimensional array does not change

3. Decomposition Diagram

- 3 Dimensional Array Monthly Sales Program
    - Input
        - User input integers 1-12
        - User inputs char y or n
    - Process
        - Iterate into array to calculate
    - Output
        - Print formatted

4. Test Strategy

- Valid Data
- Invalid Data

5. Test Plan Windows Version

***All test cases appear the same in table for Linux, validated with screenshots.

Test Strategy	#	Description	Input	Expected Output	Actual Output	Pass/Fail
Valid Data	1	1-12 Month	6	Correct Month	Correct Month	Pass
Valid Data	2	Show another month	Y	Ask for month	Asked	Pass
Valid Data	3	Show another month	n	exit	exit	Pass
Invalid Data	1	Month < 1	-1	Ask to retry	Asked	Pass
Invalid Data	2	Month > 12	13	Ask to retry	Asked	Pass

6. Initial Algorithm

1. Define Global Constants
    a. Number of Departments 2
    b. Number of Stores 2
    c. Number of Months 12
2. Create Function *printMonthlySales*
    a. Parameters and Returns
        i. Sales Array, Month
        ii. Void
    b.
3. *MAIN* FUNCTION
    a. Create and Initialize 3 Dimensional array *storeMonthlySales*

{ 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0, 4.1, 4.2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.1 }

b. Create 2 holder variables
    - i. One integer to hold month input
    - ii. One char to determine continue/exit

c. Do until user enters ‘n’ to stop program
    - i. Give user month selection
        1. 1-12 Valid
            - a. Compute Totals
            - b. Print Table
        2. Outside bounds invalid
    - ii. Prompt user: “Would you like to insert more positive integers?”

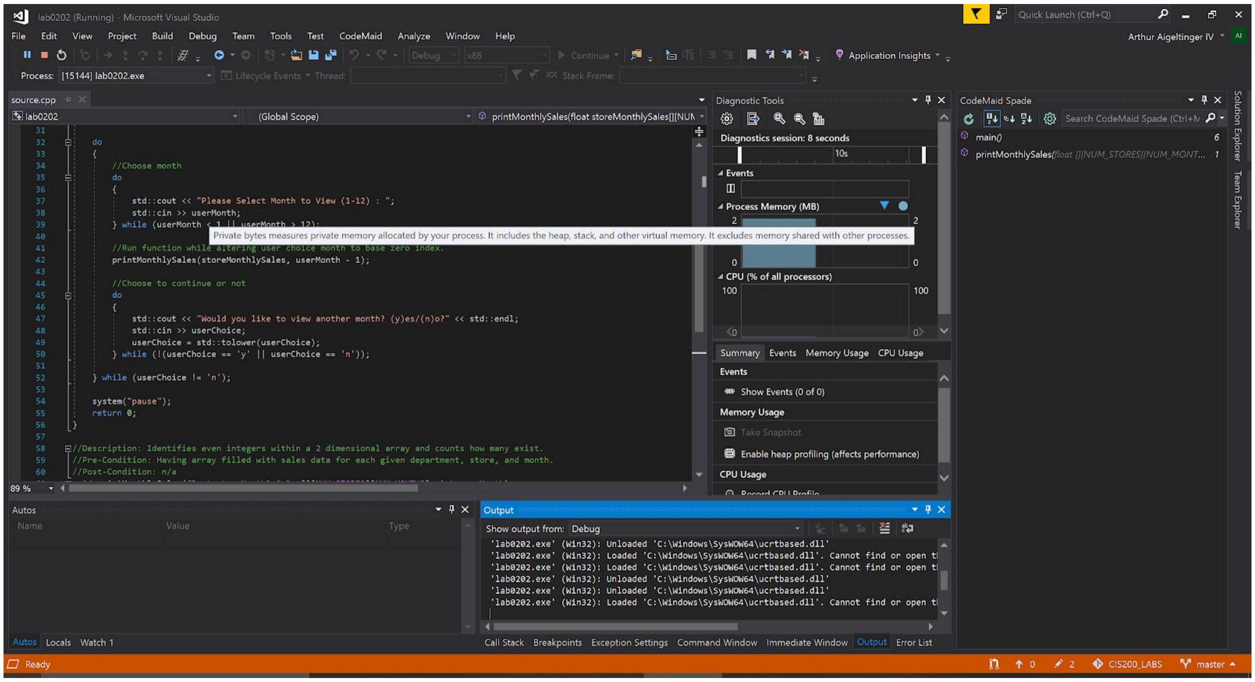
8. Code

```
//Program Name: Monthly Sales Calculator
//Programmer Name: Arthur Aigeltinger IV
//Description: Parses three dimensional array of sales data and computes totals for
given departments, stores, and totals. Then writes this data to a formatted table.
//Date Created: 09/25/18

#include <cctype>
#include <iomanip>
#include <iostream>

#define NUM_DEPTS 2
#define NUM_STORES 2
#define NUM_MONTHS 12

//Function Prototypes
void printMonthlySales(float[][NUM_STORES][NUM_MONTHS], int);


int main()
{
      //Initialize default array
      float storeMonthlySales[NUM_DEPTS][NUM_STORES][NUM_MONTHS] =
      {
            1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2,
            2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2,
            3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0, 4.1, 4.2,
            2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2
      };
      //Define and Initialize userChoice
```

```
    int userMonth = 0;
    char userChoice = ' ';

    do
    {
        //Choose month
        do
        {
            std::cout << "Please Select Month to View (1-12) : ";
            std::cin >> userMonth;
        } while (userMonth < 1 || userMonth > 12);

        //Run function while altering user choice month to base zero index.
        printMonthlySales(storeMonthlySales, userMonth - 1);

        //Choose to continue or not
        do
        {
            std::cout << "Would you like to view another month? (y)es/(n)o?"
<< std::endl;
            std::cin >> userChoice;
            userChoice = std::tolower(userChoice);
        } while (!(userChoice == 'y' || userChoice == 'n'));

    } while (userChoice != 'n');

    return 0;
}

//Description: Identifies even integers within a 2 dimensional array and counts how
many exist.
//Pre-Condition: Having array filled with sales data for each given department,
store, and month.
//Post-Condition: n/a
void printMonthlySales(float storeMonthlySales[][NUM_STORES][NUM_MONTHS], int
userMonth)
{
    //Calculate row and column additions on the fly
    float rowOne = storeMonthlySales[0][0][userMonth] +
storeMonthlySales[1][0][userMonth];
    float rowTwo = storeMonthlySales[0][1][userMonth] +
storeMonthlySales[1][1][userMonth];
    float colOne = storeMonthlySales[0][0][userMonth] +
storeMonthlySales[0][1][userMonth];
    float colTwo = storeMonthlySales[1][0][userMonth] +
storeMonthlySales[1][1][userMonth];
```

```
	float totalOverall = storeMonthlySales[0][0][userMonth] +
storeMonthlySales[0][1][userMonth] + storeMonthlySales[1][0][userMonth] +
storeMonthlySales[1][1][userMonth];

	//Formatting and printing table with values.
	std::cout << "            " << "Dept #1 " << "Dept #2 " << "Store Total" <<
std::endl;
	std::cout << "Store #1    " << std::setw(8) << std::left <<
storeMonthlySales[0][0][userMonth] << std::setw(8) << std::left <<
storeMonthlySales[1][0][userMonth] << rowOne << std::endl;
	std::cout << "Store #2    " << std::setw(8) << std::left <<
storeMonthlySales[0][1][userMonth] << std::setw(8) << std::left <<
storeMonthlySales[1][1][userMonth] << rowTwo << std::endl;
	std::cout << "Dept Total  " << std::setw(8) << std::left << colOne <<
std::setw(8) << std::left << colTwo << totalOverall << std::endl;
}
```

11. Screenshots

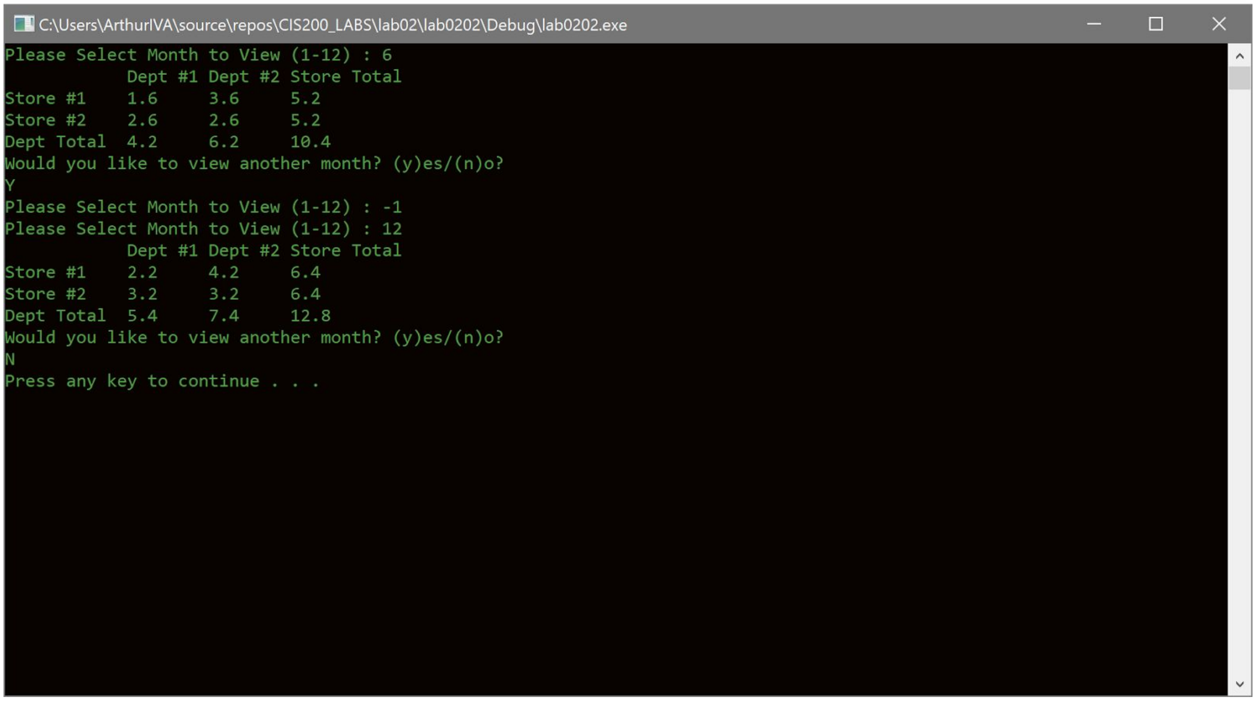


```
    do
    {
        //Choose month
        do
        {
            std::cout << "Please Select Month to View (1-12) : ";
            std::cin >> userMonth;
        } while (userMonth < 1 || userMonth > 12);

        //Run function while altering user choice month to base zero index.
        printMonthlySales(storeMonthlySales, userMonth - 1);

        //Choose to continue or not
        do
        {
            std::cout << "Would you like to view another month? (y)es/(n)o?" << std::endl;
            std::cin >> userChoice;
            userChoice = std::tolower(userChoice);
        } while (!(userChoice == 'y' || userChoice == 'n'));

    } while (userChoice != 'n');

    system("pause");
    return 0;
}

//Description: Identifies even integers within a 2 dimensional array and counts how many exist.
//Pre-Condition: Having array filled with sales data for each given department, store, and month.
//Post-Condition: n/a
```

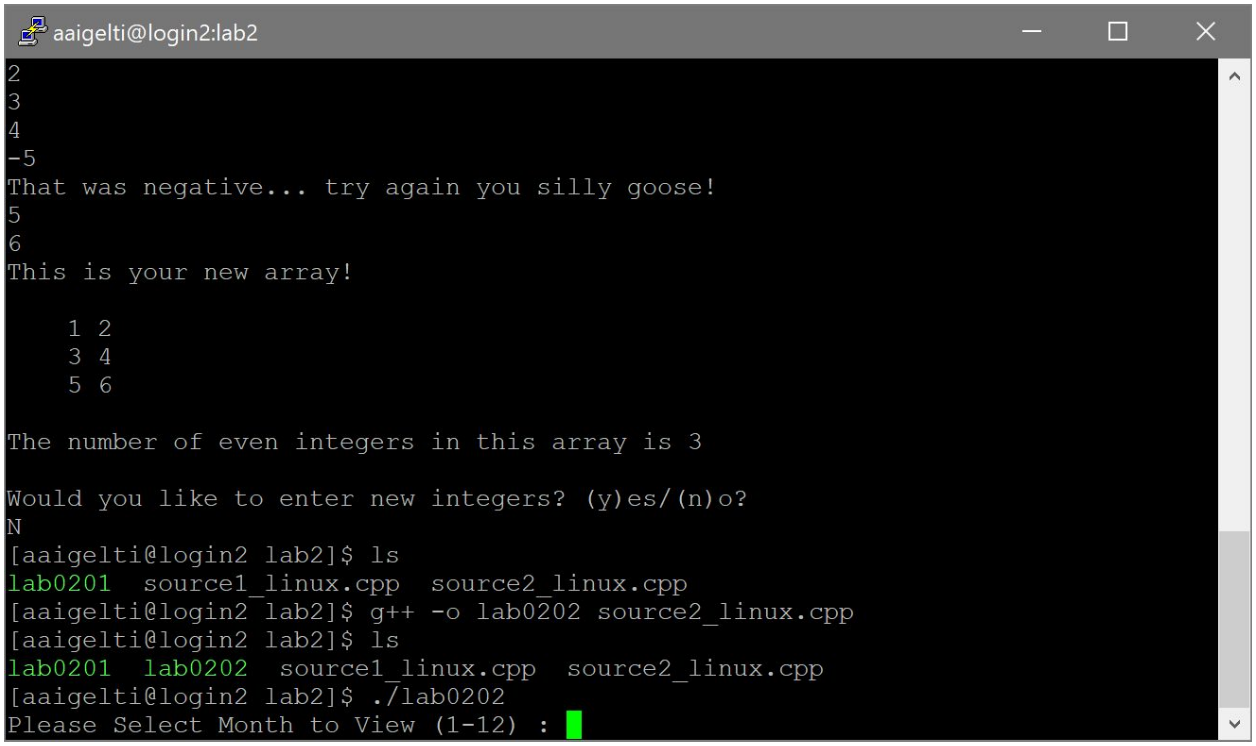
Private bytes measures private memory allocated by your process. It includes the heap, stack, and other virtual memory. It excludes memory shared with other processes.

Show output from: Debug

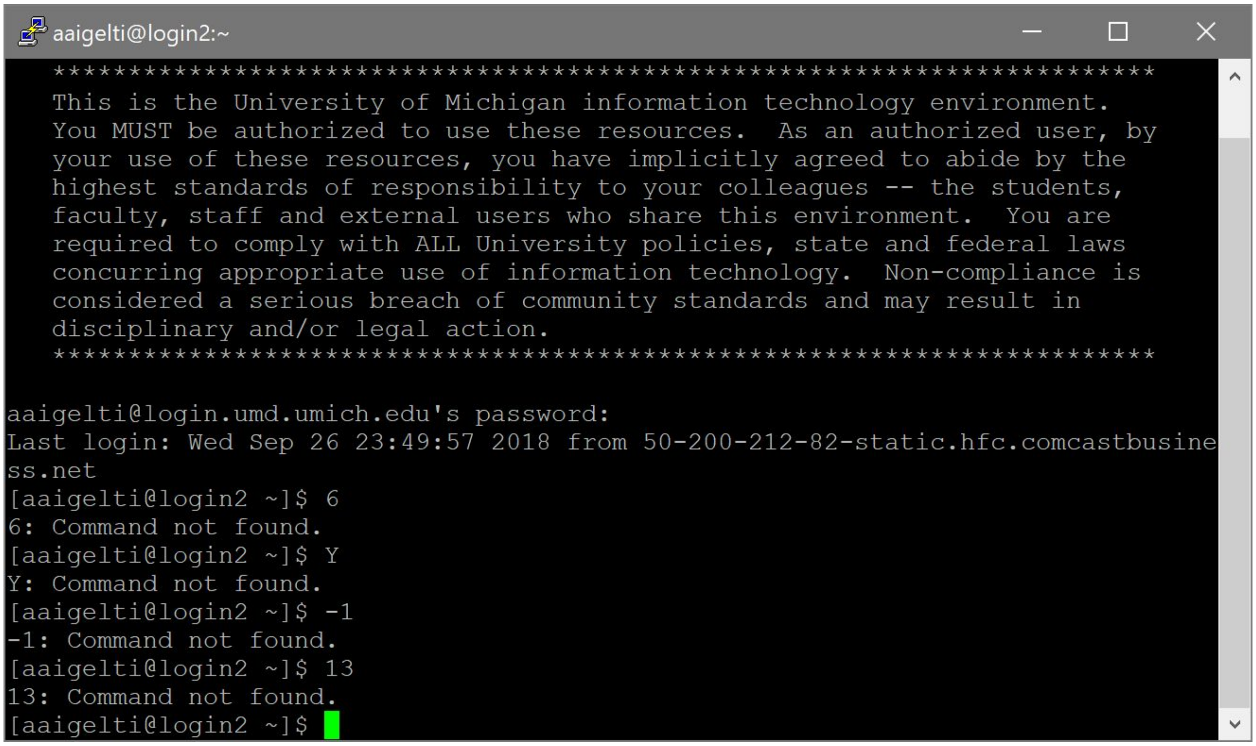
```
'lab0202.exe' (Win32): Unloaded 'C:\Windows\SysWOW64\ucrtbased.dll'
'lab0202.exe' (Win32): Loaded 'C:\Windows\SysWOW64\ucrtbased.dll'. Cannot find or open tl
'lab0202.exe' (Win32): Loaded 'C:\Windows\SysWOW64\ucrtbased.dll'. Cannot find or open tl
'lab0202.exe' (Win32): Unloaded 'C:\Windows\SysWOW64\ucrtbased.dll'
'lab0202.exe' (Win32): Unloaded 'C:\Windows\SysWOW64\ucrtbased.dll'
'lab0202.exe' (Win32): Loaded 'C:\Windows\SysWOW64\ucrtbased.dll'. Cannot find or open tl
```


C:\Users\ArthurIVA\source\repos\CIS200_LABS\lab02\lab0202\Debug\lab0202.exe

```
Please Select Month to View (1-12) : 6
            Dept #1 Dept #2 Store Total
Store #1    1.6     3.6     5.2
Store #2    2.6     2.6     5.2
Dept Total  4.2     6.2     10.4
Would you like to view another month? (y)es/(n)o?
Y
Please Select Month to View (1-12) : -1
Please Select Month to View (1-12) : 12
            Dept #1 Dept #2 Store Total
Store #1    2.2     4.2     6.4
Store #2    3.2     3.2     6.4
Dept Total  5.4     7.4     12.8
Would you like to view another month? (y)es/(n)o?
N
Press any key to continue . . .
```



```
2
3
4
-5
That was negative... try again you silly goose!
5
6
This is your new array!

    1 2
    3 4
    5 6

The number of even integers in this array is 3

Would you like to enter new integers? (y)es/(n)o?
N
[aaigelti@login2 lab2]$ ls
lab0201  source1_linux.cpp  source2_linux.cpp
[aaigelti@login2 lab2]$ g++ -o lab0202 source2_linux.cpp
[aaigelti@login2 lab2]$ ls
lab0201  lab0202  source1_linux.cpp  source2_linux.cpp
[aaigelti@login2 lab2]$ ./lab0202
Please Select Month to View (1-12) : 
```



```
   ***********************************************************************
   This is the University of Michigan information technology environment.
   You MUST be authorized to use these resources.  As an authorized user, by
   your use of these resources, you have implicitly agreed to abide by the
   highest standards of responsibility to your colleagues -- the students,
   faculty, staff and external users who share this environment.  You are
   required to comply with ALL University policies, state and federal laws
   concurring appropriate use of information technology.  Non-compliance is
   considered a serious breach of community standards and may result in
   disciplinary and/or legal action.
   ***********************************************************************

aaigelti@login.umd.umich.edu's password:
Last login: Wed Sep 26 23:49:57 2018 from 50-200-212-82-static.hfc.comcastbusine
ss.net
[aaigelti@login2 ~]$ 6
6: Command not found.
[aaigelti@login2 ~]$ Y
Y: Command not found.
[aaigelti@login2 ~]$ -1
-1: Command not found.
[aaigelti@login2 ~]$ 13
13: Command not found.
[aaigelti@login2 ~]$ 
```

13. Status

Program is operational in both Windows and Unix. Although sometimes ‘n’ in Unix spams.