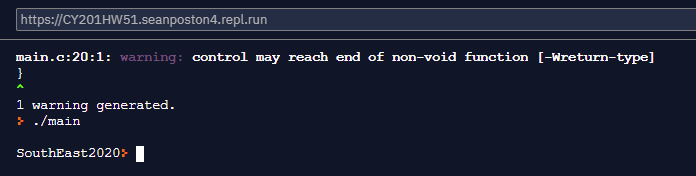
**Sean Poston**

**CS351**

**PI DAY**

**Question 1:** <https://repl.it/@seanposton4/CY201HW51>

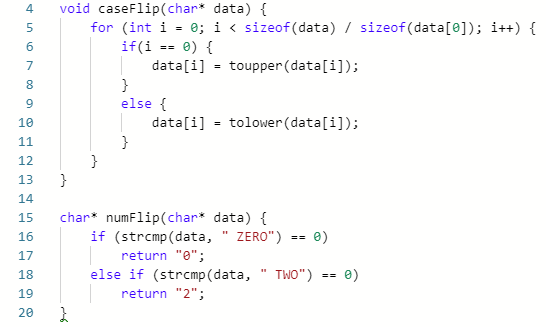
Output:



The code is written in a very iterative fashion. The program goes down the line, copies the proper amount of characters to a temp string, then does what is needed (flips the case of the character or returns a number based on the string).

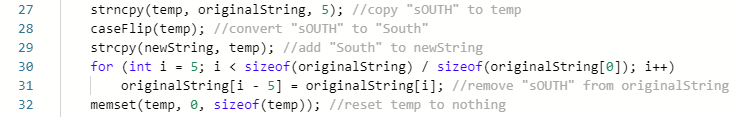
Improvement:

I could have maybe made one function with nested if statements. Instead of this:



I could have made one function using nested if statements just to save code.

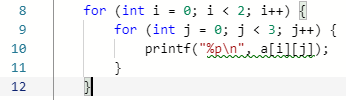
I could have also written a function to do the main part of the code:

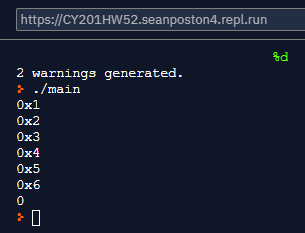


I end up writing this part about 5 times, and I could probably have easily moved it to a function with a few pointers. I did try to, but it didn’t immediately work so I returned to this version.

**Question 2:** <https://repl.it/@seanposton4/CY201HW52>

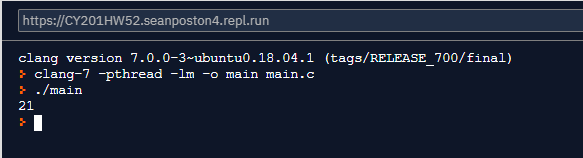
To figure this out, I tested this to make sure that the addresses of a multidimensional array were still all in a row:



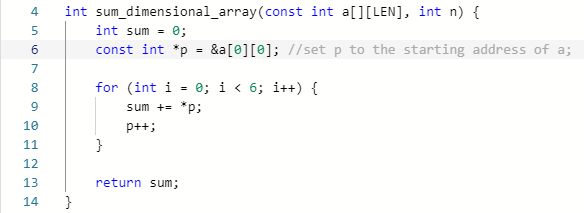


This was the output, which led me to believe that they were all in a row in the memory. So, it means that all that needs to be done is increment the pointer n times, where n is the number of elements in the array.

**Output:**



**New sum\_dimensional\_array function:**



**Source Codes:**

**Question 1:**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

void caseFlip(char\* data) {

for (int i = 0; i < sizeof(data) / sizeof(data[0]); i++) {

if(i == 0) {

data[i] = toupper(data[i]);

}

else {

data[i] = tolower(data[i]);

}

}

}

char\* numFlip(char\* data) {

if (strcmp(data, " ZERO") == 0)

return "0";

else if (strcmp(data, " TWO") == 0)

return "2";

}

int main(void) {

char originalString[] = "sOUTHeAST TWO ZERO TWO ZERO";

char newString[14];

char temp[10];

strncpy(temp, originalString, 5); //copy "sOUTH" to temp

caseFlip(temp); //convert "sOUTH" to "South"

strcpy(newString, temp); //add "South" to newString

for (int i = 5; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 5] = originalString[i]; //remove "sOUTH" from originalString

memset(temp, 0, sizeof(temp)); //reset temp to nothing

strncpy(temp, originalString, 4); //copy "eAST" to temp

caseFlip(temp);

strcat(newString, temp);

for (int i = 4; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 4] = originalString[i];

memset(temp, 0, sizeof(temp));

strncpy(temp, originalString, 4); //copy " TWO" to temp

strcat(newString, numFlip(temp));

for (int i = 4; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 4] = originalString[i];

memset(temp, 0, sizeof(temp));

strncpy(temp, originalString, 5); //copy " ZERO" to temp

strcat(newString, numFlip(temp));

for (int i = 5; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 5] = originalString[i];

memset(temp, 0, sizeof(temp));

    strncpy(temp, originalString, 4); //copy " TWO" to temp

strcat(newString, numFlip(temp));

for (int i = 4; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 4] = originalString[i];

memset(temp, 0, sizeof(temp));

    strncpy(temp, originalString, 5); //copy " ZERO" to temp

strcat(newString, numFlip(temp));

for (int i = 5; i < sizeof(originalString) / sizeof(originalString[0]); i++)

originalString[i - 5] = originalString[i];

memset(temp, 0, sizeof(temp));

printf("\n%s", newString);

return 0;

}

**Question 2:**

#include <stdio.h>

#define LEN 3

int sum\_dimensional\_array(const int a[][LEN], int n) {

int sum = 0;

    const int \*p = &a[0][0]; //set p to the starting address of a;

    for (int i = 0; i < 6; i++) {

        sum += \*p;

        p++;

    }

return sum;

}

int main(int argc, char \*argv[]) {

    int arry[2][3]={

        {1,2,3},

        {4,5,6}

    };

    printf("%d \n",sum\_dimensional\_array(arry, 6));

    return 0;

}