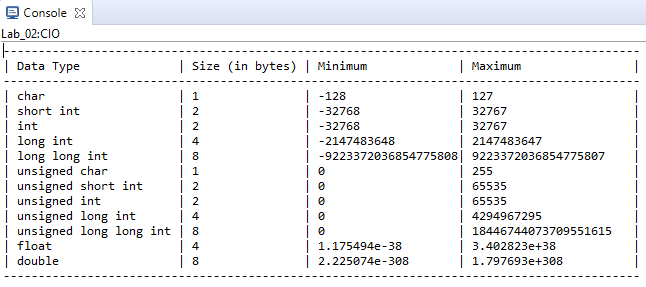
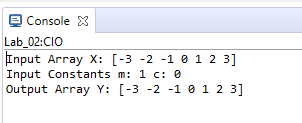
Lab 2 Solution

# Output Screenshots

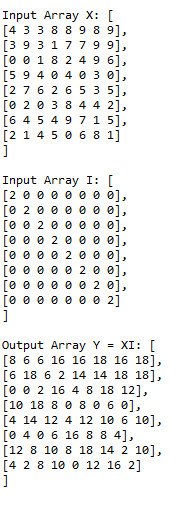
## Part 1



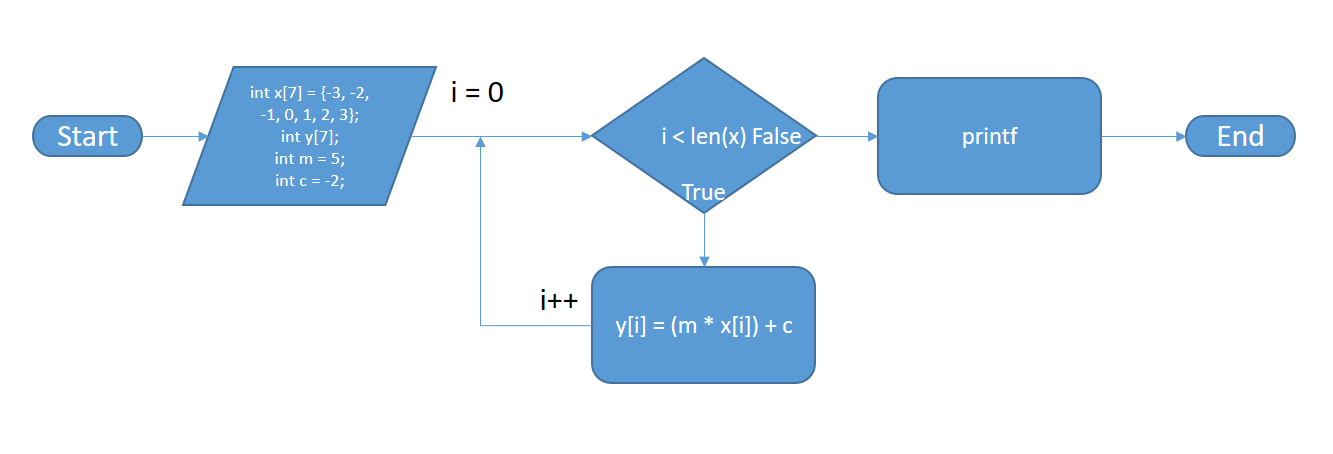
## Part 2



## Bonus



# Flow Diagram



# Source Code

#include <msp430.h>

#include <stdio.h>

#include <float.h>

#include <limits.h>

#define TABLE\_WIDTH 91

#define NUM\_TYPES 12

#define COL\_1\_WIDTH -23

#define COL\_2\_WIDTH -16

#define COL\_3\_WIDTH -20

#define COL\_4\_WIDTH -23

#define ARR\_LEN 7

/\*

--------------------------------------------------------

\*

File: Lab\_02/main.c

\*

Description: Print size and range of char, short int,

int, long int, long long int, unsigned

char, unsigned short int, unsigned int,

unsigned long int, unsigned long long int,

float, and double data types.

\*

Input: N/A

\*

Output: Print to stdout

\*

Author: Austin Bumbalough

\*

Lab Section: 8

\*

Date: 8/29/19

\*

Notes: Part 1 solution adapted from Stack Overflow user Paul Hankin https://stackoverflow.com/a/14418659

\*

--------------------------------------------------------

\*/

void part\_1**();**

void part\_2**();**

void bonus**();**

void print\_data\_type\_info**(**int**);**

int main**(**void**)**

**{**

WDTCTL **=** WDTPW **|** WDTHOLD**;** // stop watchdog timer

part\_1**();**

part\_2**();**

bonus**();**

**return** 0**;**

**}**

void part\_1**()** **{**

// Print table header row

int i**;**

**for** **(**i**=**0**;**i**<**TABLE\_WIDTH**;**i**++)** printf**(**"-"**);**

printf**(**"\n"**);**

printf**(**"| %-\*s| %-\*s| %-\*s| %-\*s|\n"**,** COL\_1\_WIDTH**,** "Data Type"**,** COL\_2\_WIDTH**,** "Size (in bytes)"**,** COL\_3\_WIDTH**,** "Minimum"**,** COL\_4\_WIDTH**,** "Maximum"**);**

**for** **(**i**=**0**;**i**<**TABLE\_WIDTH**;**i**++)** printf**(**"-"**);**

printf**(**"\n"**);**

// Print information for each data type

int j**;**

**for** **(**j**=**0**;**j**<**NUM\_TYPES**;**j**++)** **{**

print\_data\_type\_info**(**j**);**

**}**

// Print last row of table

**for** **(**i**=**0**;**i**<**TABLE\_WIDTH**;**i**++)** printf**(**"-"**);**

printf**(**"\n"**);**

**}**

void part\_2**()** **{**

// Declare variables for input array, constants, and output array

int x**[**ARR\_LEN**]** **=** **{-**3**,** **-**2**,** **-**1**,** 0**,** 1**,** 2**,** 3**};**

int m **=** 1**;**

int c **=** 0**;**

int y**[**7**];**

// Comput output array using y = mx+c

int i**;**

**for** **(**i**=**0**;**i**<**ARR\_LEN**;**i**++)** **{**

y**[**i**]** **=** **(**m **\*** x**[**i**])** **+** c**;**

**}**

// Print each input array element

printf**(**"Input Array X: ["**);**

**for** **(**i**=**0**;**i**<**ARR\_LEN**;**i**++)** **{**

**if** **(!(**i **==** ARR\_LEN **-**1**))** **{**

printf**(**"%d "**,** x**[**i**]);**

**}** **else** **{**

printf**(**"%d"**,** x**[**i**]);**

**}**

**}**

printf**(**"]\n"**);**

// Print constants

printf**(**"Input Constants m: %d c: %d\n"**,** m**,** c**);**

// Print each output array element

printf**(**"Output Array Y: ["**);**

**for** **(**i**=**0**;**i**<**ARR\_LEN**;**i**++)** **{**

**if** **(!(**i **==** ARR\_LEN **-**1**))** **{**

printf**(**"%d "**,** y**[**i**]);**

**}** **else** **{**

printf**(**"%d"**,** y**[**i**]);**

**}**

**}**

printf**(**"]\n"**);**

**}**

void bonus**()** **{**

// Declare input and output matrices and iteration variables

int x**[**8**][**8**]** **=** **{**

**{**4**,** 3**,** 3**,** 8**,** 8**,** 9**,** 8**,** 9**},**

**{**3**,** 9**,** 3**,** 1**,** 7**,** 7**,** 9**,** 9**},**

**{**0**,** 0**,** 1**,** 8**,** 2**,** 4**,** 9**,** 6**},**

**{**5**,** 9**,** 4**,** 0**,** 4**,** 0**,** 3**,** 0**},**

**{**2**,** 7**,** 6**,** 2**,** 6**,** 5**,** 3**,** 5**},**

**{**0**,** 2**,** 0**,** 3**,** 8**,** 4**,** 4**,** 2**},**

**{**6**,** 4**,** 5**,** 4**,** 9**,** 7**,** 1**,** 5**},**

**{**2**,** 1**,** 4**,** 5**,** 0**,** 6**,** 8**,** 1**}**

**};**

int i**[**8**][**8**]** **=** **{**

**{**2**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**},**

**{**0**,** 2**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**},**

**{**0**,** 0**,** 2**,** 0**,** 0**,** 0**,** 0**,** 0**},**

**{**0**,** 0**,** 0**,** 2**,** 0**,** 0**,** 0**,** 0**},**

**{**0**,** 0**,** 0**,** 0**,** 2**,** 0**,** 0**,** 0**},**

**{**0**,** 0**,** 0**,** 0**,** 0**,** 2**,** 0**,** 0**},**

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 2**,** 0**},**

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 2**}**

**};**

int y**[**8**][**8**];**

int j**;**

int k**;**

**for** **(**j**=**0**;**j**<**8**;**j**++)** **{**

**for** **(**k**=**0**;**k**<**8**;**k**++)** **{**

int sum **=** 0**;**

int m **=** 0**;**

// Multiply Row of X by Column of I and compute the sum

**while** **(**m **<** 8**)** **{**

sum **+=** x**[**j**][**m**]\***i**[**m**][**k**];**

m**++;**

**}**

y**[**j**][**k**]** **=** sum**;**

**}**

**}**

// Print input and output matrices

printf**(**"\nInput Array X: [\n"**);**

**for** **(**j**=**0**;**j**<**8**;**j**++)** **{**

printf**(**"["**);**

**for** **(**k**=**0**;**k**<**8**;**k**++)** **{**

**if** **(!(**k **==** 7**))** **{**

printf**(**"%d "**,** x**[**j**][**k**]);**

**}** **else** **{**

printf**(**"%d"**,** x**[**j**][**k**]);**

**}**

**}**

**if** **(!(**j **==** 7**))** **{**

printf**(**"],\n"**);**

**}** **else** **{**

printf**(**"]\n"**);**

**}**

**}**

printf**(**"]\n"**);**

printf**(**"\nInput Array I: [\n"**);**

**for** **(**j**=**0**;**j**<**8**;**j**++)** **{**

printf**(**"["**);**

**for** **(**k**=**0**;**k**<**8**;**k**++)** **{**

**if** **(!(**k **==** 7**))** **{**

printf**(**"%d "**,** i**[**j**][**k**]);**

**}** **else** **{**

printf**(**"%d"**,** i**[**j**][**k**]);**

**}**

**}**

**if** **(!(**j **==** 7**))** **{**

printf**(**"],\n"**);**

**}** **else** **{**

printf**(**"]\n"**);**

**}**

**}**

printf**(**"]\n"**);**

printf**(**"\nOutput Array Y = XI: [\n"**);**

**for** **(**j**=**0**;**j**<**8**;**j**++)** **{**

printf**(**"["**);**

**for** **(**k**=**0**;**k**<**8**;**k**++)** **{**

**if** **(!(**k **==** 7**))** **{**

printf**(**"%d "**,** y**[**j**][**k**]);**

**}** **else** **{**

printf**(**"%d"**,** y**[**j**][**k**]);**

**}**

**}**

**if** **(!(**j **==** 7**))** **{**

printf**(**"],\n"**);**

**}** **else** **{**

printf**(**"]\n"**);**

**}**

**}**

printf**(**"]\n"**);**

**}**

void print\_data\_type\_info**(**int i**)** **{**

**switch(**i**)** **{**

**case** 0**:** // Signed char

printf**(**"| %\*s| %\*d| %\*d| %\*d|\n"**,** COL\_1\_WIDTH**,** "char"**,** COL\_2\_WIDTH**,** **sizeof(**char**),** COL\_3\_WIDTH**,** SCHAR\_MIN**,** COL\_4\_WIDTH**,** SCHAR\_MAX**);**

**break;**

**case** 1**:** // Signed short int

printf**(**"| %\*s| %\*d| %\*d| %\*d|\n"**,** COL\_1\_WIDTH**,** "short int"**,** COL\_2\_WIDTH**,** **sizeof(**short int**),** COL\_3\_WIDTH**,** SHRT\_MIN**,** COL\_4\_WIDTH**,** SHRT\_MAX**);**

**break;**

**case** 2**:** // Signed int

printf**(**"| %\*s| %\*d| %\*d| %\*d|\n"**,** COL\_1\_WIDTH**,** "int"**,** COL\_2\_WIDTH**,** **sizeof(**int**),** COL\_3\_WIDTH**,** INT\_MIN**,** COL\_4\_WIDTH**,** INT\_MAX**);**

**break;**

**case** 3**:** // Signed long int

printf**(**"| %\*s| %\*d| %\*ld| %\*ld|\n"**,** COL\_1\_WIDTH**,** "long int"**,** COL\_2\_WIDTH**,** **sizeof(**long int**),** COL\_3\_WIDTH**,** LONG\_MIN**,** COL\_4\_WIDTH**,** LONG\_MAX**);**

**break;**

**case** 4**:** // Signed long long int

printf**(**"| %\*s| %\*d| %\*lld| %\*lld|\n"**,** COL\_1\_WIDTH**,** "long long int"**,** COL\_2\_WIDTH**,** **sizeof(**long long int**),** COL\_3\_WIDTH**,** LLONG\_MIN**,** COL\_4\_WIDTH**,** LLONG\_MAX**);**

**break;**

**case** 5**:** // Unsigned char

printf**(**"| %\*s| %\*d| %\*d| %\*d|\n"**,** COL\_1\_WIDTH**,** "unsigned char"**,** COL\_2\_WIDTH**,** **sizeof(**unsigned char**),** COL\_3\_WIDTH**,** 0**,** COL\_4\_WIDTH**,** CHAR\_MAX**);**

**break;**

**case** 6**:** // Unsigned short int

printf**(**"| %\*s| %\*d| %\*d| %\*u|\n"**,** COL\_1\_WIDTH**,** "unsigned short int"**,** COL\_2\_WIDTH**,** **sizeof(**unsigned short int**),** COL\_3\_WIDTH**,** 0**,** COL\_4\_WIDTH**,** USHRT\_MAX**);**

**break;**

**case** 7**:** // Unsigned int

printf**(**"| %\*s| %\*d| %\*d| %\*u|\n"**,** COL\_1\_WIDTH**,** "unsigned int"**,** COL\_2\_WIDTH**,** **sizeof(**unsigned int**),** COL\_3\_WIDTH**,** 0**,** COL\_4\_WIDTH**,** UINT\_MAX**);**

**break;**

**case** 8**:** // Unsigned long int

printf**(**"| %\*s| %\*d| %\*d| %\*lu|\n"**,** COL\_1\_WIDTH**,** "unsigned long int"**,** COL\_2\_WIDTH**,** **sizeof(**unsigned long int**),** COL\_3\_WIDTH**,** 0**,** COL\_4\_WIDTH**,** ULONG\_MAX**);**

**break;**

**case** 9**:** // Unsigned long long int

printf**(**"| %\*s| %\*d| %\*d| %\*llu|\n"**,** COL\_1\_WIDTH**,** "unsigned long long int"**,** COL\_2\_WIDTH**,** **sizeof(**unsigned long long int**),** COL\_3\_WIDTH**,** 0**,** COL\_4\_WIDTH**,** ULLONG\_MAX**);**

**break;**

**case** 10**:** // Float

printf**(**"| %\*s| %\*d| %\*e| %\*e|\n"**,** COL\_1\_WIDTH**,** "float"**,** COL\_2\_WIDTH**,** **sizeof(**float**),** COL\_3\_WIDTH**,** FLT\_MIN**,** COL\_4\_WIDTH**,** FLT\_MAX**);**

**break;**

**case** 11**:** // Double

printf**(**"| %\*s| %\*d| %\*e| %\*e|\n"**,** COL\_1\_WIDTH**,** "double"**,** COL\_2\_WIDTH**,** **sizeof(**double**),** COL\_3\_WIDTH**,** DBL\_MIN**,** COL\_4\_WIDTH**,** DBL\_MAX**);**

**break;**

**default:**

**return;**

**}**

**}**