NEW YORK CITY COLLEGE OF TECHNOLOGY

THE CITY UNIVERSITY OF NEW YORK

Department of Computer Engineering Technology

300 Jay Street, Brooklyn, NY 11201-1909

LAB REPORT

CET 3510 – OL71

(MICROCOMPUTER SYSTEMS TECHNOLOGY LABORATORY)

LAB #4

Memory Addresses

Name: Puja Roy

Date: 10/3/21

Due Date: 10/17/21

Table of Contents

Objective:

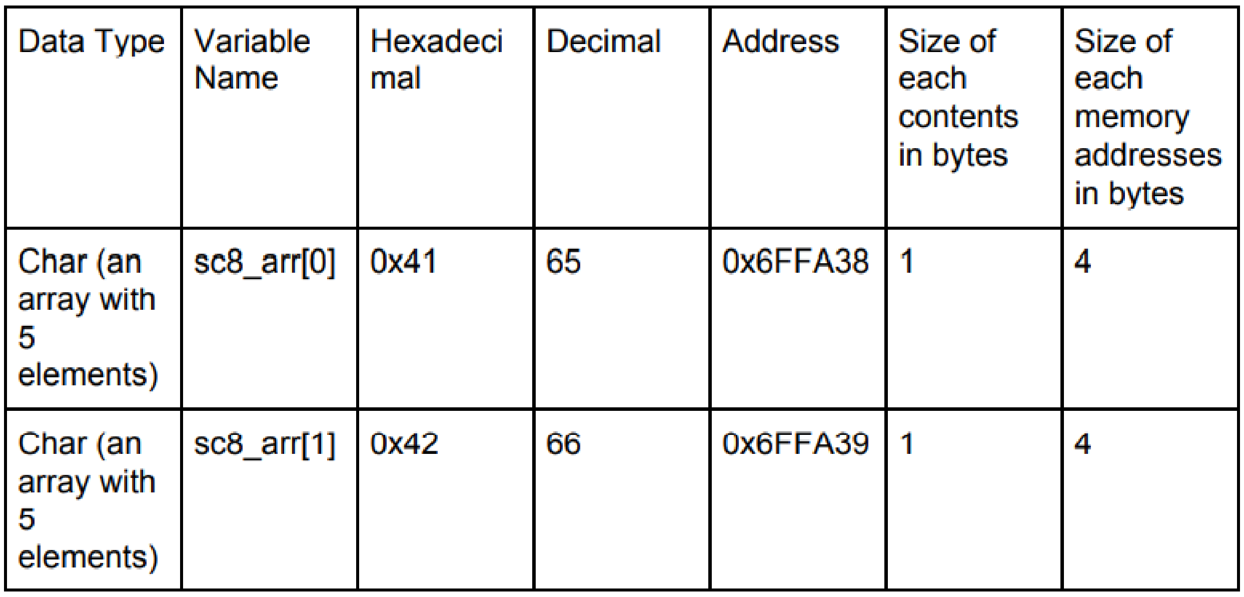
The purpose of this lab is to access and explain the computer memory and memory addresses. This lab will allow to learn and develop a program for moving data between the CPU and RAM and find the contents and addresses for each memory variable. I examined the address relationship among the memory variables and declared pointers. I also found the variable size in bytes.

Materials:

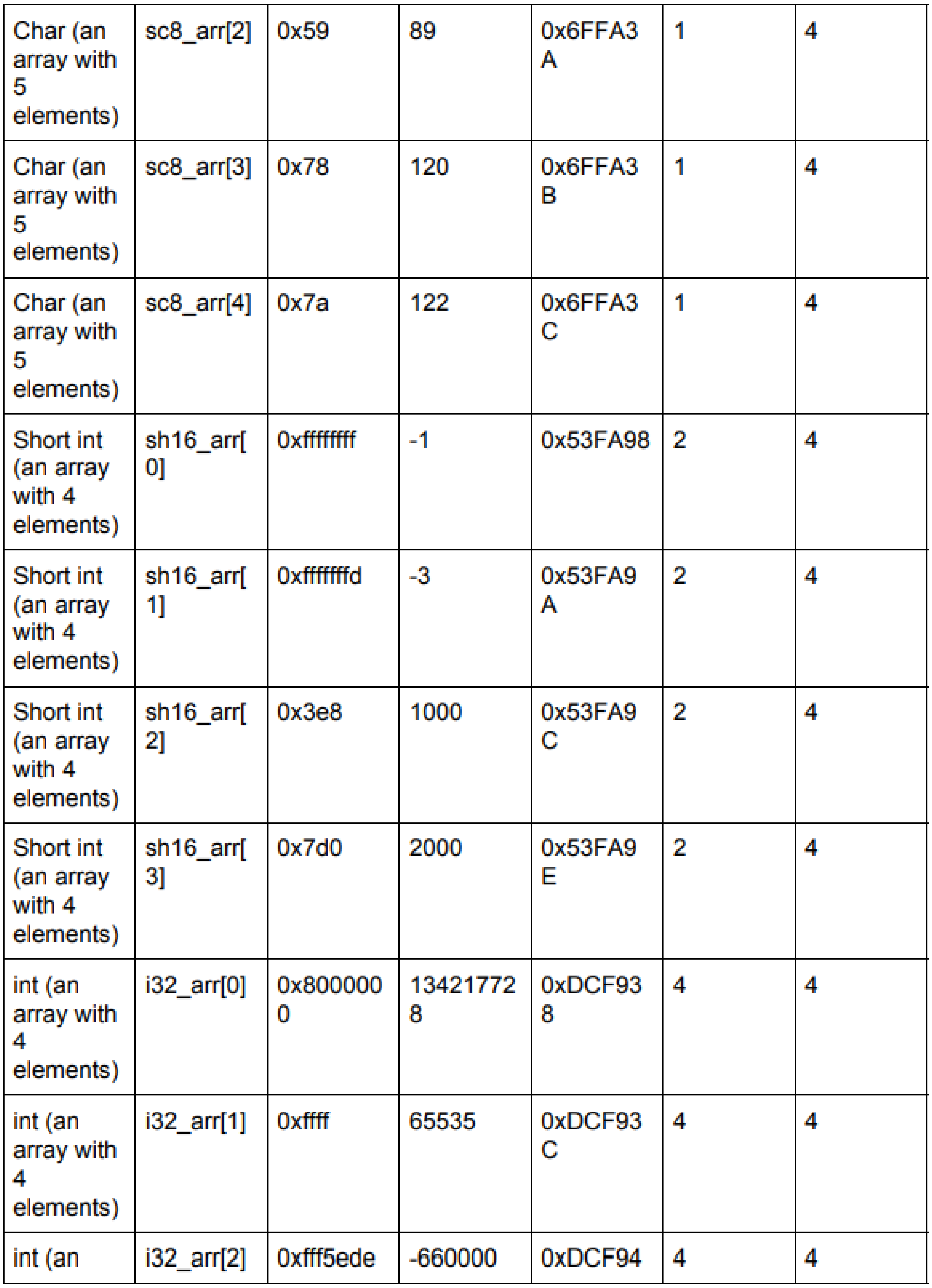
- Microsoft Visual Studio C++ Community Edition 2019

Procedure:

1. First, open Microsoft Visual Studio C++ Community Edition 2019
2. Type Program#1, compile, build and run the program
3. Include:
   - ➢ A pointer to store the address of each element in the array
   - ➢ The address of operator (&) to initialize a pointer
   - ➢ The index of array
   - ➢ Sizeof() operator to figure out the number of bytes of each element in the array
4. Modify Program#1 to code Program#2 by addressing 16-bit array
5. Find the contents for each element of a 16-bit array
6. Find the addresses for each element of a 16-bit array
7. Find the addresses for the number of bytes of each element in a 16-bit array
8. Find the addresses for the number of the addresses of each element in a 16-bit array
9. Modify Program#1 again to code Program#3 by addressing 32-bit array
10. Find the contents for each element of a 32-bit array
11. Find the addresses for each element of a 32-bit array
12. Find the addresses for the number of bytes of each element in a 32-bit array
13. Find the addresses for the number of the addresses of each element in a 32-bit array
14. Then, examine the relationship among the addresses in the memory and interpret the addresses for a byte, a word, and a double word
15. Draw a table of the relationship among the variable value and the memory address of the variable



Data Type	Variable Name	Hexadecimal	Decimal	Address	Size of each contents in bytes	Size of each memory addresses in bytes
Char (an array with 5 elements)	sc8_arr[0]	0x41	65	0x6FFA38	1	4
Char (an array with 5 elements)	sc8_arr[1]	0x42	66	0X6FFA39	1	4



Char (an array with 5 elements)	sc8_arr[2]	0x59	89	0x6FFA3A	1	4
Char (an array with 5 elements)	sc8_arr[3]	0x78	120	0x6FFA3B	1	4
Char (an array with 5 elements)	sc8_arr[4]	0x7a	122	0x6FFA3C	1	4
Short int (an array with 4 elements)	sh16_arr[0]	0xffffffff	-1	0x53FA98	2	4
Short int (an array with 4 elements)	sh16_arr[1]	0xfffffffd	-3	0x53FA9A	2	4
Short int (an array with 4 elements)	sh16_arr[2]	0x3e8	1000	0x53FA9C	2	4
Short int (an array with 4 elements)	sh16_arr[3]	0x7d0	2000	0x53FA9E	2	4
int (an array with 4 elements)	i32_arr[0]	0x8000000	134217728	0xDCF938	4	4
int (an array with 4 elements)	i32_arr[1]	0xffff	65535	0xDCF93C	4	4
int (an	i32_arr[2]	0xfff5ede	-660000	0xDCF94	4	4

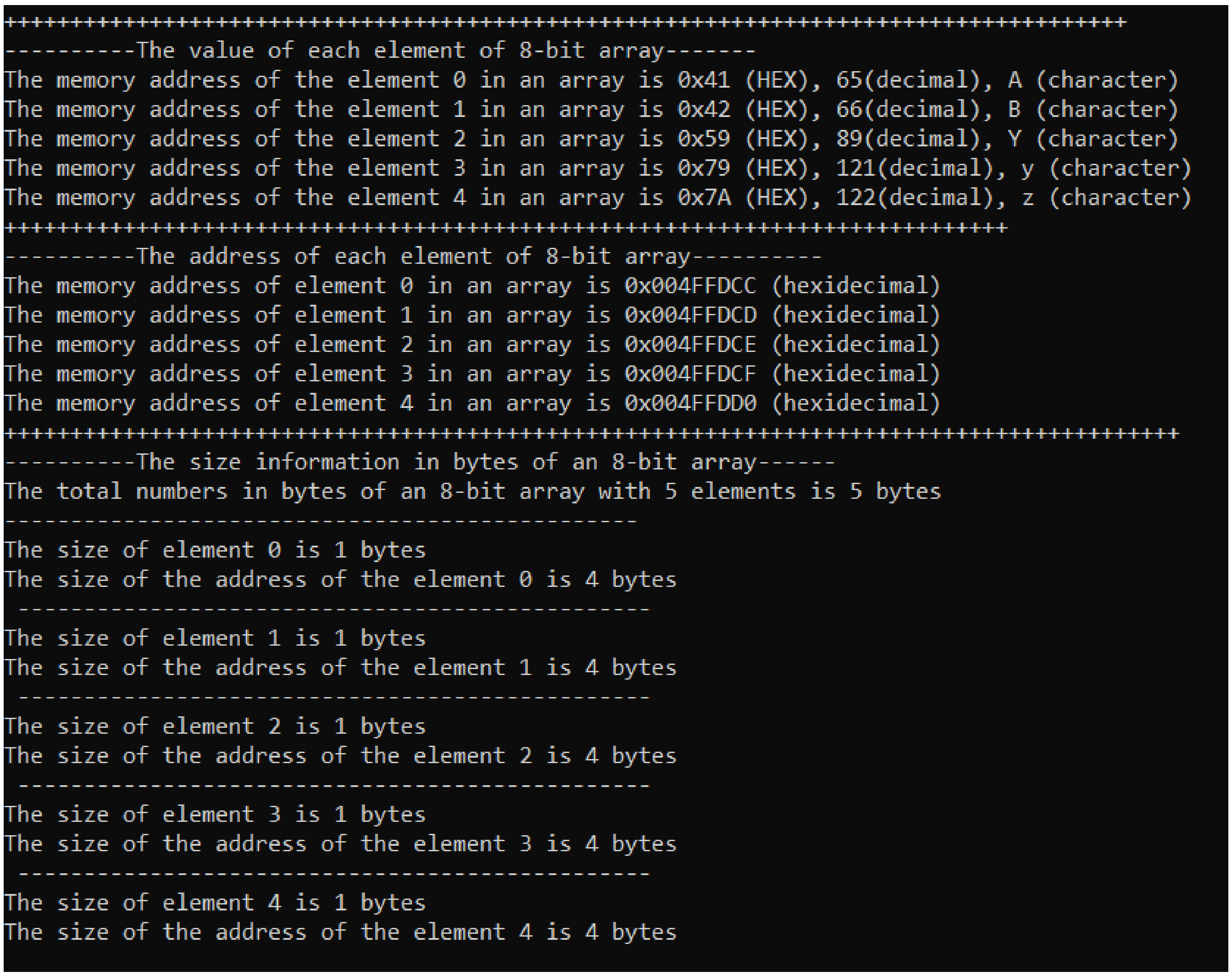
16. Then, analyze the information in the table to interpret the Little Endian Computer
17. Finally, exam the byte-addressable memory to store a byte, word and a double byte

Program#1:

```
#include <stdio.h>
#include <iostream>
int main()
{
	char sc8_arr[5] = { 0x41,
				   0x42,
				   0x59,
				   0x79,
				   0x7A };
	char* scPtr0, * scPtr1, * scPtr2, * scPtr3, * scPtr4;
	int md[5], madd[5];
	int i;  //used as index
```

```c
    // address for each element in an array sc8_arr[5]
    scPtr0 = &sc8_arr[0];
    scPtr1 = &sc8_arr[1];
    scPtr2 = &sc8_arr[2];
    scPtr3 = &sc8_arr[3];
    scPtr4 = &sc8_arr[4];

    //Display Hex value, decimal value, and char value for each element of char array
    printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
    printf("----------The value of each element of 8-bit array-------\n");
    for (i = 0; i < 5; i++)
    {
        printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
            i, sc8_arr[i], sc8_arr[i], sc8_arr[i]);
    }

    //Display address in hexadecimal for each element of char array
    printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
    printf("----------The address of each element of 8-bit array----------\n");
    for (i = 0; i < 5; i++)
    {
        printf("The memory address of element %d in an array is 0x%X (hexidecimal)\n", i, scPtr0 + i);
    }
    printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

    //Find the total numbers in byte of array sc8_arr[5]
    printf("----------The size information in bytes of an 8-bit array------\n");
    printf("The total numbers in bytes of an 8-bit array with 5 elements is %d bytes\n",
        sizeof(sc8_arr));

    //Find the total number of byte(s) of each element of array
    for (i = 0; i < 5; i++)
    {
        md[i] = sizeof(sc8_arr[i]);
        madd[i] = sizeof(scPtr0 + i);
        printf("------------------------------------------------\n");
        printf("The size of element %d is %d bytes\n", i, md[i]);
        printf("The size of the address of the element %d is %d bytes\n ", i, madd[i]);
    }
    system("pause");
    return 0;
}
```

Output:



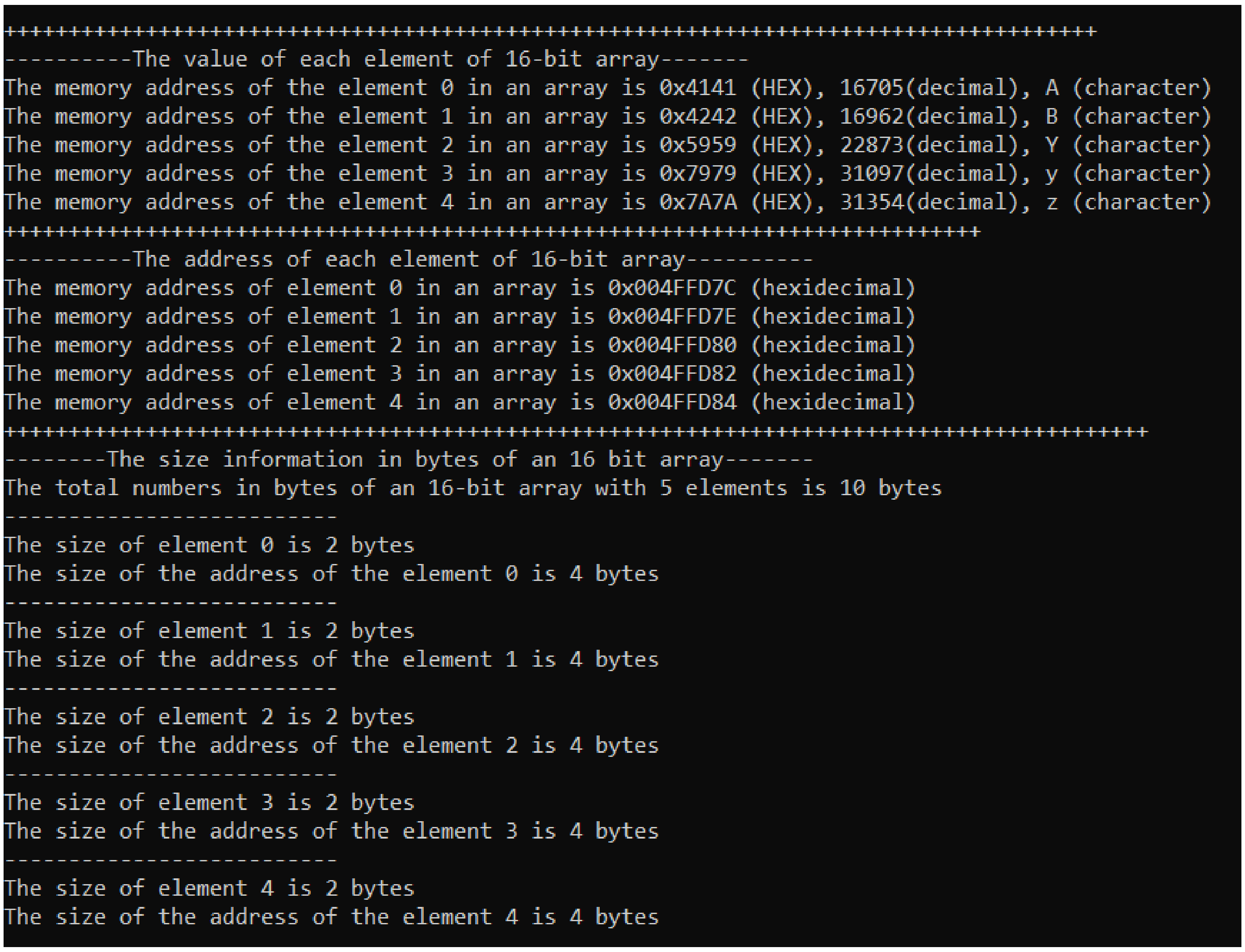
```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The value of each element of 8-bit array-------
The memory address of the element 0 in an array is 0x41 (HEX), 65(decimal), A (character)
The memory address of the element 1 in an array is 0x42 (HEX), 66(decimal), B (character)
The memory address of the element 2 in an array is 0x59 (HEX), 89(decimal), Y (character)
The memory address of the element 3 in an array is 0x79 (HEX), 121(decimal), y (character)
The memory address of the element 4 in an array is 0x7A (HEX), 122(decimal), z (character)
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The address of each element of 8-bit array----------
The memory address of element 0 in an array is 0x004FFDCC (hexidecimal)
The memory address of element 1 in an array is 0x004FFDCD (hexidecimal)
The memory address of element 2 in an array is 0x004FFDCE (hexidecimal)
The memory address of element 3 in an array is 0x004FFDCF (hexidecimal)
The memory address of element 4 in an array is 0x004FFDD0 (hexidecimal)
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The size information in bytes of an 8-bit array------
The total numbers in bytes of an 8-bit array with 5 elements is 5 bytes
------------------------------------------------
The size of element 0 is 1 bytes
The size of the address of the element 0 is 4 bytes
 ------------------------------------------------
The size of element 1 is 1 bytes
The size of the address of the element 1 is 4 bytes
 ------------------------------------------------
The size of element 2 is 1 bytes
The size of the address of the element 2 is 4 bytes
 ------------------------------------------------
The size of element 3 is 1 bytes
The size of the address of the element 3 is 4 bytes
 ------------------------------------------------
The size of element 4 is 1 bytes
The size of the address of the element 4 is 4 bytes
```

Program#2:

```
#include <stdio.h>
#include <iostream>
int main()
{
       char sc8_arr[5] = { 0x41,
                                         0x42,
                                         0x59,
                                         0x79,
                                         0x7A };
       char* scPtr0, * scPtr1, * scPtr2, * scPtr3, * scPtr4;


       short int sc16_arr[5]{ 0x4141,
                                            0x4242,
                                            0x5959,
                                            0x7979,
                                            0x7A7A };
       short int* sc16Ptr0, * sc16Ptr1, * sc16Ptr2, * sc16Ptr3, * sc16Ptr4;

       int md[5], madd[5];
       int i;  //used as index

       // address for each element in an array sc8_arr[5]
       scPtr0 = &sc8_arr[0];
       scPtr1 = &sc8_arr[1];
       scPtr2 = &sc8_arr[2];
       scPtr3 = &sc8_arr[3];
       scPtr4 = &sc8_arr[4];

       // address for each element in an array sc16_arr[5]
       sc16Ptr0 = &sc16_arr[0];
       sc16Ptr1 = &sc16_arr[1];
       sc16Ptr2 = &sc16_arr[2];
       sc16Ptr3 = &sc16_arr[3];
       sc16Ptr4 = &sc16_arr[4];

       //Display Hex value, decimal value, and char value for each element of char array
       printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
       printf("----------The value of each element of 8-bit array-------\n");
       for (i = 0; i < 5; i++)
       {
              printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
                     i, sc8_arr[i], sc8_arr[i], sc8_arr[i]);
       }

       //Display address in hexadecimal for each element of char array
       printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
       printf("----------The address of each element of 8-bit array----------\n");
       for (i = 0; i < 5; i++)
       {
              printf("The memory address of element %d in an array is 0x%p (hexidecimal)\n", i, scPtr0 + i);
       }
       printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

       //Find the total numbers in byte of array sc8_arr[5]
       printf("----------The size information in bytes of an 8-bit array------\n");
       printf("The total numbers in bytes of an 8-bit array with 5 elements is %d bytes\n",
              sizeof(sc8_arr));

       //Find the total number of byte(s) of each element of array
       for (i = 0; i < 5; i++)
       {
              md[i] = sizeof(sc8_arr[i]);
              madd[i] = sizeof(scPtr0 + i);
              printf("------------------------------------------------\n");
              printf("The size of element %d is %d bytes\n", i, md[i]);
              printf("The size of the address of the element %d is %d bytes\n ", i, madd[i]);
       }

       /********************************************** 16-bit print outs***************************/

//Display Hex value, decimal value, and char value for each element of char array
       printf("\n\n");
       printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
       printf("----------The value of each element of 16-bit array-------\n");
       for (i = 0; i < 5; i++)
       {
```

```
		printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
			i, sc16_arr[i], sc16_arr[i], sc16_arr[i]);
	}

	//Display address in hexadecimal for each element of char array
	printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
	printf("----------The address of each element of 8-bit array----------\n");
	for (i = 0; i < 5; i++)
	{
		printf("The memory address of element %d in an array is 0x%p (hexidecimal)\n", i, sc16Ptr0 + i);
	}
	printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

	// Find the total number in byte of array sc16_arr[5]
	printf("--------The size information in bytes of an 16 bit array-------\n");
	printf("The total numbers in bytes of an 16-bit array with 5 elements is %d bytes\n",
		sizeof(sc16_arr));

	//Find the total number of byte(s) of each element of array
	for (i = 0; i < 5; i++) {
		md[i] = sizeof(sc16_arr[i]);
		madd[i] = sizeof(sc16Ptr0 + i);
		printf("--------------------------\n");
		printf("The size of element %d is %d bytes\n",
			i,
			md[i]);
		printf("The size of the address of the element %d is %d bytes\n",
			i,
			madd[i]);
	}

	system("pause");
	return 0;
}
```

Output:



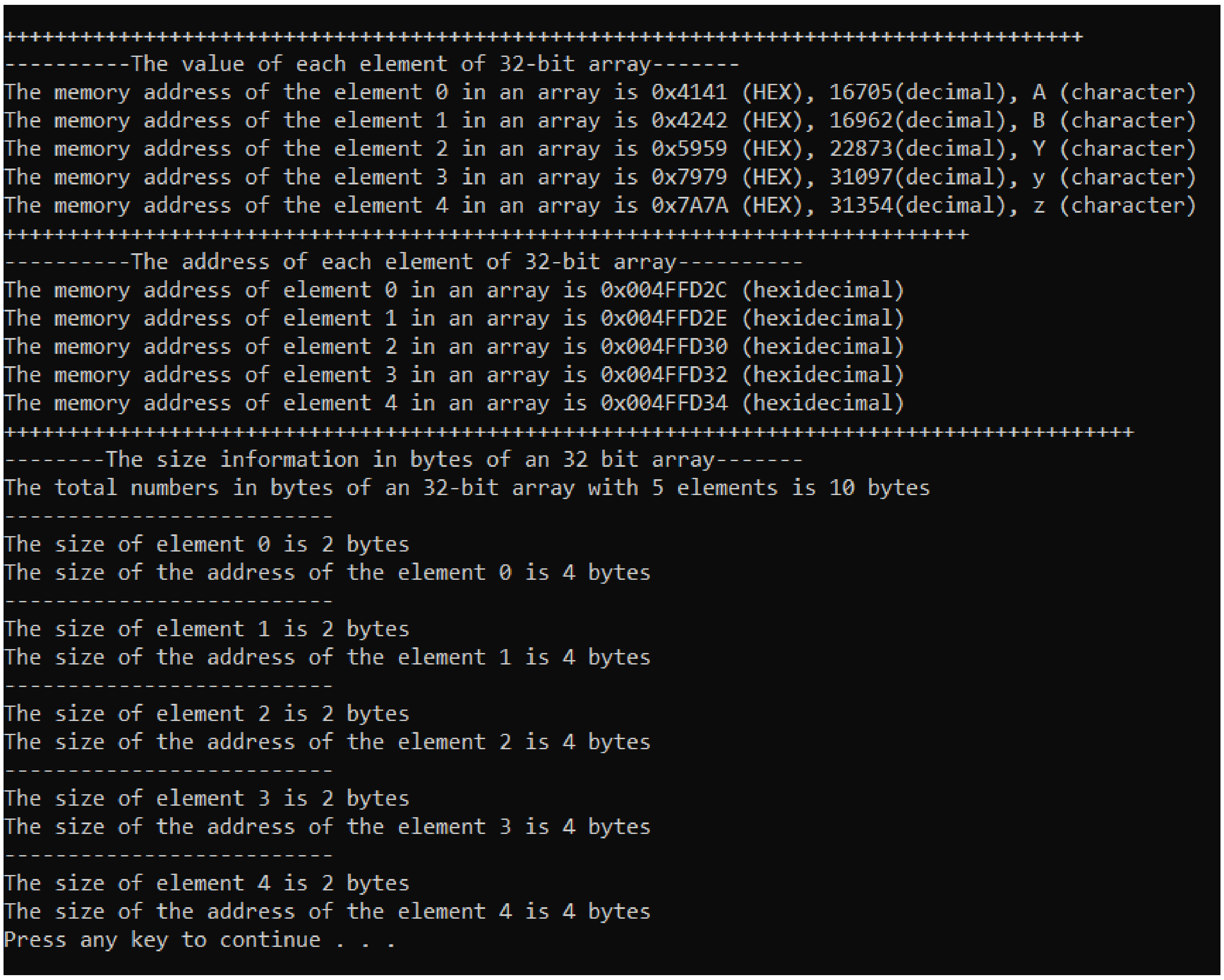
```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The value of each element of 16-bit array-------
The memory address of the element 0 in an array is 0x4141 (HEX), 16705(decimal), A (character)
The memory address of the element 1 in an array is 0x4242 (HEX), 16962(decimal), B (character)
The memory address of the element 2 in an array is 0x5959 (HEX), 22873(decimal), Y (character)
The memory address of the element 3 in an array is 0x7979 (HEX), 31097(decimal), y (character)
The memory address of the element 4 in an array is 0x7A7A (HEX), 31354(decimal), z (character)
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The address of each element of 16-bit array----------
The memory address of element 0 in an array is 0x004FFD7C (hexidecimal)
The memory address of element 1 in an array is 0x004FFD7E (hexidecimal)
The memory address of element 2 in an array is 0x004FFD80 (hexidecimal)
The memory address of element 3 in an array is 0x004FFD82 (hexidecimal)
The memory address of element 4 in an array is 0x004FFD84 (hexidecimal)
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
--------The size information in bytes of an 16 bit array-------
The total numbers in bytes of an 16-bit array with 5 elements is 10 bytes
--------------------------
The size of element 0 is 2 bytes
The size of the address of the element 0 is 4 bytes
--------------------------
The size of element 1 is 2 bytes
The size of the address of the element 1 is 4 bytes
--------------------------
The size of element 2 is 2 bytes
The size of the address of the element 2 is 4 bytes
--------------------------
The size of element 3 is 2 bytes
The size of the address of the element 3 is 4 bytes
--------------------------
The size of element 4 is 2 bytes
The size of the address of the element 4 is 4 bytes
```

Program#3:

```
#include <stdio.h>
#include <iostream>
int main()
{
        char sc8_arr[5] = { 0x41,
                                          0x42,
                                          0x59,
                                          0x79,
                                          0x7A };
        char* scPtr0, * scPtr1, * scPtr2, * scPtr3, * scPtr4;


        short int sc16_arr[5]{ 0x4141,
                                             0x4242,
                                             0x5959,
                                             0x7979,
                                             0x7A7A };
        short int* sc16Ptr0, * sc16Ptr1, * sc16Ptr2, * sc16Ptr3, * sc16Ptr4;


        short int sc32_arr[5]{ 0x4141,
                                             0x4242,
                                             0x5959,
                                             0x7979,
                                             0x7A7A };
        short int* sc32Ptr0, * sc32Ptr1, * sc32Ptr2, * sc32Ptr3, * sc32Ptr4;

        int md[5], madd[5];
        int i;  //used as index

        // address for each element in an array sc8_arr[5]
        scPtr0 = &sc8_arr[0];
        scPtr1 = &sc8_arr[1];
        scPtr2 = &sc8_arr[2];
        scPtr3 = &sc8_arr[3];
        scPtr4 = &sc8_arr[4];

        // address for each element in an array sc16_arr[5]
        sc16Ptr0 = &sc16_arr[0];
        sc16Ptr1 = &sc16_arr[1];
        sc16Ptr2 = &sc16_arr[2];
        sc16Ptr3 = &sc16_arr[3];
        sc16Ptr4 = &sc16_arr[4];

        // address for each element in an array sc32_arr[5]
        sc32Ptr0 = &sc32_arr[0];
        sc32Ptr1 = &sc32_arr[1];
        sc32Ptr2 = &sc32_arr[2];
        sc32Ptr3 = &sc32_arr[3];
        sc32Ptr4 = &sc32_arr[4];

        //Display Hex value, decimal value, and char value for each element of char array
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The value of each element of 8-bit array-------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
                        i, sc8_arr[i], sc8_arr[i], sc8_arr[i]);
        }

        //Display address in hexadecimal for each element of char array
        printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The address of each element of 8-bit array----------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of element %d in an array is 0x%p (hexidecimal)\n", i, scPtr0 + i);
        }
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

        //Find the total numbers in byte of array sc8_arr[5]
        printf("----------The size information in bytes of an 8-bit array------\n");
        printf("The total numbers in bytes of an 8-bit array with 5 elements is %d bytes\n",
                sizeof(sc8_arr));

        //Find the total number of byte(s) of each element of array
        for (i = 0; i < 5; i++)
        {
                md[i] = sizeof(sc8_arr[i]);
                madd[i] = sizeof(scPtr0 + i);
                printf("------------------------------------------------\n");
                printf("The size of element %d is %d bytes\n", i, md[i]);
                printf("The size of the address of the element %d is %d bytes\n ", i, madd[i]);
        }

        /********************************************** 16 bit print outs***************************/

//Display Hex value, decimal value, and char value for each element of char array
```

```
        printf("\n\n");
        printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The value of each element of 16-bit array-------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
                        i, sc16_arr[i], sc16_arr[i], sc16_arr[i]);
        }

        //Display address in hexadecimal for each element of char array
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The address of each element of 16-bit array----------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of element %d in an array is 0x%p (hexidecimal)\n", i, sc16Ptr0 + i);
        }
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

        // Find the total number in byte of array sc16_arr[5]
        printf("--------The size information in bytes of an 16 bit array-------\n");
        printf("The total numbers in bytes of an 16-bit array with 5 elements is %d bytes\n",
                sizeof(sc16_arr));

        //Find the total number of byte(s) of each element of array
        for (i = 0; i < 5; i++) {
                md[i] = sizeof(sc16_arr[i]);
                madd[i] = sizeof(sc16Ptr0 + i);
                printf("--------------------------\n");
                printf("The size of element %d is %d bytes\n",
                        i,
                        md[i]);
                printf("The size of the address of the element %d is %d bytes\n",
                        i,
                        madd[i]);
        }

        /********************************************** 32 bit print outs***************************/

//Display Hex value, decimal value, and char value for each element of char array
        printf("\n\n");
        printf("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The value of each element of 32-bit array-------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of the element %d in an array is 0x%X (HEX), %d(decimal), %c (character)\n",
                        i, sc32_arr[i], sc32_arr[i], sc32_arr[i]);
        }

        //Display address in hexadecimal for each element of char array
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("----------The address of each element of 32-bit array----------\n");
        for (i = 0; i < 5; i++)
        {
                printf("The memory address of element %d in an array is 0x%p (hexidecimal)\n", i, sc32Ptr0 + i);
        }
        printf("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n");

        // Find the total number in byte of array sc32_arr[5]
        printf("--------The size information in bytes of an 32 bit array-------\n");
        printf("The total numbers in bytes of an 32-bit array with 5 elements is %d bytes\n",
                sizeof(sc32_arr));

        //Find the total number of byte(s) of each element of array
        for (i = 0; i < 5; i++) {
                md[i] = sizeof(sc32_arr[i]);
                madd[i] = sizeof(sc32Ptr0 + i);
                printf("--------------------------\n");
                printf("The size of element %d is %d bytes\n",
                        i,
                        md[i]);
                printf("The size of the address of the element %d is %d bytes\n",
                        i,
                        madd[i]);
        }

        system("pause");
        return 0;
}
```

Output:



```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The value of each element of 32-bit array-------
The memory address of the element 0 in an array is 0x4141 (HEX), 16705(decimal), A (character)
The memory address of the element 1 in an array is 0x4242 (HEX), 16962(decimal), B (character)
The memory address of the element 2 in an array is 0x5959 (HEX), 22873(decimal), Y (character)
The memory address of the element 3 in an array is 0x7979 (HEX), 31097(decimal), y (character)
The memory address of the element 4 in an array is 0x7A7A (HEX), 31354(decimal), z (character)
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
----------The address of each element of 32-bit array----------
The memory address of element 0 in an array is 0x004FFD2C (hexidecimal)
The memory address of element 1 in an array is 0x004FFD2E (hexidecimal)
The memory address of element 2 in an array is 0x004FFD30 (hexidecimal)
The memory address of element 3 in an array is 0x004FFD32 (hexidecimal)
The memory address of element 4 in an array is 0x004FFD34 (hexidecimal)
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
--------The size information in bytes of an 32 bit array-------
The total numbers in bytes of an 32-bit array with 5 elements is 10 bytes
--------------------------
The size of element 0 is 2 bytes
The size of the address of the element 0 is 4 bytes
--------------------------
The size of element 1 is 2 bytes
The size of the address of the element 1 is 4 bytes
--------------------------
The size of element 2 is 2 bytes
The size of the address of the element 2 is 4 bytes
--------------------------
The size of element 3 is 2 bytes
The size of the address of the element 3 is 4 bytes
--------------------------
The size of element 4 is 2 bytes
The size of the address of the element 4 is 4 bytes
Press any key to continue . . .
```

Conclusion:

Throughout lab 4, I accessed and learned the computer memory and memory addresses. I learned to develop a program for moving data between the CPU and RAM and find the contents and addresses for each memory variable. To add on, I examined the address relationship among the memory variables and declared pointers. I also found the variable size in bytes. I distinguished how and where those variable takes place when you play with the code. Now, I know how the address of the operator to initialize a pointer comes in hand and each different bit array has different values that I found for each of them.