To:
Tenouk

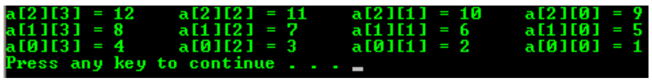
C LAB WORKSHEET 10a
2D Array Manipulation Part 2

1. More practice on 2D array data type.
2. Array indexes (rows and columns) manipulation.
3. Character, strings and 2D array.
4. Tutorial references that should be used together with this worksheet are C & C++ array part 1 and C & C++ array part 2.

7. Change the previous code with the following modification. Show the output and answer the questions.

```
#include <stdio.h>

void main()
{
     int i, j, a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};
     for(i = 2; i >= 0; i = i - 1)
     {
          for(j = 3; j >= 0; j = j - 1)
               printf("a[%d][%d] = %d\t", i, j, a[i][j]);
          printf("\n");
     }
}
```

a. When i is 2 and j is 3, the element in which row and which column is printed?
b. When i is 2 and j is 2, the element in which row and which column is printed?
c. When i is 2 and j is 0, the element in which row and which column is printed?
d. When i is 1 and j is 3, the element in which row and which column is printed?
e. When the last element is printed, what is i and what is j?



```
a[2][3] = 12    a[2][2] = 11    a[2][1] = 10    a[2][0] = 9
a[1][3] = 8     a[1][2] = 7     a[1][1] = 6     a[1][0] = 5
a[0][3] = 4     a[0][2] = 3     a[0][1] = 2     a[0][0] = 1
Press any key to continue . . .
```

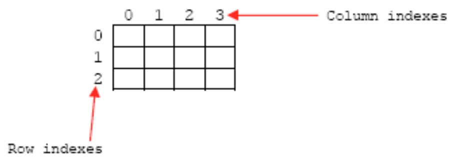
a. Element in third row and fourth column.
b. Element in third row and third column.
c. Element in the third row and first column.
d. Element in second row and fourth column.
e. i is 0 and j is 0.

It is confusing isn't it? To make it clearer we can include another curly braces in the array declaration as shown below. The order of the element in the array won't change in this program. However how the elements displayed can be different.

a[3][4] = {{1,2,3,4},{5,6,7,8},{9,10,11,12}};

8. In the following experiment, the first loop scans characters into a character array and the second prints them. For both loops, i stays fixed while j varies or the row stays fixed while the column varies. Hence, the data is read in the array row-wise and printed out row-wise. Enter this data: a, b, c, d, e, f, g, h, i, j, k, l. Remember to put a space preceding the %c in the scanf_s(). Show the output and answer the questions.

```
#include <stdio.h>

void main()
{
     int i, j;
     char a[3][4];
     for(i = 0; i <= 2; i = i + 1)
          for(j = 0; j <= 3; j = j + 1)
               // scanf(" %c", &a[i][j]);
               scanf_s(" %c", &a[i][j], sizeof(a));
     for(i = 0; i <= 2; i = i + 1)
     {
          for(j = 0; j <= 3; j = j + 1)
               printf("a[%d][%d] = %c\t", i, j, a[i][j]);
          printf("\n");
     }
}
```



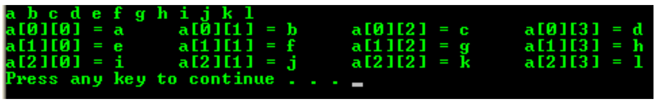
Questions for the reading loop:

a. Initially when i is 0 and j is 0, the first character entered is read. In the Figure above, write this first character read in the proper slot.
b. Write the second character in its proper slot. What is i and j?
c. Fill in all the characters that are read into this array properly.

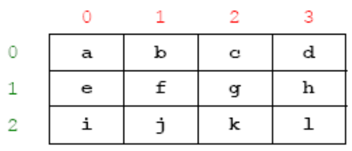
Questions for the printing loop.

a. Initially when i is 0 and j is 0, which character is printed?
b. What is the second character printed?
c. How many characters are printed before the '\n' is printed?
d. Was the first row or the first column of the array printed first?

Answers for the reading loop:



```
a b c d e f g h i j k l
a[0][0] = a     a[0][1] = b     a[0][2] = c     a[0][3] = d
a[1][0] = e     a[1][1] = f     a[1][2] = g     a[1][3] = h
a[2][0] = i     a[2][1] = j     a[2][2] = k     a[2][3] = l
Press any key to continue . . .
```



	0	1	2	3
0	a	b	c	d
1	e	f	g	h
2	i	j	k	l

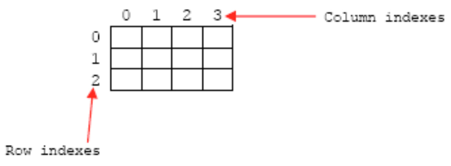
b. From the above figure, the green colour is i (row) and the red colour is j (column). Then, i is 0 and j is 1.

Answers for the printing loop:

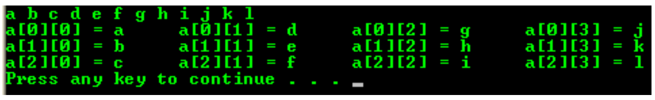
a. 'a' was printed.
b. 'b' is the second character that was printed.
c. Four characters.
d. The first row was printed first.

9. Use the same array and the input data from the previous program example. Do some modification as shown below for the previous code that is for reading only, i and j have been reverse.

```c
#include <stdio.h>

void main()
{
    int i, j;
    char a[3][4];
    for(i = 0; i <= 3; i = i + 1)
        for(j = 0; j <= 2; j = j + 1)
            // scanf(" %c", &a[j][i]);
            scanf_s(" %c", &a[j][i], sizeof(a));
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 3; j = j + 1)
            printf("a[%d][%d] = %c\t", i, j, a[i][j]);
        printf("\n");
    }
}
```



a. When reading, notice that j and not i is the row subscript. This is because j is used as the first index with a[ ][ ]. When reading, why did i go up to 3 and j go only to 2?
b. When reading and when i was 0 and j was 0, show the character that was read in the correct slot in the Figure.
c. When i was 0 and j was 1, show the character that was read.
d. Fill in all elements in the array as they are read in. Does your answer match with the output?



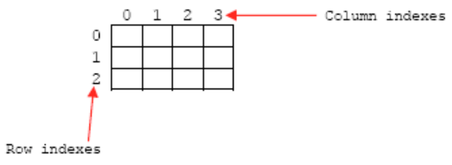
```
a b c d e f g h i j k l
a[0][0] = a     a[0][1] = d     a[0][2] = g     a[0][3] = j
a[1][0] = b     a[1][1] = e     a[1][2] = h     a[1][3] = k
a[2][0] = c     a[2][1] = f     a[2][2] = i     a[2][3] = l
Press any key to continue . . .
```

	0	1	2	3
0	a	d	g	j
1	b	e	h	k
2	c	f	i	l

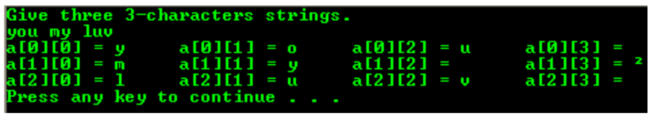
a. From the first two for loops, i <=3 and j <= 2.
b. When j was 0 and j was 0, the character read was a.
c. The character read was d.

10. Strings are read in by the rows. Each row will have one string. Enter the following data: “you”, “my”, “luv”. Remember that after each string, a null character is added. We are reading in strings but printing out only characters.

```c
#include <stdio.h>

void main()
{
    int i, j;
    char a[3][4];
    printf("Give three 3-characters strings.\n");
    for(i = 0; i <= 2; i = i + 1)
        // scanf("%s", &a[i]);
        scanf_s("%s", &a[i], 4);
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 3; j = j + 1)
            printf("a[%d][%d] = %c\t", i, j, a[i][j]);
        printf("\n");
    }
}
```



a. Show the contents of the array in memory after the three strings are read in the array.
b. Does your output agree?
c. How is the null character, ‘\0’ printed?
d. Is there a garbage character in a[1][3]? If so, why?



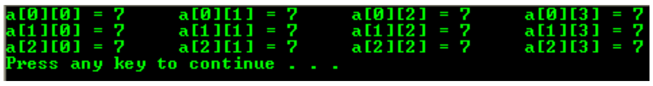
```
Give three 3-characters strings.
you my luv
a[0][0] = y     a[0][1] = o     a[0][2] = u     a[0][3] =
a[1][0] = m     a[1][1] = y     a[1][2] =       a[1][3] = ²
a[2][0] = l     a[2][1] = u     a[2][2] = v     a[2][3] =
Press any key to continue . . .
```

	0	1	2	3
0	y	o	u	\0
1	m	y	\0	
2	l	u	v	\0

a. Shown above.
b. The output matched except the garbage.
c. Just an empty space.
d. Yes. This slot has been reserved but not filled so whatever the previous data that has been stored here would be displayed (if possible).

11. Show the outputs for the following program examples. Can you explain what does the following program do?

```c
#include <stdio.h>

void main()
{
    int i, j, a[3][4];
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 3; j = j + 1)
        {
            a[i][j] = 7;
            printf("a[%d][%d] = %d\t", i, j, a[i][j]);
        }
```

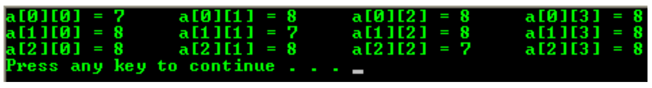


```
a[0][0] = 7     a[0][1] = 7     a[0][2] = 7     a[0][3] = 7
a[1][0] = 7     a[1][1] = 7     a[1][2] = 7     a[1][3] = 7
a[2][0] = 7     a[2][1] = 7     a[2][2] = 7     a[2][3] = 7
Press any key to continue . . .
```

In this program we assign or fill the array with integer 7.

```c
                printf("\n");
        }
}
```

```c
#include <stdio.h>

void main()
{
    int i, j, a[3][4];
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 3; j = j + 1)
        {
            if(i == j)
            {
                a[i][j] = 7;
                printf("a[%d][%d] = %d\t", i, j, a[i][j]);
            }
            else
            {
                a[i][j] = 8;
                printf("a[%d][%d] = %d\t", i, j, a[i][j]);
            }
        }
            printf("\n");
    }
}
```

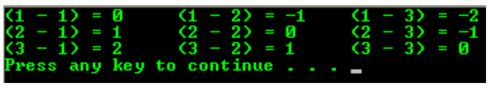


```
a[0][0] = 7     a[0][1] = 8     a[0][2] = 8     a[0][3] = 8
a[1][0] = 8     a[1][1] = 7     a[1][2] = 8     a[1][3] = 8
a[2][0] = 8     a[2][1] = 8     a[2][2] = 7     a[2][3] = 8
Press any key to continue . . .
```

In this program, if the i and j index are equal then fill an integer 7 and fill the rest with integer 8.

12. Show the output for each question and explain what the codes do.

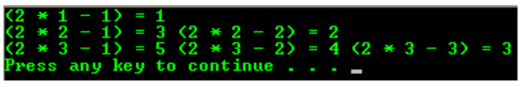
```c
#include <stdio.h>

void main()
{
    int i, j;
    for(i = 1; i <= 3; i = i + 1)
    {
        for(j = 1; j <= 3; j = j + 1)
            printf("(%d - %d) = %d\t", i, j, i - j);
        printf("\n");
    }
}
```



```
(1 - 1) = 0     (1 - 2) = -1    (1 - 3) = -2
(2 - 1) = 1     (2 - 2) = 0     (2 - 3) = -1
(3 - 1) = 2     (3 - 2) = 1     (3 - 3) = 0
Press any key to continue . . .
```

The array elements are the yields of the i - j or subtraction operations.

```c
#include <stdio.h>

void main()
{
    int i, j;
    for(i = 1; i <= 3; i = i + 1)
    {
        for(j = 1; j <= i; j = j + 1)
            printf("(2 * %d - %d) = %d\t", i, j, 2 * i - j);
        printf("\n");
    }
}
```



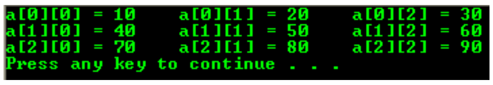
```
(2 * 1 - 1) = 1
(2 * 2 - 1) = 3 (2 * 2 - 2) = 2
(2 * 3 - 1) = 5 (2 * 3 - 2) = 4 (2 * 3 - 3) = 3
Press any key to continue . . .
```

In this program the array element are the yields of the 2*i-j operations while j <= i.

13. Show the output for each of the following programs. Use the following sample array data:

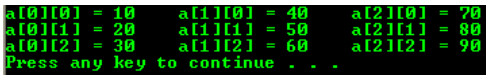
```c
int a[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};
```

```c
#include <stdio.h>

void main()
{
    int i, j;
    int a[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 2; j = j + 1)
            printf("a[%d][%d] = %d\t", i, j, a[i][j]);
        printf("\n");
    }
}
```



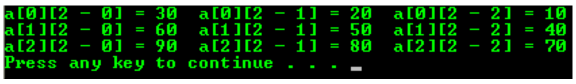
```
a[0][0] = 10    a[0][1] = 20    a[0][2] = 30
a[1][0] = 40    a[1][1] = 50    a[1][2] = 60
a[2][0] = 70    a[2][1] = 80    a[2][2] = 90
Press any key to continue . . .
```

```c
#include <stdio.h>

void main()
{
    int i, j;
    int a[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};
    for(i = 0; i <= 2; i = i + 1)
    {
        for(j = 0; j <= 2; j = j + 1)
            printf("a[%d][%d] = %d\t", j, i, a[j][i]);
        printf("\n");
    }
}
```



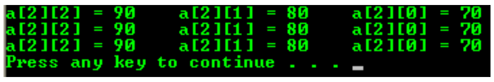
```
a[0][0] = 10    a[1][0] = 40    a[2][0] = 70
a[0][1] = 20    a[1][1] = 50    a[2][1] = 80
a[0][2] = 30    a[1][2] = 60    a[2][2] = 90
Press any key to continue . . .
```

```
#include <stdio.h>

void main()
{
     int i, j;
     int a[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};
     for(i = 0; i <= 2; i = i + 1)
     {
          for(j = 0; j <= 2; j = j + 1)
               printf("a[%d][2 - %d] = %d  ", i, j, a[i][2 - j]);
          printf("\n");
     }
}
```



```
a[0][2 - 0] = 30  a[0][2 - 1] = 20  a[0][2 - 2] = 10
a[1][2 - 0] = 60  a[1][2 - 1] = 50  a[1][2 - 2] = 40
a[2][2 - 0] = 90  a[2][2 - 1] = 80  a[2][2 - 2] = 70
Press any key to continue . . .
```

```
#include <stdio.h>

void main()
{
     int i, j;
     int a[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};
     for(i = 0; i <= 2; i = i + 1)
     {
          for(j = 2; j >= 0; j = j - 1)
               printf("a[2][%d] = %d\t",  j, a[2][j]);
          printf("\n");
     }
}
```



```
a[2][2] = 90    a[2][1] = 80    a[2][0] = 70
a[2][2] = 90    a[2][1] = 80    a[2][0] = 70
a[2][2] = 90    a[2][1] = 80    a[2][0] = 70
Press any key to continue . . .
```