CS1010 Programming Methodology

Week 8: Two-Dimensional Arrays

> Man's mind, once stretched by a new idea, never regains its original dimensions. ~*Oliver Wendell Holmes*

To students:

Many programs for this discussion can be downloaded from cs1010 account. For example, to copy **Week8_Q1.c**, you can type:

```
cp  ~cs1010/discussion/Week8_Q1.c  .
```

I. Basic Syntax

1. Given the following program **Week8_Q1.c**

Download source code from cs1010 account

```
#include <stdio.h>
#define MAX_ROW 2
#define MAX_COL 3

void printArray(int [][], int, int);

int main(void) {
   int values[MAX_ROW][MAX_COL];
   int row, col;

   printf("Enter values: \n");
   for (row=0; row<MAX_ROW; row++)
      for (col=0; col<MAX_COL; col++)
         scanf("%d", values[row][col]);

   printf("Array entered contains:\n");
   printArray(values, MAX_ROW, MAX_COL);

   return 0;
}

// To print elements in array arr
void printArray(int arr[][], int row_size, int col_size) {
   int row, col;

   for (row=0; row<row_size; row++)
   {
      for (col=0; col<col_size; col++)
         printf("%d ", arr[row][col]);
      printf("\n");
   }
}
```

(a) Spot the errors in above program.

(b) After you have corrected the errors in the program, run it by entering the data in each of the following formats. Do they work? What can you deduce?

```
8 1 2
3 0 9
```

```
  8  1   2
 3  0     9
```

```
8 1
2 3
0 9
```

```
8
1 2 3

0 9
```

```
8
1
2
3
0
9
```

2. Modify your corrected program for Q1 above by changing the integer array into a **char array**, and changing the **%d** format specifier in the printf and scanf statement to **%c**.

   Run your program on the following input. Does it work? If not, why and how would you correct the program in a simple way without having to rewrite much of the code?

```
abc
def
```

3. **Manual tracing**: For each of the code fragments below, write out its output.

   (a) This is adapted from CS1101 exam paper AY2006/7 Semester 1.

```
   int sum[4][4], k, m, n;

   for (k=0; k<4; k++) sum[k][0] = 1;
   for (k=0; k<4; k++) sum[0][k] = 1;

   for (m=1; m<4; m++)
      for (n=1; n<4; n++)
         sum[m][n] = sum[m-1][n] + sum[m][n-1];

   for (n=1; n<4; n++)
      printf("%d ", sum[3][n]);
   printf("\n");
```

Download source code **Week8_Q3a.c** from cs1010 account for verification after manual tracing

(b) This is adapted from CS1101 exam paper AY2008/9 Semester 1.

```
int array[][3] = { {1,1,1}, {2,2,2}, {3,3,3} };
int i, j;

for (i=0; i<3; i++)
   for (j=0; j<3; j++)
      array[i][j] += array[j][i];

for (i=0; i<3; i++)
{
   for (j=0; j<3; j++)
      printf("%d ", array[i][j]);
   printf("\n");
}
```

Download source code **Week8_Q3b.c** from cs1010 account for verification after manual tracing

II. Self-exploration (non-examinable)

4. Why is it that for a two-dimensional array, the size of the second dimension must be included in the function header or prototype? Example:

```
void printArray(int [][10], int);
```

III. Problem Solving with 2D Arrays

5. A **square matrix** is a two-dimensional array where the number of rows and columns are the same. Write a program **Wee8_Q5.c** to read in values for an *n*×*n* square matrix containing integer values, and check whether the matrix is (a) a diagonal matrix, or (b) an upper-triangular matrix.

A **diagonal matrix** is a square matrix in which the elements outside the main diagonal (↘) are all zeroes, for example:

$$\begin{bmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & -2 \end{bmatrix} \qquad \begin{bmatrix} 12 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & 7 \end{bmatrix}$$

An **upper triangular matrix** (or right triangular matrix) is a square matrix U of the form:

$$U_{ij} = \begin{cases} a_{ij} & \text{for } i \le j \\ 0 & \text{for } i > j. \end{cases}$$

Written explicitly,

$$\mathsf{U} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

Note that a diagonal matrix is also an upper triangular matrix.

A sample run is shown below. The first line contains a single integer indicating the size of the square matrix, *n*. The next *n*×*n* values are the elements of the matrix. The output is in bold.

Download skeleton **Week8_Q5.c** from cs1010 account

```
5
2 -1 3 4 1
0 7 5 -2 0
0 0 6 0 4
0 0 0 0 8
0 0 0 0 2
Matrix is not a diagonal matrix.
Matrix is an upper triangular matrix.
```

6. **UNIX Input and output redirection**

If the array is big, it might be too troublesome to key in all its elements interactively. One approach is to use a file to store the input data and run the program to read values from this input file. This will be covered later.

An alternative at this moment is to use the input redirection feature '**<**' in UNIX. Create an input text file called **matrix.in** with the following content using any editor:

```
5
2 -1 3 4 1
0 7 5 -2 0
0 0 6 0 4
0 0 0 0 8
0 0 0 0 2
```

When you run your matrix program in question 5 above, you just need to type this:

```
square_matrix < matrix.in
```

The program will read data from matrix.in as if you have entered these data on the keyboard. Try this out.

You may also do output redirection **>** in UNIX. Try out the following commands.

```
square_matrix < matrix.in > outfile
cat outfile
```

IV. Term Test Review

Your discussion leader will discuss some of the term test questions with you.