Chapter 9
Multidimensional Arrays and Double Indirection
Topics

• Two Dimensional Arrays
• Storage Class and Scope of Multi-Dimensional Arrays
• Arrays with more than 2 dimensions
• Arrays of Pointers
• Double Indirection
• Command Line Parameters
Two Dimensional Arrays

• Up till now, all arrays were considered to be one dimensional.
• Each array had a set of values that were associated with a single index.
Two Dimensional Arrays

• A two-dimensional array still has only one value per cell, but the location of the cell is handled by two indexes.

• It is sometimes handy to think of a two-dimensional array as being a set of rows and columns of a table.
## Two Dimensional Arrays

<table>
<thead>
<tr>
<th>Row</th>
<th>Col 0</th>
<th>Col 1</th>
<th>Col 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 0</td>
<td>6.1</td>
<td>1.2</td>
<td>8.3</td>
</tr>
<tr>
<td>Row 1</td>
<td>7.4</td>
<td>5.5</td>
<td>3.6</td>
</tr>
<tr>
<td>Row 2</td>
<td>2.7</td>
<td>9.8</td>
<td>4.9</td>
</tr>
</tbody>
</table>
Two Dimensional Arrays

- datatype identifier[first limit][second limit];
- float two_d_float[2][3];
- float two_d_float[3][2];
- int two_d_int[4][2];
- char two_d_char[3][5];
Two Dimensional Arrays

- The contents of the cell of the array can be accessed.
- Given:
  - float Table15[3][3]
  - Table15[0][0]
    - 6.1
  - Table15[1][1]
    - 5.5
  - Table15[2][2]
    - 4.9

<table>
<thead>
<tr>
<th>Row</th>
<th>Col 0</th>
<th>Col 1</th>
<th>Col 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6.1</td>
<td>1.2</td>
<td>8.3</td>
</tr>
<tr>
<td>1</td>
<td>7.4</td>
<td>5.5</td>
<td>3.6</td>
</tr>
<tr>
<td>2</td>
<td>2.7</td>
<td>9.8</td>
<td>4.9</td>
</tr>
</tbody>
</table>
Multidimensional arrays have the same behavior as earlier datatypes.

- The default is automatic
- They can be declared static
- They can be local, global, or external global.
• 2-dimensional arrays can be assigned at the time of declaration

```c
static float two_d[3][3]=
{{1.1,2.3,3.5},
 {5.8,8.13,13.21},
 {21.34,34.55,55.89}};
```
Multidimensional Arrays as Parameters to Functions

• If a two-dimensional array is passed to a function, it is not the same as a one-dimensional array.

• A two-dimensional array has to have at least one of its dimensions defined.
Multidimensional Arrays as Parameters to Functions

```c
void TwoDimFunction1(float dataset[][COL], int m)
{
    int i, j;
    for (j=0; j<COL; j++)
    {
        for (i=0; i<m; i++)
            printf("%g ", dataset[i][j]);
        printf("\n");
    }
}

#define COL 3

void main()
{
    float dataarray[3][3]={{1.0,2.0,3.0},
                            {4.0,5.0,6.0},
                            {7.0,8.0,9.0}};

    TwoDimFunction1(dataarray, 2);

    while ('q'!=getchar());
}
```
Multidimensional Arrays as Parameters to Functions

```c
void TwoDimFunction2(float (*dataset) [3], int m, int n)
{
    int i, j;
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            printf("%g ", dataset[i][j]);
        }
        printf("\n");
    }
}

void main()
{
    float dataarray[3][3] = {{1.0, 2.0, 3.0},
                              {4.0, 5.0, 6.0},
                              {7.0, 8.0, 9.0}};
    TwoDimFunction1(dataarray, 3, 3);
    while ('q' != getchar());
}
```
Arrays with more than two dimensions

• A multidimensional array is not a representation of the physical dimensions.

• An array can have three dimensions, four dimensions, five dimensions, depending on how many indexes of information are used to identify a cell.
Arrays with more than two dimensions

float dataarray1[2][2][2] =
{
    {
        {1,2},
        {3,4}
    },
    {
        {5,6},
        {7,8}
    }
};

void ThreeDimFunction1 (float dataset[][2][2],int m,int n,int p)
{
    int i,j,k;

    for (i=0;i<m;i++)
    {
        for (j=0;j<n;j++)
        {
            for (k=0;k<p;k++)
            {
                printf("%g\n",dataset[i][j][k]);
            }
        }
    }
}
int data[2][2][n];
Arrays with more than two dimensions

float dataarray1[2][2][2]=
{
    {
        {1,2},
        {3,4}
    },
    {
        {5,6},
        {7,8}
    }
};

void ThreeDimFunction2(float (*dataset)[2][2][2],int m,int n, int p)
{
    int i,j,k;

    for(i=0;i<m;i++)
        for (j=0;j<n;j++)
            for(k=0;k<p;k++)
            {
                printf("%g\n",dataset[i][j][k]);
            }
}
Arrays of Pointers

• It is possible to make an array of pointers

• char *ptrarray[4];
Double Indirection

• Any variable in C can have a pointer to it.
• A pointer is a variable in C
• Therefore, a pointer can point to another pointer
Double Indirection

- int **ptr; is a pointer to a pointer.
- Also,
  - int ***ptr;
  - int ****ptr;
  - int *****ptr;
  - int ******ptr;
Command Line Parameters

• The main function of a C program is like any other function
• The main has a return type
• The main actually has a parameter list
Command Line Parameters

```c
int main(int argc, char *argv[])
{
    /*Code Here*/
}
```
Command line parameters

- The default return type of the main function is an int
- This value usually comes from the `exit(n)` command
- The default return value of the main function is a 0.
Command line parameters

• The default parameter list for the main function has two types, an int, and a two dimensional array of char.
Command line parameters

• The int parameter
  • usually named with the identifier argc,
  • for “argument count”
  • is a count of how many parameters are called in the main plus one.
Command line parameters

C:\MyProgram A B C D

What is the value of argc?
5
Command line parameters

- The two-dimensional array of characters is actually an array of strings.
- Usually named with the identifier argv, for “argument vector.”
- The command line is parsed by the blank space to make each entry of the argument vector, including the name of the program.
Command line parameters

What is the value of argv?
argv[0] = “MyProgram”
argv[1] = “A”