Arrays

Abdallah Karakra

Computer Science Department

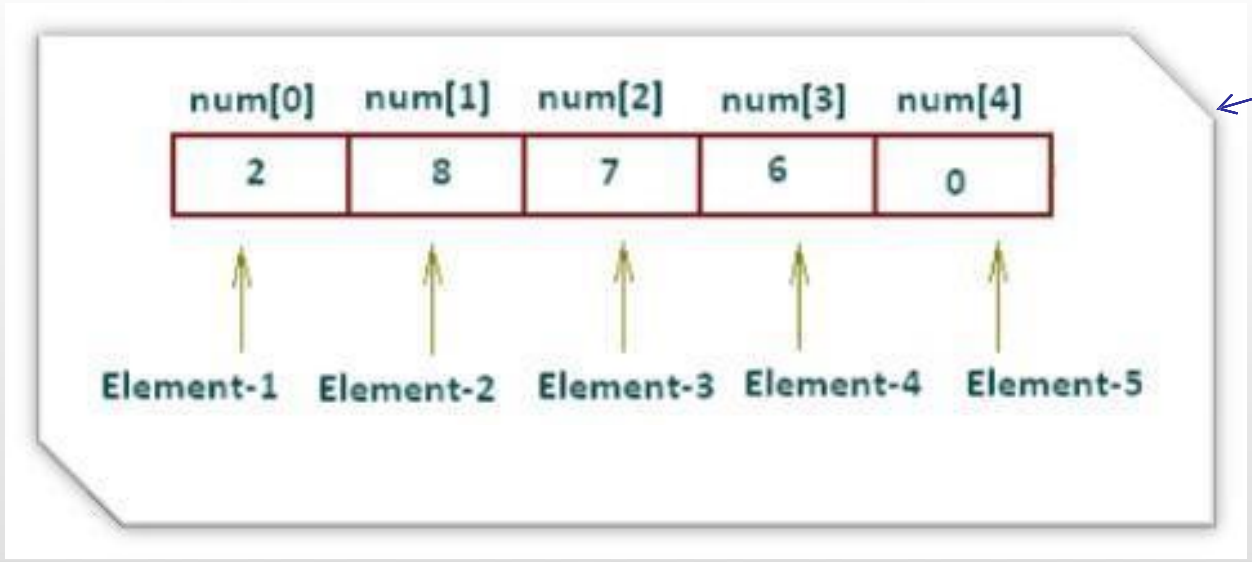
COMP230

Arrays

Array is a collection of data items of the same type.

Array element is a data item that is part of an array.

integers

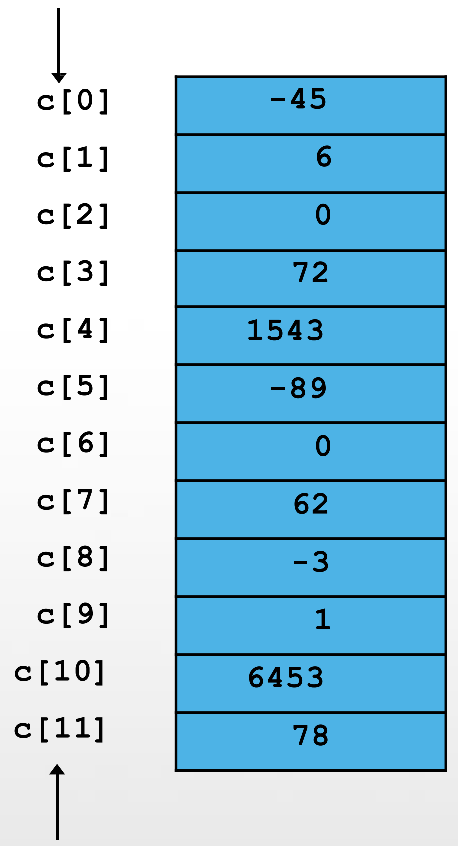


Arrays

- Array
  - Group of consecutive memory locations
  - Same name and type
- To refer to an element, specify
  - Array name
  - Position number
- Format:

    arrayname [*position number*]

  - First element at position 0
  - n element array named c:
    - c[0], c[1]...c[n – 1]



Declaring Arrays

- When declaring arrays, specify

```
arrayType arrayName[numberOfElements ];
```

e.g.
```
int c[ 10 ];
float myArray[ 100 ];
```

- Declaring multiple arrays of same type
  - Format similar to regular variables

e.g.
```
int b[ 100 ], x[ 27 ];
```

Declaring Arrays

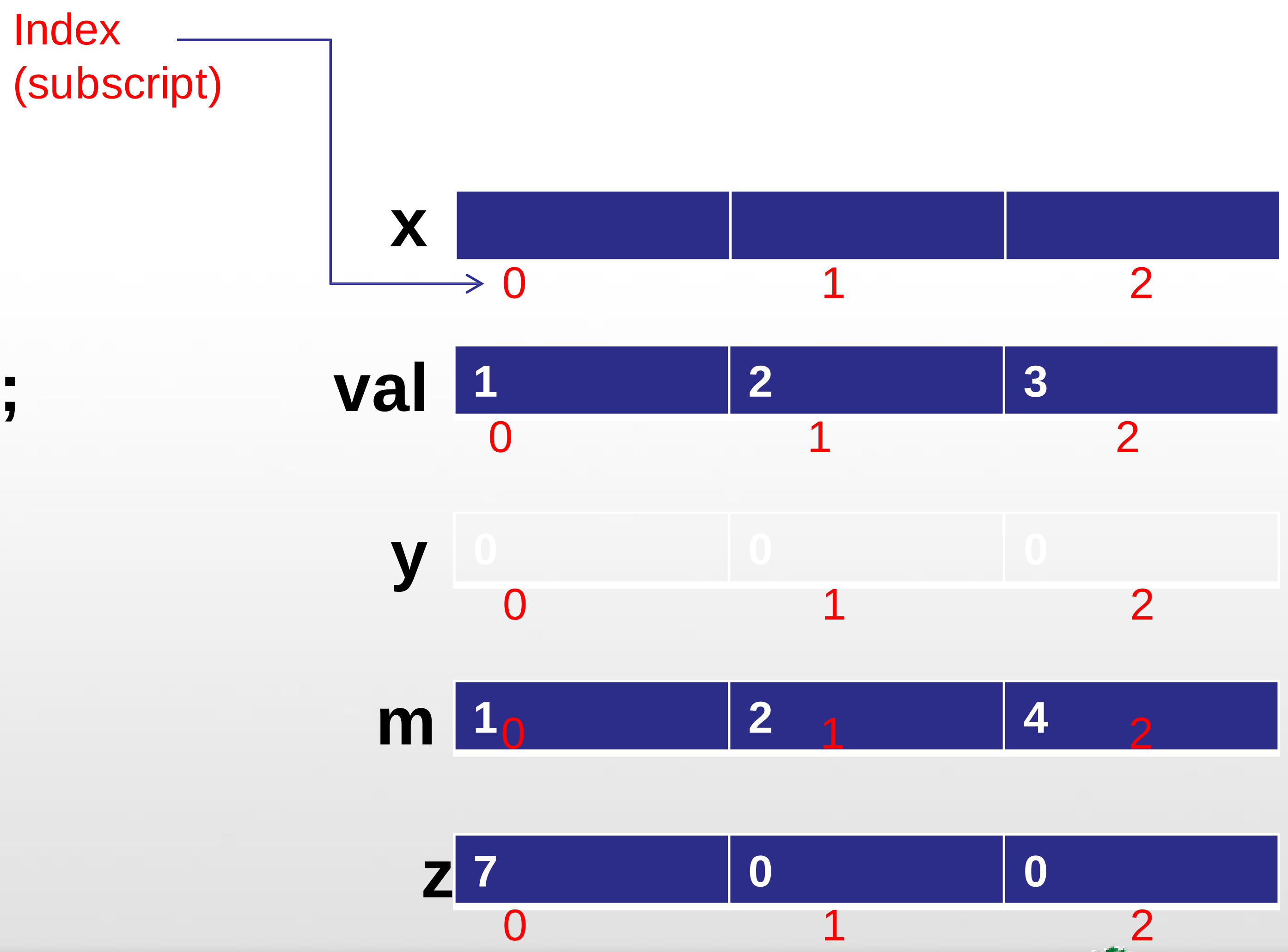
int x [3];

int val[3]={1,2,3};

int y[3]={0};

int m[ ]={1,2,4};

int z[3]={7};



Arrays

Array elements are like normal variables

```
c[ 0 ] =  3;
printf( "%d", c[ 0 ] );
c[1]= c[0]+c[2]
c[3]= c[2]+5
```

Perform operations in subscript (index).

```
c[ 5 - 2 ] == c[ 3 ] == c[ x ]
```

Examples Using Arrays

- Initializers

```
int n[ 5 ] = { 1, 2, 3, 4, 5 };
char alphabet[5] = {‘A’,’B’,’C’,’D’,’E’};
```

- All elements 0

```
int n[ 5 ] = { 0 }
```

- If size omitted, initializers determine it

```
int n[ ] = { 1, 2, 3, 4, 5 };
```

5 initializers, therefore 5 element array

Examples Using Arrays

```
int a [5] = {5,2,9,10,31};
int result = a[3%2] + a[2]+a[4/2];
printf("%d\n",result);
printf("%d",a[5%3]);
```

```
Output:
20
9
```

```
int a [5] = {5,2,9,10,31};
int temp;
printf("%d   %d",a[0], a[4]);
temp=a[0];
a[0]=a[4];
a[4]=temp;
printf("\n%d   %d",a[0], a[4]);
```

```
Output:
5  31
31  5
```

Example: Fill and Print Array

```c
#include <stdio.h>

int main ()
{
   int n[ 10 ];  // n is an array of 10 integers
   int i,j;

    // initialize elements of array n to 0 (Fill Array)
   for ( i = 0; i < 10; i++ )
   {
      n[ i ] = i + 1; /* set element at location i to i + 1 */
   }

    // output each array element's value (Print Array)
   for (j = 0; j < 10; j++ )
   {
      printf("Element[%d] = %d\n", j, n[j] );
   }

   return 0;
}
```

Output:

```
Element[0] = 1
Element[1] = 2
Element[2] = 3
Element[3] = 4
Element[4] = 5
Element[5] = 6
Element[6] = 7
Element[7] = 8
Element[8] = 9
Element[9] = 10
```

Example: Fill and Print Array

```c
#include <stdio.h>
#define size 5 //   array size= 5
int main ()
{
   int n[ size ]; //  n is an array of 5 integers
   int i,j;

   //  initialize elements of array n (Fill Array)
   for ( i = 0; i < size; i++ )
   {
      scanf ("%d",&n[ i ]);
   }

   //  output each array element's value (Print Array)
   for (j = 0; j < size; j++ )
   {
      printf("Element[%d] = %d\n", j, n[j] );
   }

   return 0;
}
```

Input:

```
1 2 3 4 5
```

Output:

```
Element[0] = 1
Element[1] = 2
Element[2] = 3
Element[3] = 4
Element[4] = 5
```

Sending Arrays to Functions

- An array can be seen as a Pointer that points to the first index in the array.
- E.g., int c[12];

- E.g., Write a function that receives and array and initialize its contents to -1.

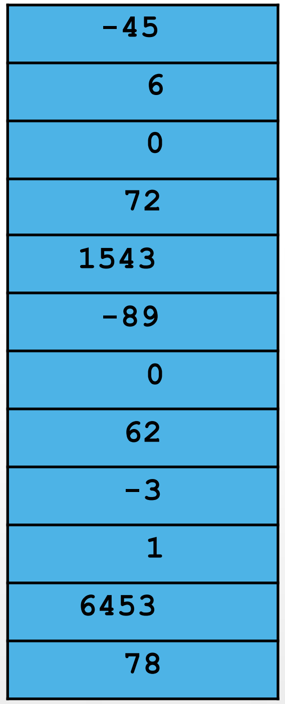
```
#include<stdio.h>

void initialize(int a[], int size);
int main(){
      int array[1000], i;

      initialize(array, 1000);

      for(i=0; i<1000; i++)
            printf("%d\n", array[i]);

      return 0;
}
```



Sending Arrays to Functions

E.g., Write a function that receives and array and initialize its contents to -1.

```
#include<stdio.h>

void initialize(int a[], int size);
int main(){
          int array[1000], i;

          initialize(array, 1000);

          for(i=0; i<1000; i++)
                    printf("%d\n", array[i]);

          return 0;
}

void initialize(int a[], int size){
          int j;

          for(j=0; j<size; j++)
                    a[j] = -1;

}
```

Sending Arrays to Functions

E.g., Write a function that receives an array and print its contents.

```
#include<stdio.h>

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

int main(){
          int array[5]={10, 20, 30, 40, 50}, i;

          printArray (array, 5);


          return 0;
}

void printArray(int a[], int size){
          int j;

          for(j=0; j<size; j++)
                    printf("%d\n", a[j]);

}
```

Sending Arrays to Functions

E.g., Write a function that receives two arrays and put their sum in a third array. Print the sum of arrays in the main.

```
#include<stdio.h>

void sum(int a[], int b[], int c[], int size);
int main(){
          int a1[10], a2[10], result[10], i;

          printf("Enter the content of the first array (10 elements)\n";
          for(i=0; i<10; i++)
                    scanf("%d", a1[i]);

          printf("Enter the content of the secondarray (10 elements)\n";
          for(i=0; i<10; i++)
                    scanf("%d", a2[i]);

          sum(a1, a2, result, 10);

          printf("The sum of array elements:\n";
          for(i=0; i<10; i++)
                    printf("%d + %d = $d\n", a1[i], a2[i], result[i]);


          return 0;
}
```

Sending Arrays to Functions

E.g., Write a function that receives two arrays and put their sum in a third array. Print the sum of arrays in the main.

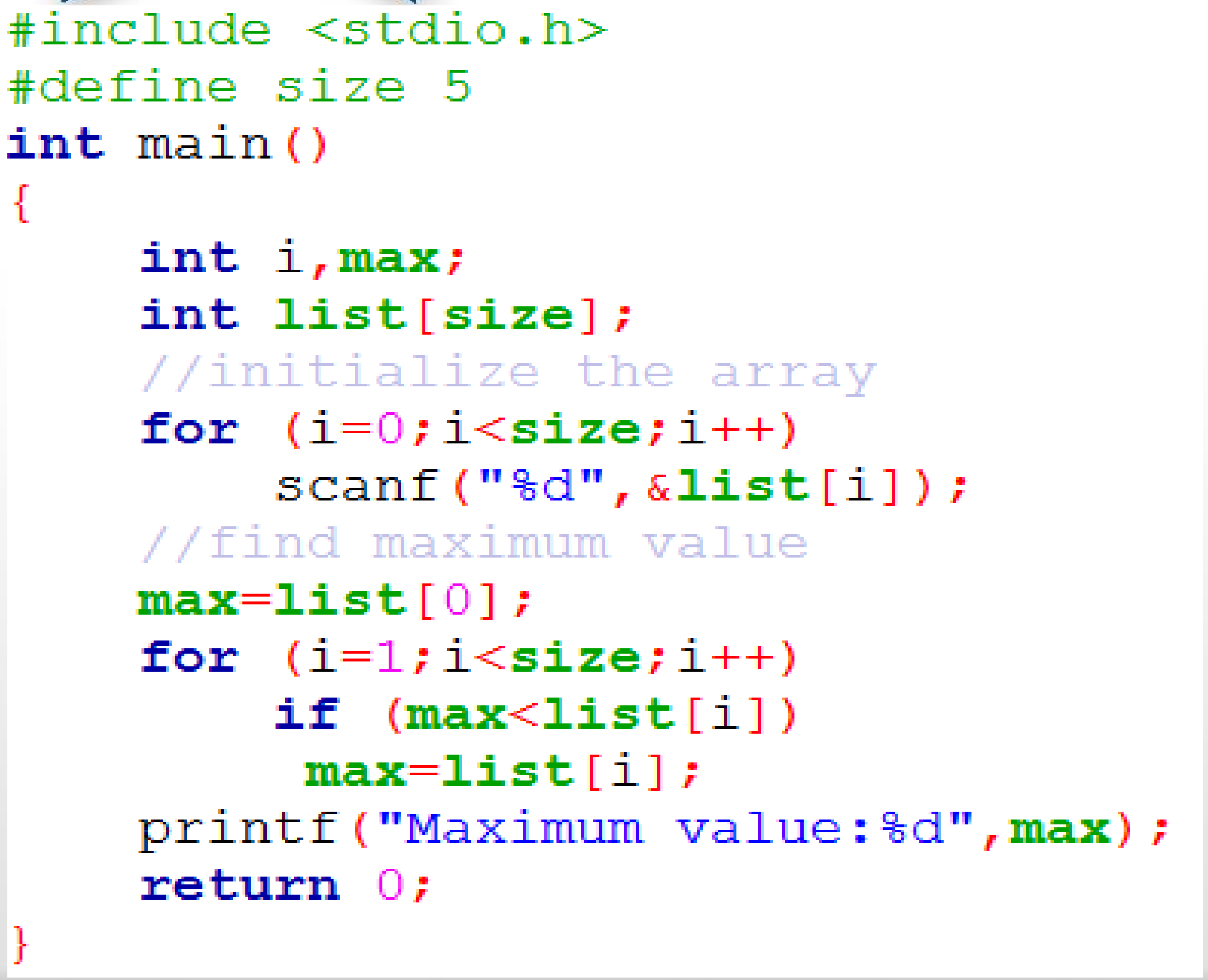
```
#include<stdio.h>

void sum(int a[], int b[], int c[], int size)
{
          int j;

          for(j=0; j<size; j++)
                    c[j] = a[j] + b[j];

}
```

Example: Finding the Maximum

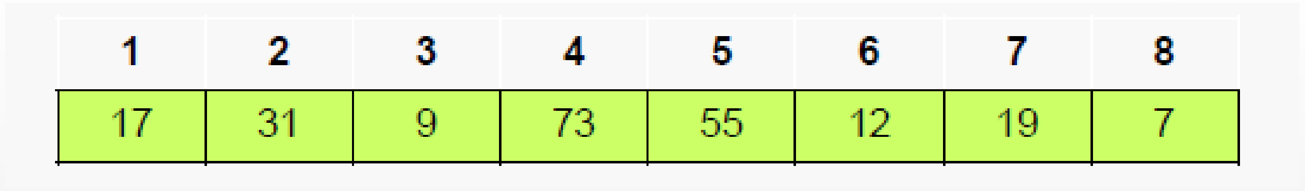


```
#include <stdio.h>
#define size 5
int main()
{
    int i,max;
    int list[size];
    //initialize the array
    for (i=0;i<size;i++)
        scanf("%d",&list[i]);
    //find maximum value
    max=list[0];
    for (i=1;i<size;i++)
        if (max<list[i])
         max=list[i];
    printf("Maximum value:%d",max);
    return 0;
}
```

Linear Search

Problem:
Given a list of N values, determine whether a given value X occurs in the list.

For example, consider the problem of determining whether the value 55 occurs in:



1	2	3	4	5	6	7	8
17	31	9	73	55	12	19	7

Solution:
start at one end of the list,
if the current element doesn't equal the search target, move to the next element,
stopping when a match is found or the opposite end of the list is reached.

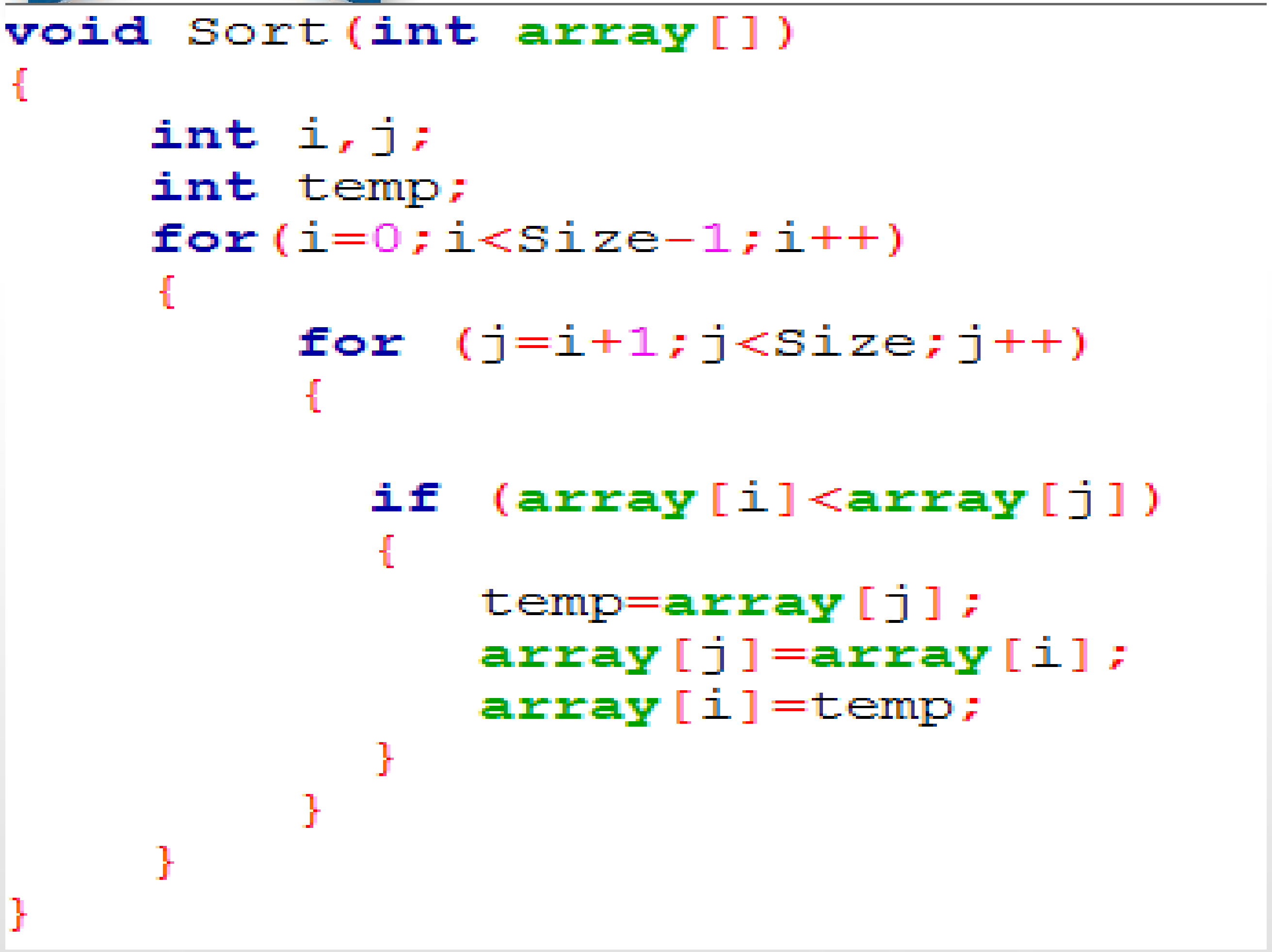
Example

Write a linear search function that receives an array and an integer to search for. Return the index of the element if found or -1 if the element does not exist.

Exercise

Write a program that takes 7 integers as input and prints the number with the smallest sum of digits and its location in the array.

Example: sorting it in descending order



```
void Sort(int array[])
{
     int i,j;
     int temp;
     for(i=0;i<Size-1;i++)
     {
          for (j=i+1;j<Size;j++)
          {

             if (array[i]<array[j])
             {
                 temp=array[j];
                 array[j]=array[i];
                 array[i]=temp;
             }
          }
     }
}
```

```
Enter array of integers with size 3
3 4 5
array after sorted :5 4 3
```

Creating a 2D Array

Create array elements by telling how many ROWS and COLUMNS

Example:

```
int grades[5][3];
```

grades is a two-dimensional array, with 5 rows and 3 columns.
One row for each student. One column for each test.

Example

```
int a[2][4];

a[1][0]=9;

a[0][3]=5;

a[0][1]=a[0][3]+ a[1][0];
```

	0	1	2	3
0		14		5
1	9			

Declare & Initialize

Example:

```
int grades[5][3] =
  { { 78, 83, 82 },
    { 90, 88, 94 },
    { 71, 73, 78 },
    { 97, 96, 95 },
    { 89, 93, 90 } };
```

A Two-D Array is an array of arrays.

Each row is itself a One-D array.

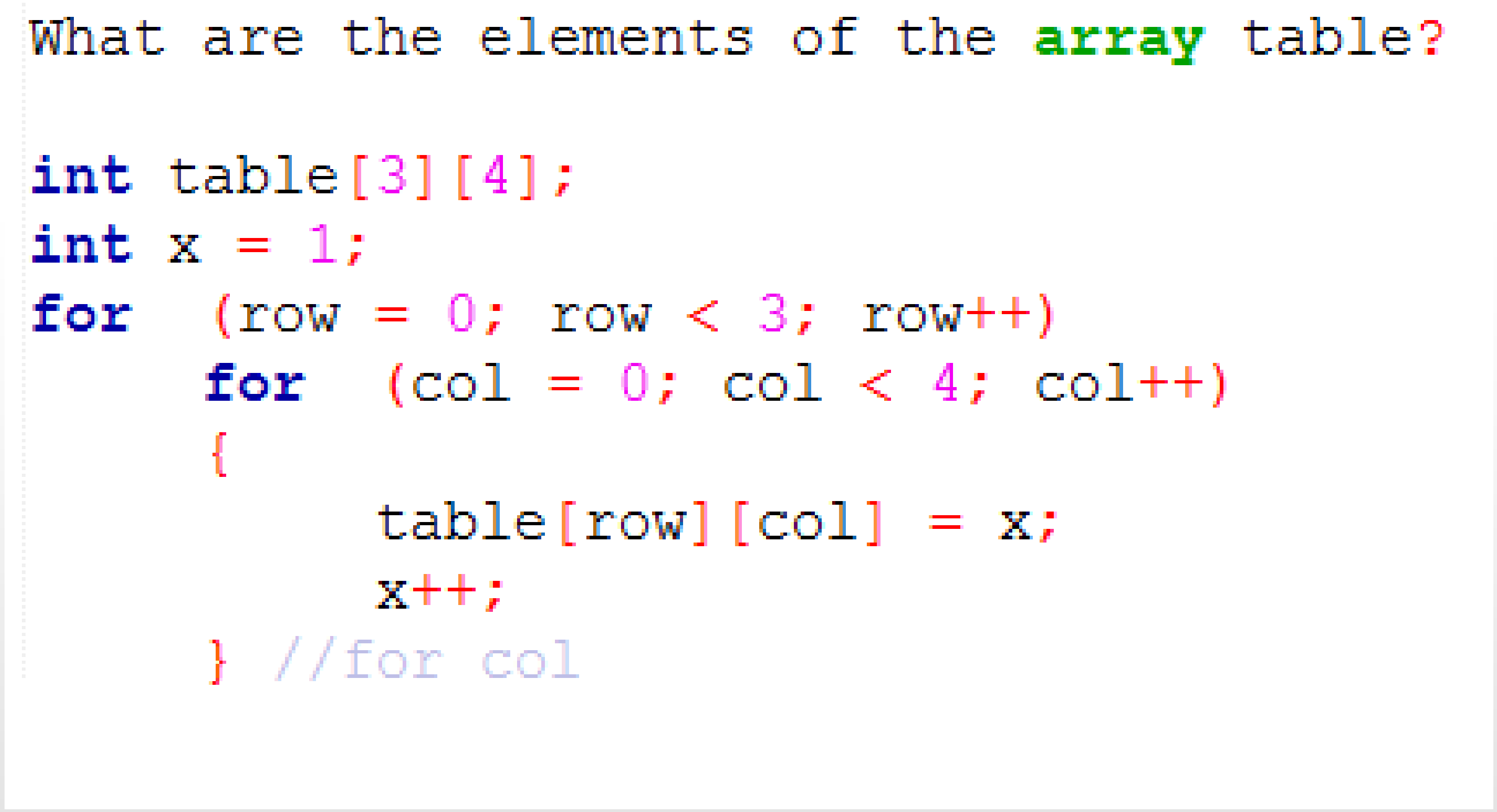
Row, Column Indices

	0	1	**2**
0	78	83	82
1	90	88	94
2	71	73	78
3	97	96	**95**
4	89	93	90

Give both the ROW and COLUMN indices to pick out an individual element.

The fourth student's third test score is at ROW 3, COLUMN 2

Example : Fill Array



What are the elements of the array table?

```
int table[3][4];
int x = 1;
for  (row = 0; row < 3; row++)
     for  (col = 0; col < 4; col++)
     {
          table[row][col] = x;
          x++;
     } //for col
```

Example

Write a program that adds up two 2x2 arrays and stores the sum in third array.

```
#include <stdio.h>
#define rows 2
#define cols 4

void add(int A[][cols], int B[][cols], int C[][cols]) ;

int main()
{
    int A[rows][cols] = { {1, 1, 1, 1},  {2, 2, 2, 2}};
    int B[rows][cols] = { {3, 3, 3, 3}, {4, 4, 4, 4}};

    int C[rows][cols]; // to store result
    int i, j;
    add(A, B, C);

    printf("Result matrix is \n");
    for (i = 0; i < rows; i++)
        for (j = 0; j < cols; j++)
           printf("%d\t", C[i][j]);

    return 0;
}
```

Example

Write a program that adds up two 2x2 arrays and stores the sum in third array.

```
// This function adds A[][] and B[][], and stores
// the result in C[][]
void add(int A[][cols], int B[][cols], int C[][cols])
{
    int i, j;
    for (i = 0; i < rows; i++)
        for (j = 0; j < cols; j++)
            C[i][j] = A[i][j] + B[i][j];
}
```

Iterating through a file with multiple lines

```
#include<stdio.h>
int main()
{
    FILE *fp = fopen("input.txt", "r");
    int number;
    int status = fscanf(fp, "%d", &number);

    while (status != EOF){
        /* display contents of file on screen */
        printf("number is: %d\n", number);

        status = fscanf(fp, "%d", &number);
    }

    fclose(fp);

    return 0;
}
```

```
1
2
3
4
```

input.txt