Universiti Malaysia PAHANG
Engineering • Technology • Creativity

Technical Informatics I
Arrays

by
Dr. Fatimah
Faculty of Mechanical Engineering
fatimahd@ump.edu.my

Arrays

- Aims
  - Introduce the concept of one-dimensional arrays and multi-dimensional array

- Expected Outcomes
  - Students are able to construct and implement numeric arrays in their C programs

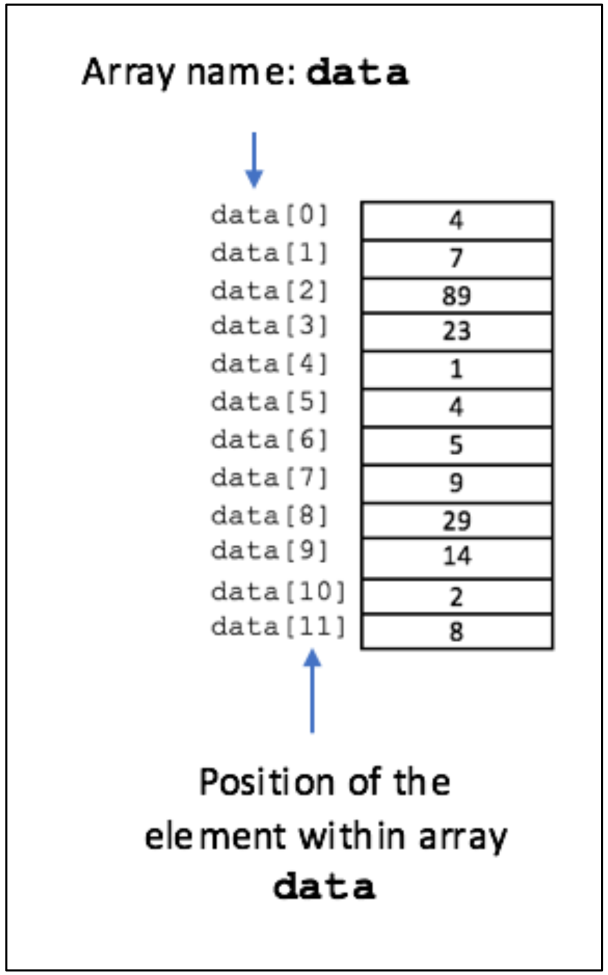
- References
  - Harry H. Cheng, 2010. C for Engineers and Scientists: An Interpretive Approach, McGraw Hill

Content

- Introduction to Arrays
- Initializing arrays
- Arrays in standard functions
- Multi-dimensional arrays

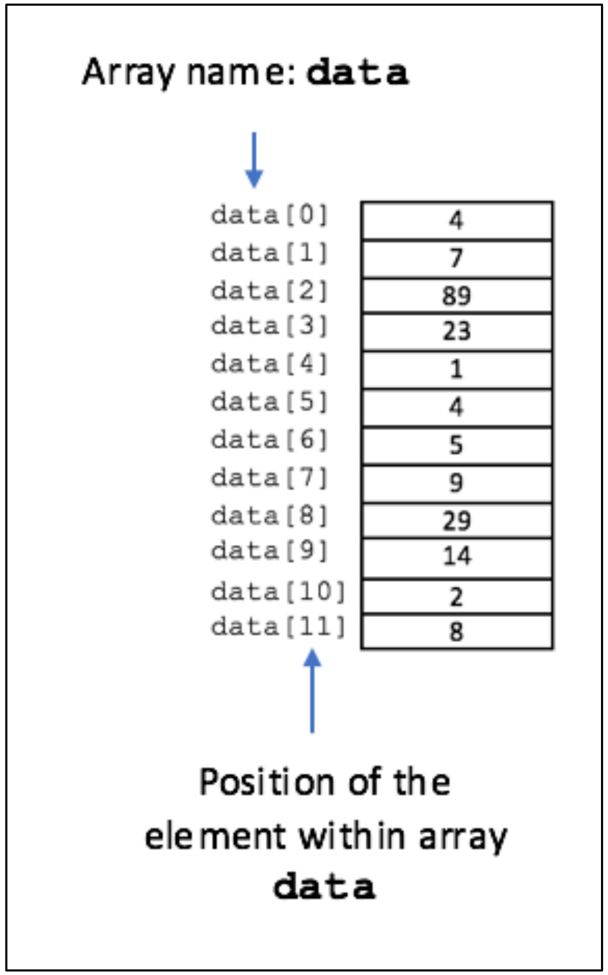
Introduction to Arrays

- Arrays
  - Structures of related data items
  - Static entity – same size throughout program
  - Conceptually similar to matrices in maths



Definitions Related to Arrays

- **Array Rank**: The number of dimensions in an array
- **Array extent**: The number of elements in each dimension
- **Array shape**: vector where each element of the vector is the extent in the corresponding dimension of the array.
- **Array size**: number of bytes used to store the total number of elements of the array.
- For example, for array: `int a[6][5];`
  - Array Rank is 2,
  - Array extent corresponding to the first and second dimensions are 6 and 5, respectively.
  - Array shape is a vector with two elements 6 and 5 as [6, 5],
  - Array size is `sizeof(int)*6*4 = 4*6*5=120` bytes.



Initializing and declaring arrays: One dimensional array

- Declaration of one dimensional arrays:

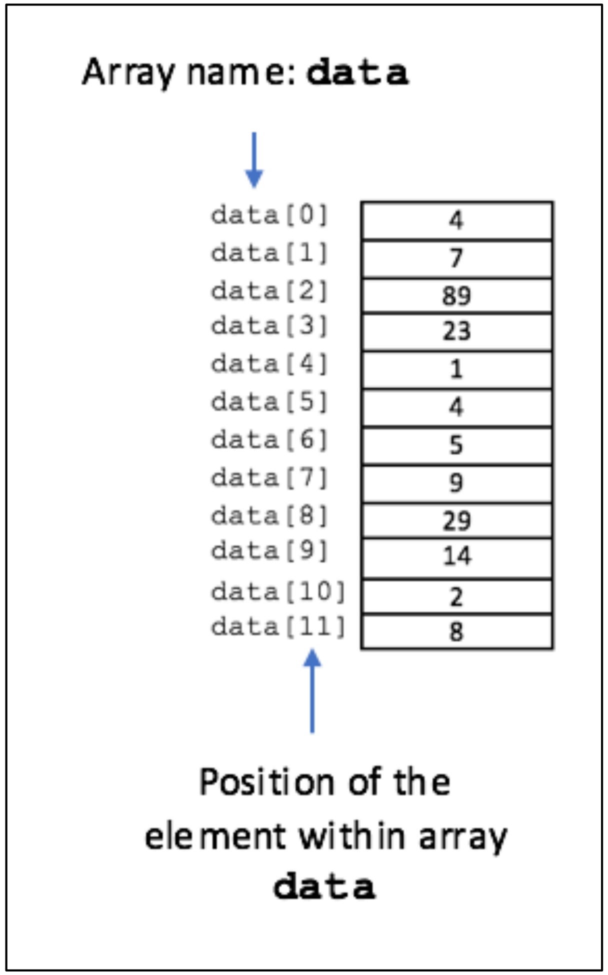
```
type name[expr];
```

  - `type` is a data type
  - `name` is an identifier
  - `expr` number of elements of the array.

- For example, the one dimensional array:

```
int data[12];
```

  - The name of array is `data`.
  - The data type of array elements is `int`.
  - Array a has 12 elements from `data[0]` to `data[11]`.



Initializing and declaring arrays: one dimensional array

1. int a[5] = {89, 34, 12, 5, 78};
   - If not enough initializers, rightmost elements become 0
   - If too many, the result is a syntax error
2. int a[5] = {0};
   - All 5-elements will contain 0
3. int a[] = {89, 34, 12, 5, 78};
   - If the size is omitted, the initializers will determine it
   - Here, it is a 5 element array

Initializing and declaring arrays: two dimensional array

- Using following format to declare a two-dimensional array

```
type name[expr1][expr2];
```

  - `expr1`: number of rows of the array
  - `expr2` : number of columns of the array.
  -

- For example,

```
int data[12][34];
```

  - declares a two-dimensional array. Array a has 12 rows and 36 columns with
  - Number of elements = 12*36 = 432

Initializing and declaring arrays: two dimensional array

1. int a[2][2] = {{11, 32 },
                 { 39, 44 } };
   - Initializers grouped by row in braces
   - a[0][0] = 11, a[0][1] = 32, a[1][0] = 39, a[1][1] = 44

2. int b[2][2] = {{11},
                 {31, 41 } };
   - If not enough, unspecified elements set to zero
   - b[0][0] = 11, b[0][1] = 0, b[1][0] = 31, b[1][1] = 41

4. int c[2][2] = {15, 25, 35, 45};
   - c[0][0] = 15, c[0][1] = 25, c[1][0] = 35, c[1][1] = 45

One dimensional array

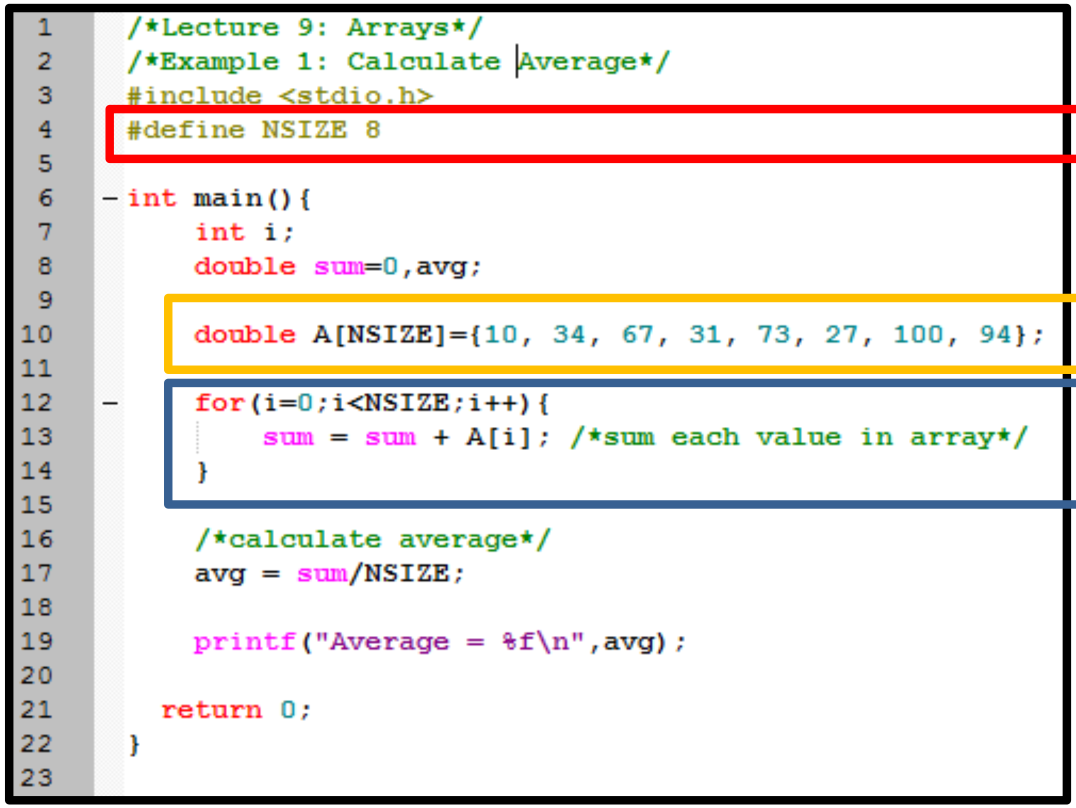
Example 1:

Calculate the average of the number in the array:

A={10, 34, 67, 31, 73, 27, 100, 94}

One dimensional array

Example 1:



```
/*Lecture 9: Arrays*/
/*Example 1: Calculate Average*/
#include <stdio.h>
#define NSIZE 8

int main(){
    int i;
    double sum=0,avg;

    double A[NSIZE]={10, 34, 67, 31, 73, 27, 100, 94};

    for(i=0;i<NSIZE;i++){
        sum = sum + A[i]; /*sum each value in array*/
    }

    /*calculate average*/
    avg = sum/NSIZE;

    printf("Average = %f\n",avg);

  return 0;
}
```

Number of elements in array

Initialize array

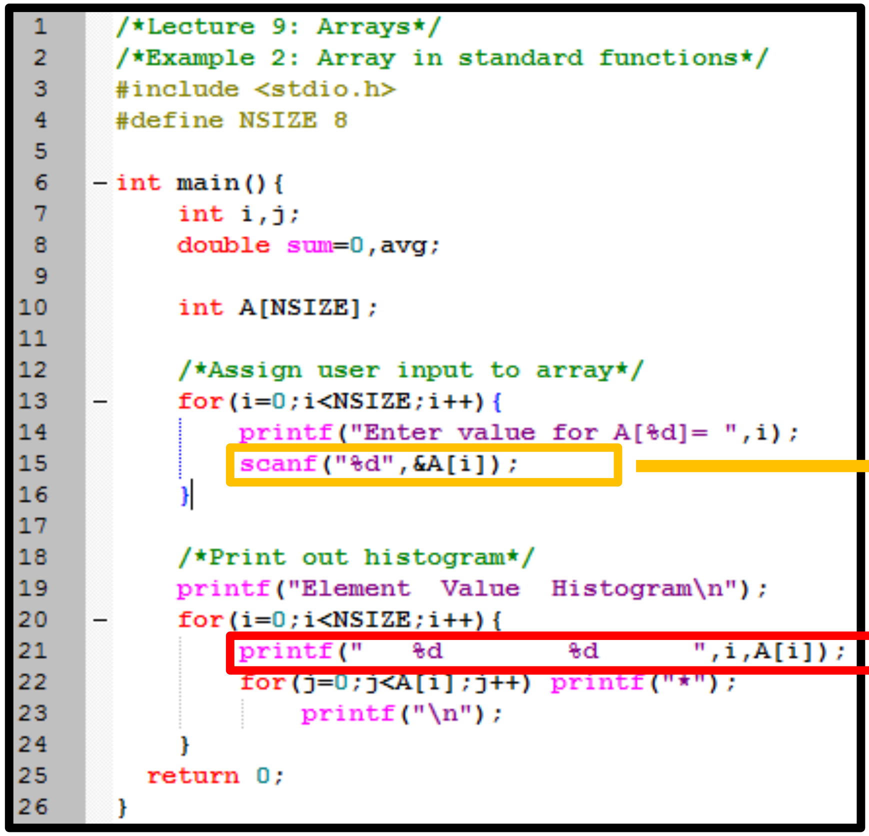
Loop

Arrays in standard functions

- Function `printf()`
  - Must print out each elements one by one
  - Example: `printf(“Value of %d”, A[i]);`
  - **NOT:** `printf(“Value of %d”, A);`
- Function `scanf()`
  - Must scan and assign to array elements one by one
  - Example: `scanf(“%d”, &A[i]);`
  - **NOT**: `scanf(“%d”, &A);`

Arrays in standard functions

Example 2:

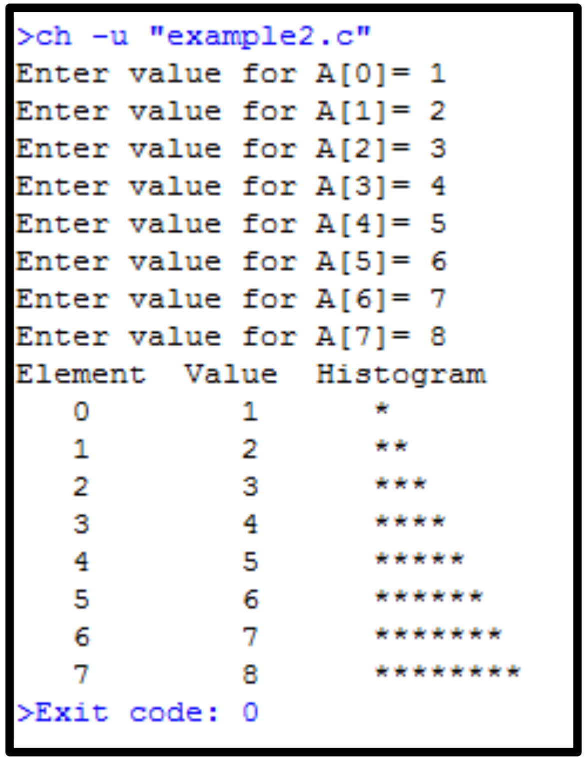


```c
/*Lecture 9: Arrays*/
/*Example 2: Array in standard functions*/
#include <stdio.h>
#define NSIZE 8

int main(){
    int i,j;
    double sum=0,avg;

    int A[NSIZE];

    /*Assign user input to array*/
    for(i=0;i<NSIZE;i++){
        printf("Enter value for A[%d]= ",i);
        scanf("%d",&A[i]);
    }

    /*Print out histogram*/
    printf("Element  Value  Histogram\n");
    for(i=0;i<NSIZE;i++){
        printf("    %d         %d        ",i,A[i]);
        for(j=0;j<A[i];j++) printf("*");
            printf("\n");
    }
  return 0;
}
```

scanf

printf

Output:



```
>ch -u "example2.c"
Enter value for A[0]= 1
Enter value for A[1]= 2
Enter value for A[2]= 3
Enter value for A[3]= 4
Enter value for A[4]= 5
Enter value for A[5]= 6
Enter value for A[6]= 7
Enter value for A[7]= 8
Element  Value  Histogram
   0       1       *
   1       2       **
   2       3       ***
   3       4       ****
   4       5       *****
   5       6       ******
   6       7       *******
   7       8       ********
>Exit code: 0
```

Technical Informatics I

Lecture 9

Dr Fatimah