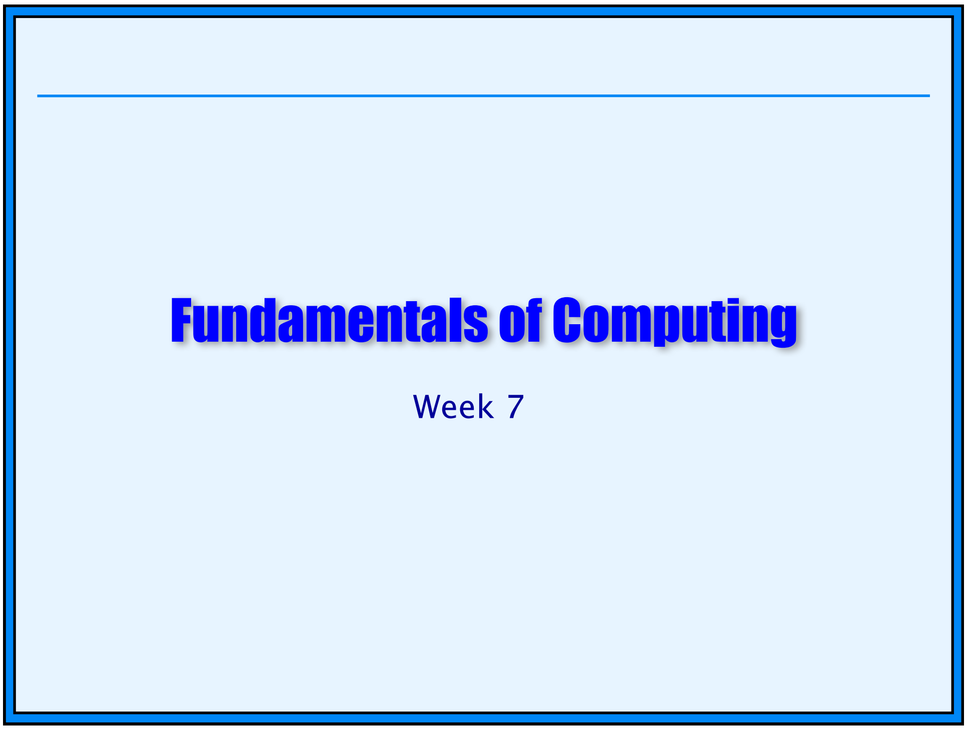
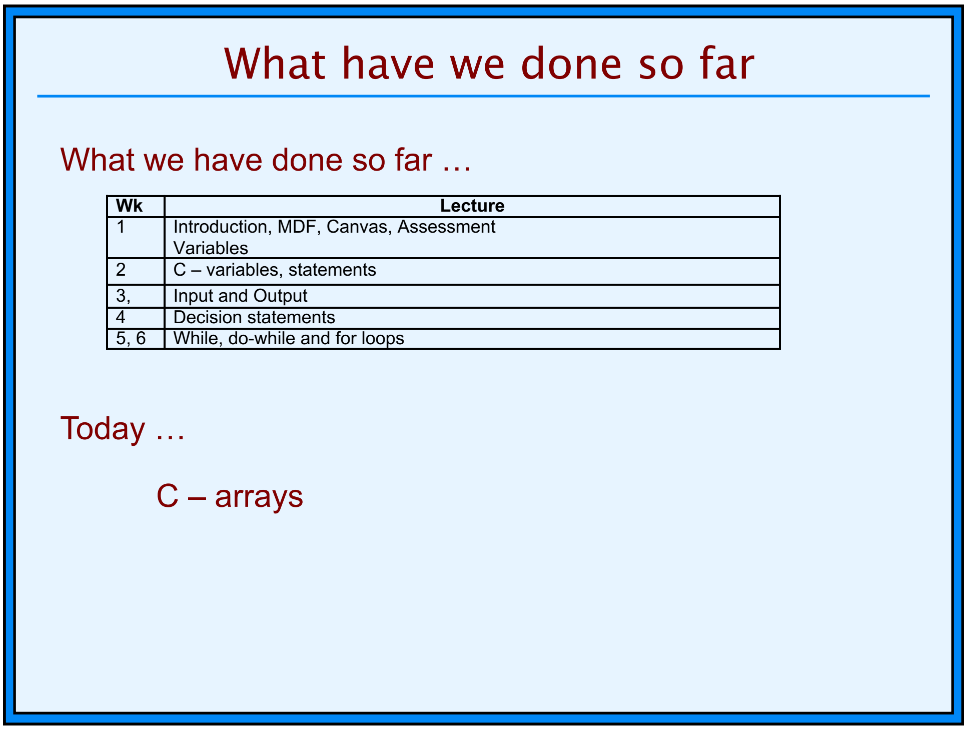


Fundamentals of Computing

Week 7



What have we done so far

What we have done so far …

Wk	Lecture
1	Introduction, MDF, Canvas, Assessment Variables
2	C – variables, statements
3,	Input and Output
4	Decision statements
5, 6	While, do-while and for loops

Today …

C – arrays

Learning Outcomes

1. Learn about arrays and their importance.
2. Learn how and when to use arrays.
3. Be able to write programs using arrays to store data.

Array – Definition

An array in C / C++ is a collection of logically related data elements of the same type that are referenced by a common name.

All the elements of an array occupy a set of contiguous memory locations and, by using an index, we can identify each element.

Array – Definition

Example

We have 100 marks of integer type

- declare them by using simple variables:

```
        int  mark1, mark2, ..., mark100;
```

- declare them using an array:

```
        int  mark[100];
```

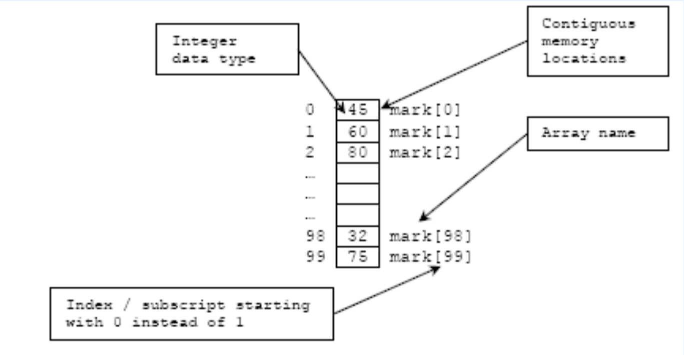
This will reserve 100 contiguous/sequential memory locations for storing the integer data type.

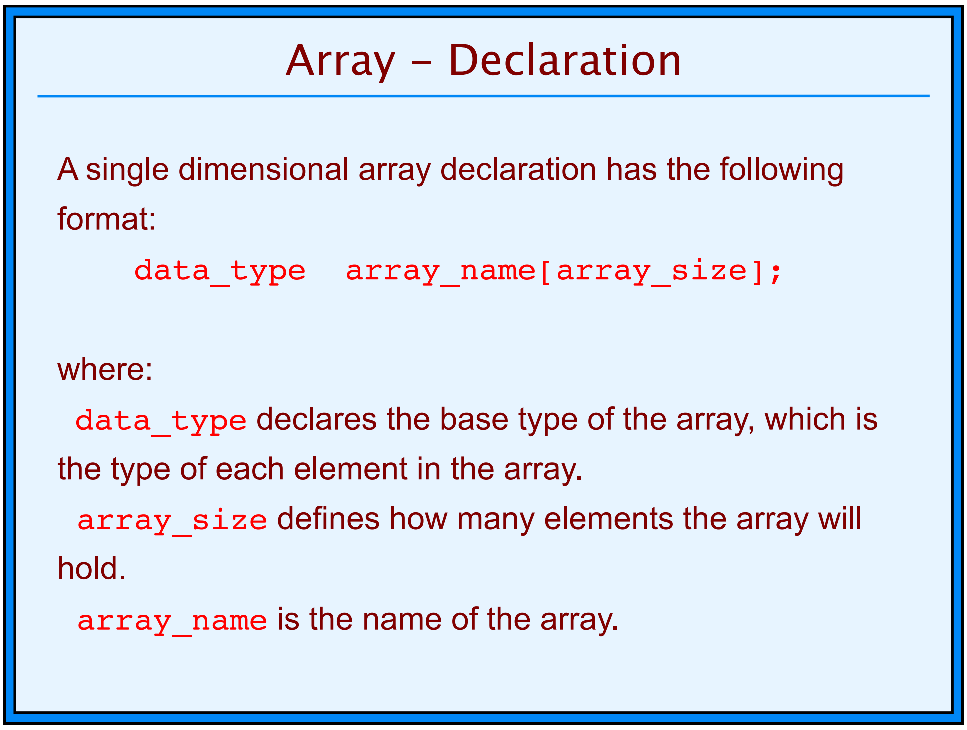
Array – Definition

Example

We have 100 marks of integer type

- declare them using an array:





Array – Declaration

A single dimensional array declaration has the following format:

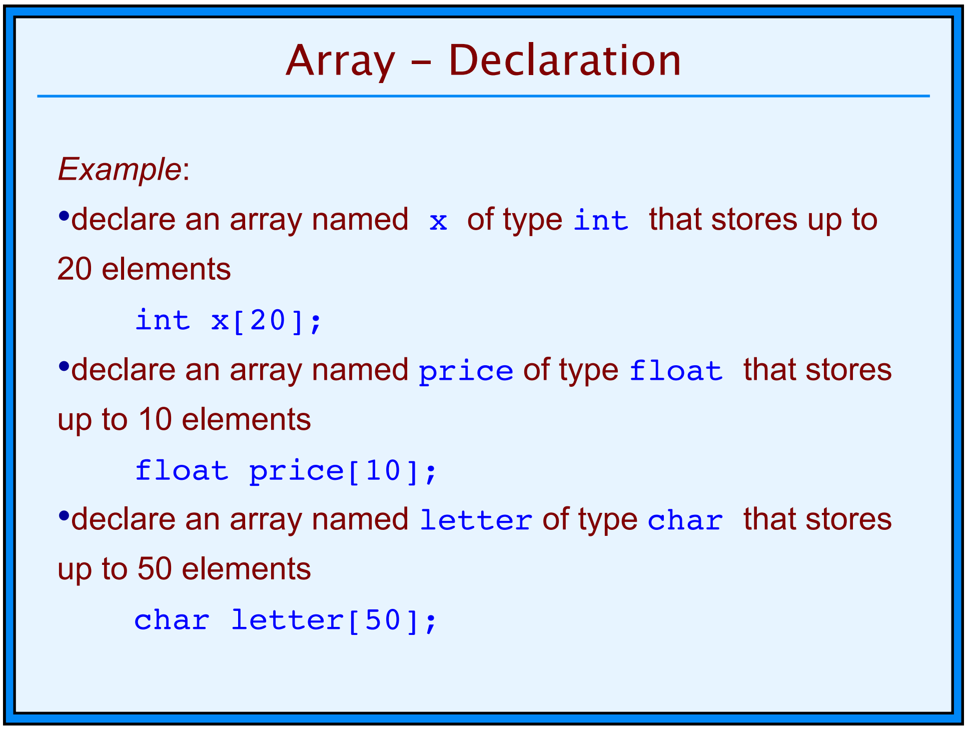
```
data_type   array_name[array_size];
```

where:

`data_type` declares the base type of the array, which is the type of each element in the array.

`array_size` defines how many elements the array will hold.

`array_name` is the name of the array.



Array – Declaration

Example:

- declare an array named `x` of type `int` that stores up to 20 elements

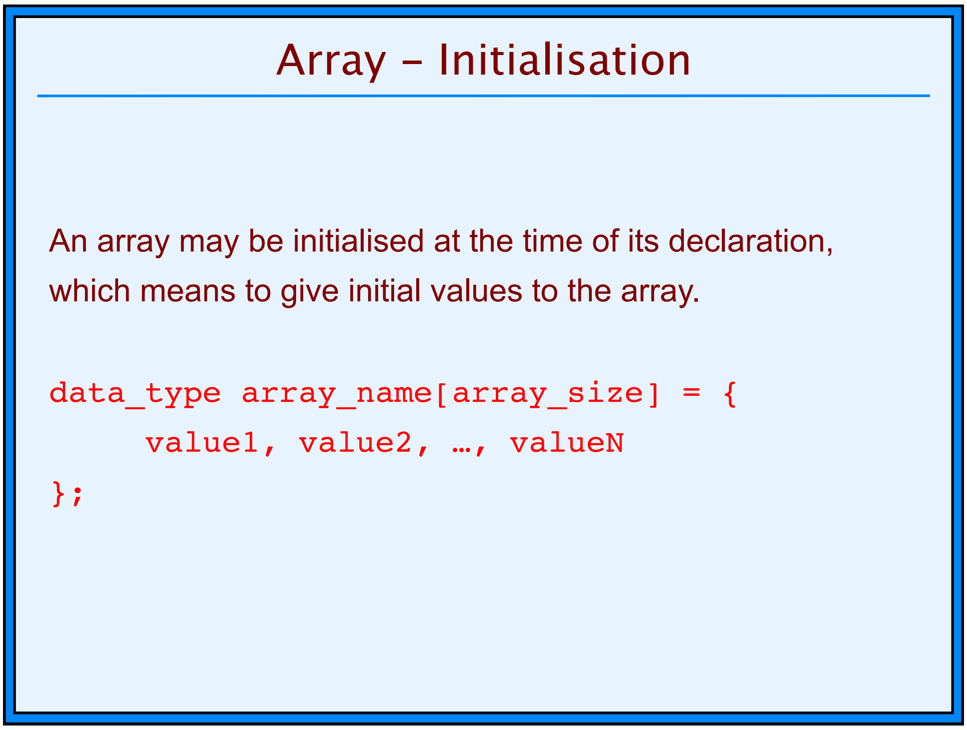
```
int x[20];
```

- declare an array named `price` of type `float` that stores up to 10 elements

```
float price[10];
```

- declare an array named `letter` of type `char` that stores up to 50 elements

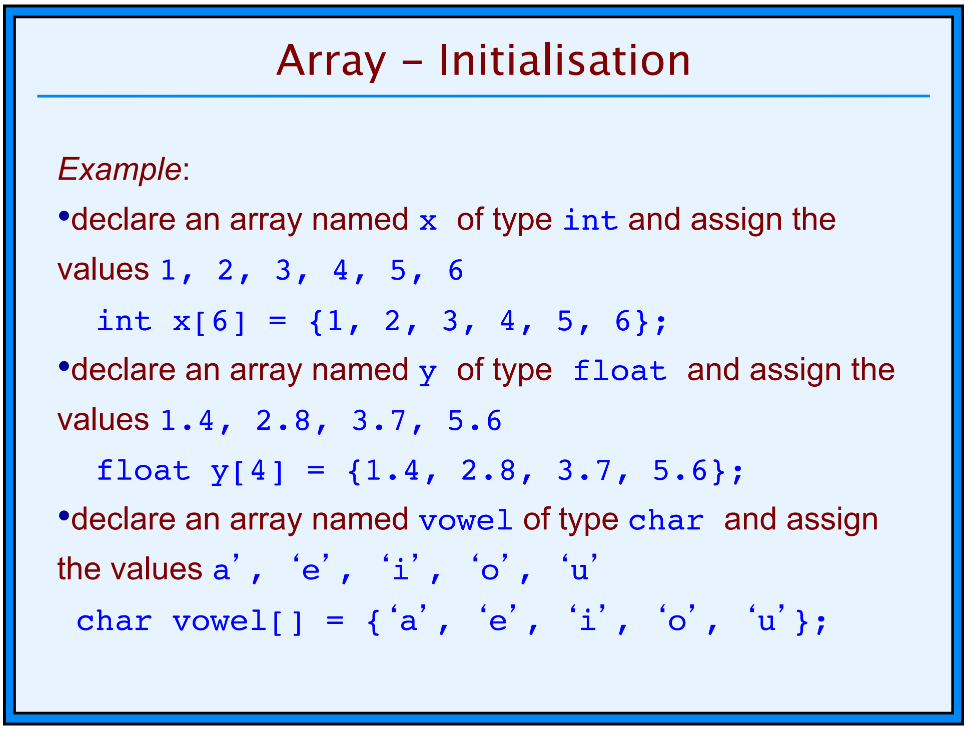
```
char letter[50];
```



Array – Initialisation

An array may be initialised at the time of its declaration, which means to give initial values to the array.

```
data_type array_name[array_size] = {
     value1, value2, …, valueN
};
```



Array – Initialisation

Example:

- declare an array named `x` of type `int` and assign the values `1, 2, 3, 4, 5, 6`

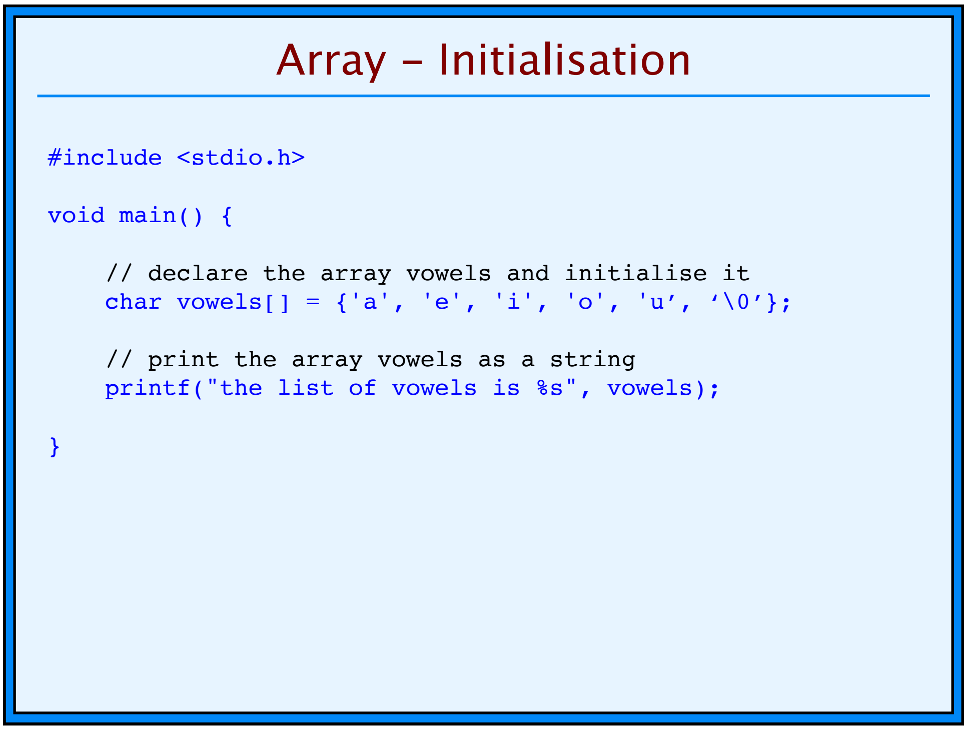
```
  int x[6] = {1, 2, 3, 4, 5, 6};
```

- declare an array named `y` of type `float` and assign the values `1.4, 2.8, 3.7, 5.6`

```
  float y[4] = {1.4, 2.8, 3.7, 5.6};
```

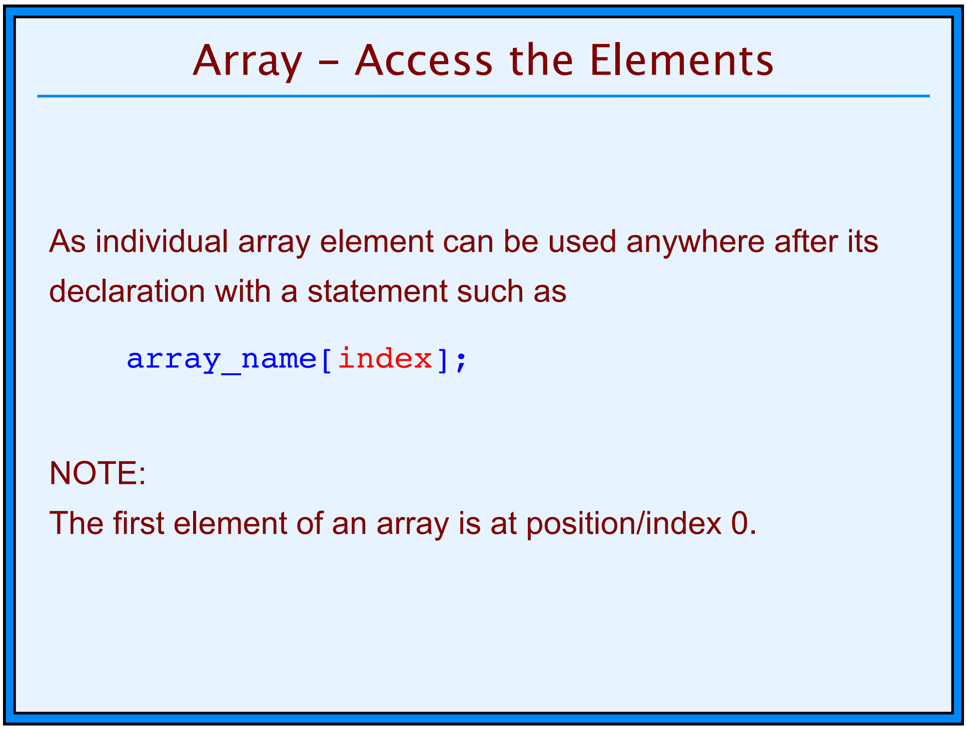
- declare an array named `vowel` of type `char` and assign the values `a’, ‘e’, ‘i’, ‘o’, ‘u’`

```
 char vowel[] = {‘a’, ‘e’, ‘i’, ‘o’, ‘u’};
```



Array – Initialisation

```
#include <stdio.h>

void main() {

    // declare the array vowels and initialise it
    char vowels[] = {'a', 'e', 'i', 'o', 'u’, ‘\0’};

    // print the array vowels as a string
    printf("the list of vowels is %s", vowels);

}
```

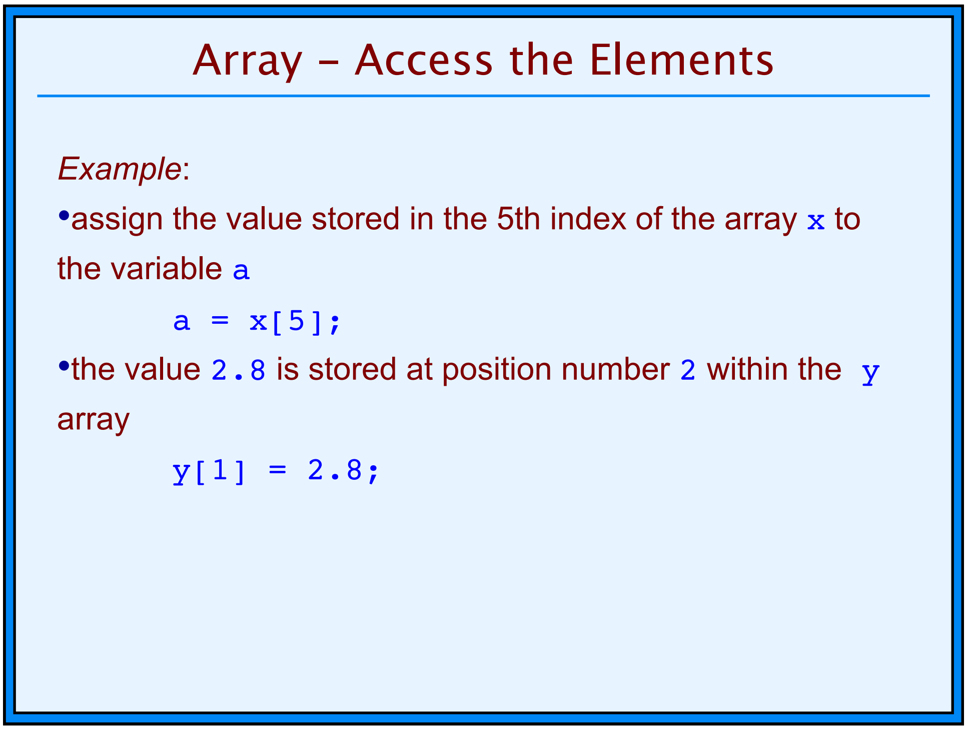


Array – Access the Elements

As individual array element can be used anywhere after its declaration with a statement such as

```
    array_name[index];
```

NOTE:
The first element of an array is at position/index 0.



Array – Access the Elements

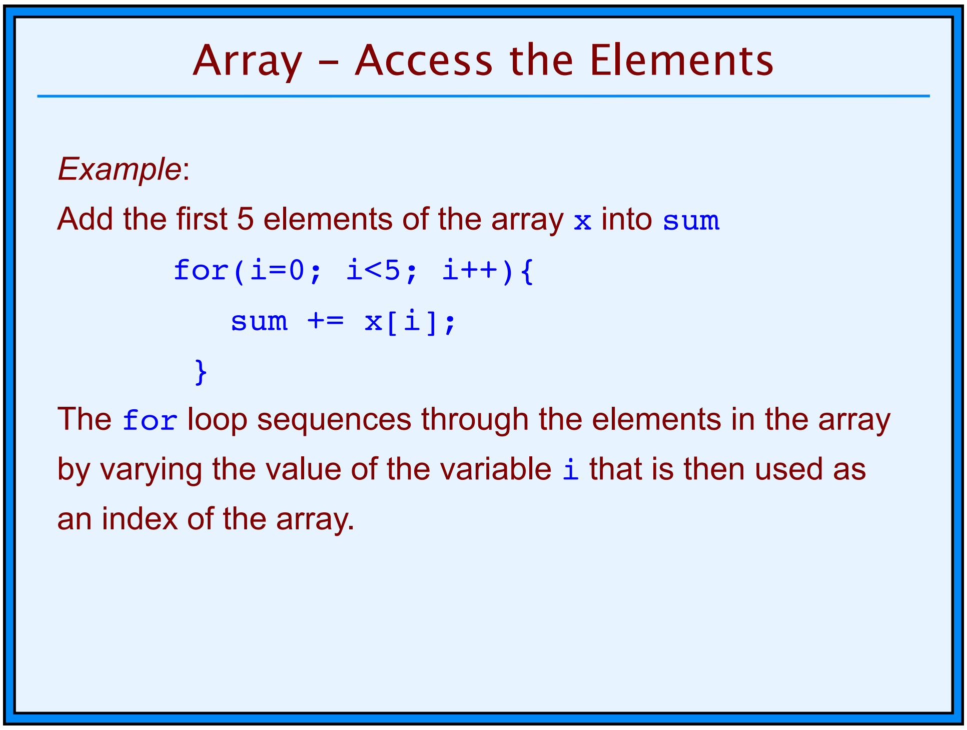
Example:

- assign the value stored in the 5th index of the array `x` to the variable `a`

```
        a = x[5];
```

- the value `2.8` is stored at position number `2` within the `y` array

```
        y[1] = 2.8;
```



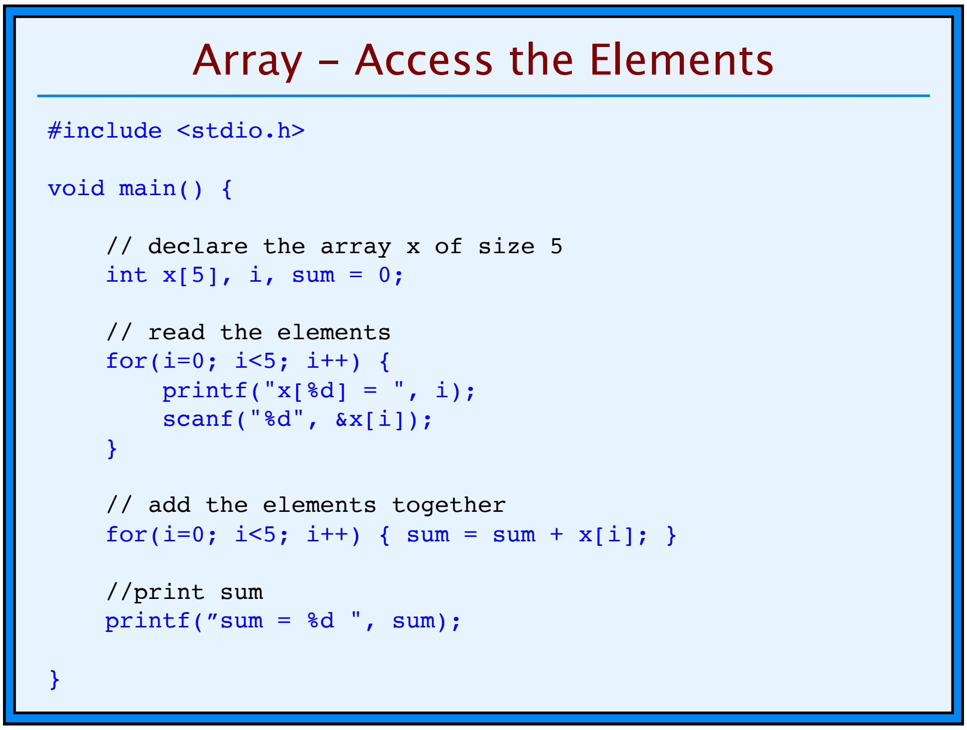
Array – Access the Elements

Example:

Add the first 5 elements of the array `x` into `sum`

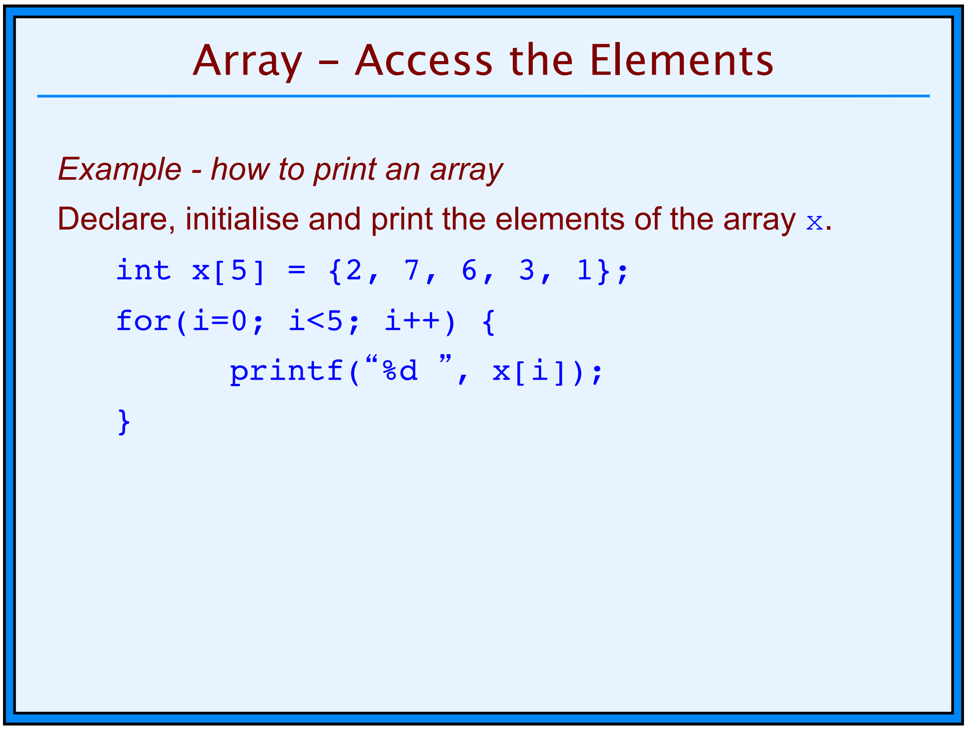
```
        for(i=0; i<5; i++){
            sum += x[i];
         }
```

The `for` loop sequences through the elements in the array by varying the value of the variable `i` that is then used as an index of the array.



Array – Access the Elements

```
#include <stdio.h>

void main() {

    // declare the array x of size 5
    int x[5], i, sum = 0;

    // read the elements
    for(i=0; i<5; i++) {
        printf("x[%d] = ", i);
        scanf("%d", &x[i]);
    }

    // add the elements together
    for(i=0; i<5; i++) { sum = sum + x[i]; }

    //print sum
    printf(”sum = %d ", sum);

}
```

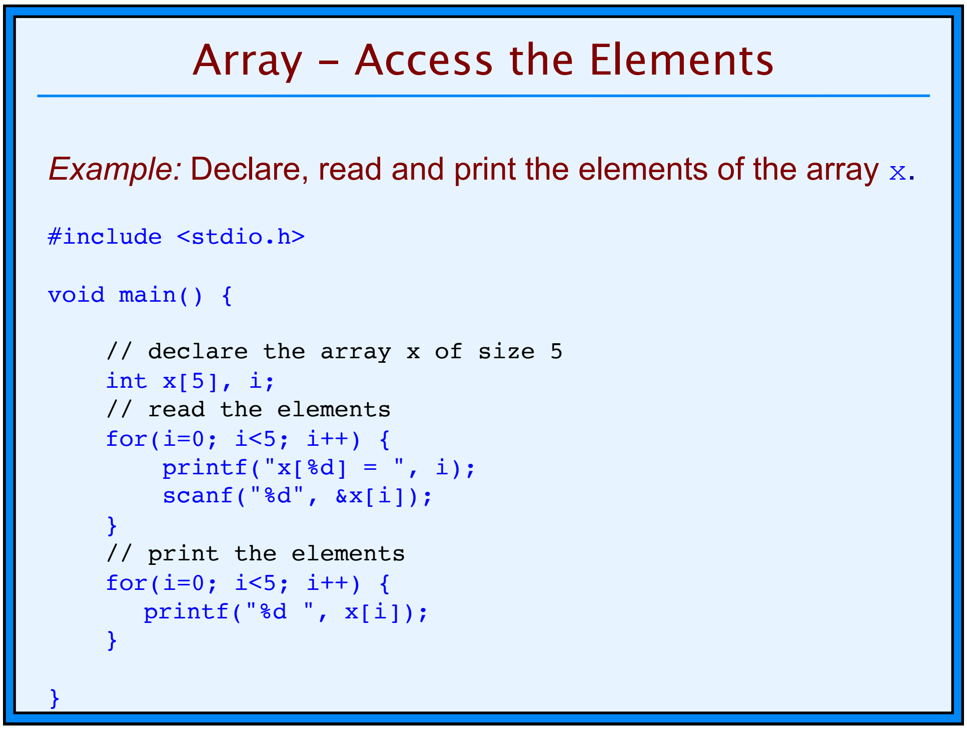


Array – Access the Elements

Example - how to print an array

Declare, initialise and print the elements of the array `x`.

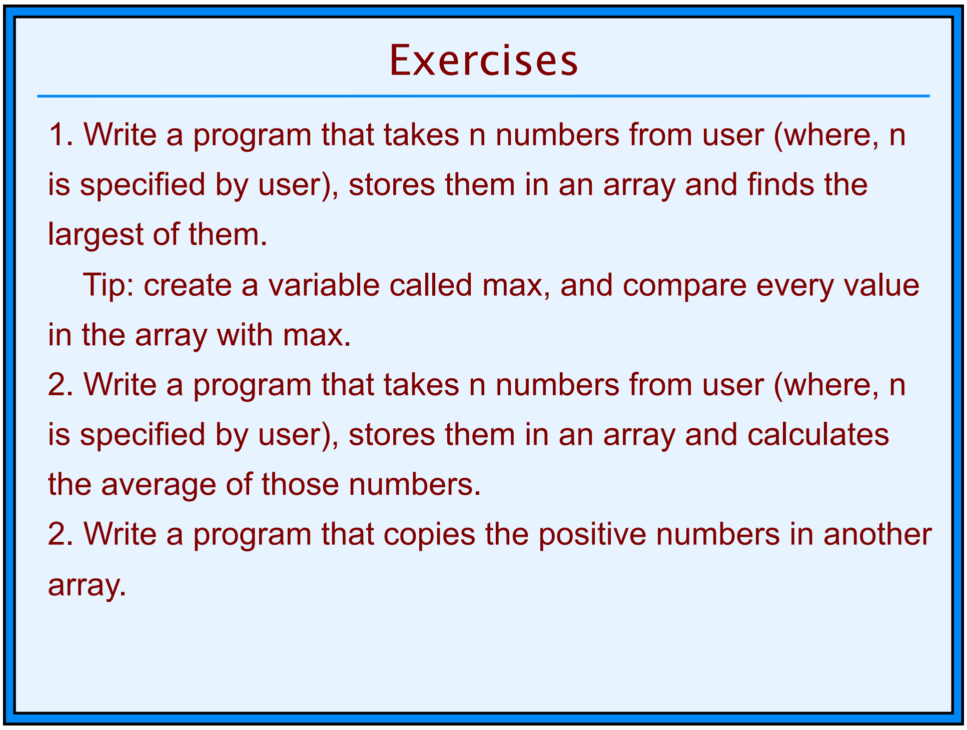
```
int x[5] = {2, 7, 6, 3, 1};
for(i=0; i<5; i++) {
        printf(“%d ”, x[i]);
}
```



Array – Access the Elements

Example: Declare, read and print the elements of the array `x`.

```
#include <stdio.h>

void main() {

    // declare the array x of size 5
    int x[5], i;
    // read the elements
    for(i=0; i<5; i++) {
        printf("x[%d] = ", i);
        scanf("%d", &x[i]);
    }
    // print the elements
    for(i=0; i<5; i++) {
       printf("%d ", x[i]);
    }

}
```



Exercises

1. Write a program that takes n numbers from user (where, n is specified by user), stores them in an array and finds the largest of them.

Tip: create a variable called max, and compare every value in the array with max.

2. Write a program that takes n numbers from user (where, n is specified by user), stores them in an array and calculates the average of those numbers.

2. Write a program that copies the positive numbers in another array.