



Syrian Private University

Introduction to Algorithms and Programming

Instructor: Dr. Mouhib Alnoukari



Arrays



Problem:

Read 10 numbers from the keyboard and store them

```
// solution #1
int a0, a1, a2, a3, a4, a5, a6, a7, a8, a9;

printf("Enter a number: ");
scanf(" %d", &a0);

printf("Enter a number: ");
scanf(" %d", &a1);

//...

printf("Enter a number: ");
scanf(" %d", &a9);
```



Arrays

- Arrays are C data types that help us organize large amounts of information

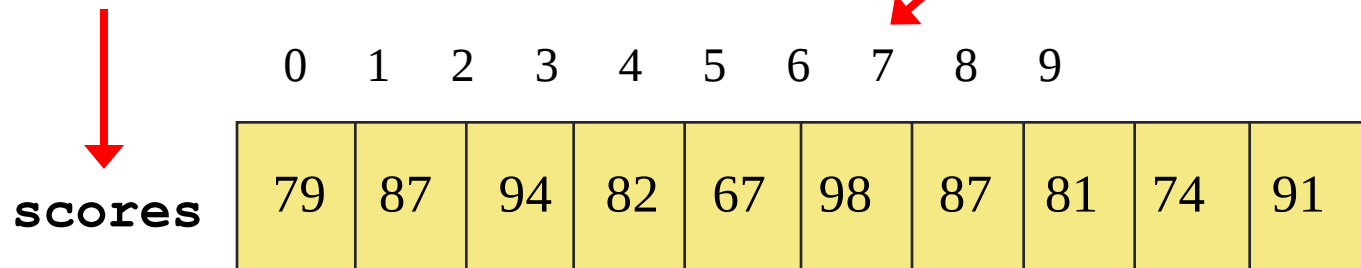


Arrays

- An *array* is an ordered list of values

The entire array
has a single name

Each value has a numeric *index*



	0	1	2	3	4	5	6	7	8	9
scores	79	87	94	82	67	98	87	81	74	91

An array of size N is indexed from zero to N-1

This array holds 10 values that are indexed from 0 to 9

An array with 8 elements of type double

```
double x[8];
```

Array x

x[0] x[1] x[2] x[3] x[4] x[5] x[6] x[7]

16.0	12.0	6.0	8.0	2.5	12.0	14.0	-54.5
------	------	-----	-----	-----	------	------	-------

Problem:

Read 10 numbers from the keyboard and store them

```
// solution #2
int a[10]; // use an array
for(i=0; i< 10; i++)
{
    printf("Enter a number: ");
    scanf(" %d", &a[i]);
}
```



Arrays

- A particular value in an array is referenced using the array name followed by the index in brackets
- For example, the expression

`scores[2]`

refers to the value 94 (the 3rd value in the array)

- That expression represents a place to store a single integer and can be used wherever an integer variable can be used



Arrays

- For example, an array element can be assigned a value, printed, or used in a calculation :

```
scores[2] = 89;
```

```
scores[first] = scores[first] + 2;
```

```
mean = (scores[0] + scores[1])/2;
```

```
printf ("Top = %d", scores[5]);
```

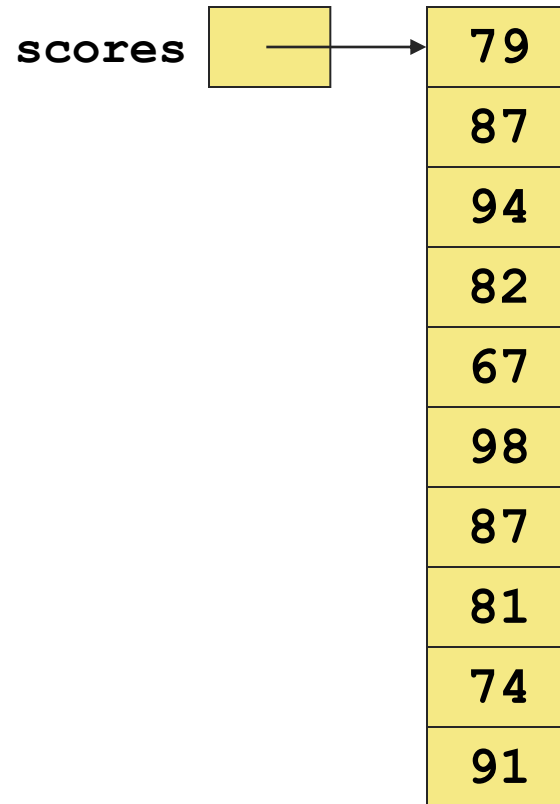


Arrays

- The values held in an array are called *array elements*
- An array stores multiple values of the same type – the *element type*
- The element type can be a primitive type
- Therefore, we can create an array of integers, an array of floats, an array of doubles.

Arrays

- Another way to depict the `scores` array:



Declaring Arrays

- It is possible to initialize an array when it is declared:

```
float prices[3] = {1.0, 2.1,  
2.0};
```

- Or to initialize it later:

```
int a[6];
```

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

```
a[1]=6;
```



Declaring Arrays

- Declaring an array of characters of size 3:

```
char letters[3] = { 'a', 'b',  
'c' };
```

- Or we can skip the 3 and leave it to the compiler to estimate the size of the array:

```
char letters[] = { 'a', 'b', 'c' };
```



For loops and arrays

```
#define N 10
```

```
int a[N];
```

```
int i;
```

```
...
```

```
for (i=0; i < N; i++)
```

```
    printf("%d\n", a[i]);
```

```
for (i=0; i <= N; i++)    // this is  
    an error
```

```
    printf("%d\n", a[i]); // out of  
    bounds
```

For loops and arrays

```
#define N 10  
int a[N+1];  
int i;  
  
...  
for (i=0; i <= N; i++)  
    printf("%d\n", a[i]);
```

Problem:

Input 10 student IDs and their corresponding grades (A through F). Then find out the number of As, and print the names of the students that got an A.



Comparing Float Values

- You should rarely use the equality operator (`==`) when comparing two floating point values (`float` or `double`)
- Two floating point values are equal only if their underlying binary representations match exactly
- Computations often result in slight differences that may be irrelevant
- In many situations, you might consider two floating point numbers to be "close enough" even if they aren't exactly equal

Comparing Float Values

- To determine the equality of two floats, you may want to use the following technique:

```
if (fabs(f1 - f2) < TOLERANCE)
    printf ("Essentially equal");
```

- If the difference between the two floating point values is less than the tolerance, they are considered to be equal
- The tolerance could be set to any appropriate level, such as 0.000001

Comparing Characters

- As we've discussed, C character data is based on the ASCII character set
- ASCII establishes a particular numeric value for each character, and therefore an ordering
- We can use relational operators on character data based on this ordering
- For example, the character '+' is less than the character 'J' because it comes before it in the ASCII character set

