# CS10003: Programming & Data Structures

## Dept. of Computer Science & Engineering Indian Institute of Technology Kharagpur

Autumn 2020



### Printing numbers in reverse

• Given 5 integers as inputs, print them in reverse order.

```
#include<stdio.h>
int main()
{
  int a, b, c, d, e;
  printf("Enter 5 integers: ");
  scanf("%d%d%d%d%d", &a, &b, &c, &d, &e);
  printf("The numbers in reverse order: ");
  printf("%d, %d, %d, %d, %d", e, d, c, b, a);
  return 0;
```

## Printing numbers in reverse: continued

- What if there are 1000 integers? Use 1000 variables?
- Solution: use arrays.
- Array is a data structure which can represent a collection of data items which have the same data type (float/int/char/...).
- This is exactly what will help us here!

### Printing in numbers in reverse Using Arrays

```
int main()
```

```
{
```

}

```
int n, A[100], i;
```

```
printf("How many numbers to read? ");
scanf("%d", &n);
for (i = 0; i < n; ++i)
scanf("%d", &A[i]);
for (i = n -1; i >= 0; --i)
printf("%d ", A[i]);
printf("\n");
return 0;
```

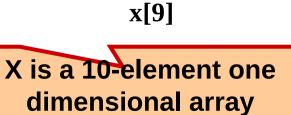
## **Using Arrays**

All the data items constituting the group share the same name int x[10];

Individual elements are accessed by specifying the index



x[0] x[1] x[2]



6

### Another example

```
int main()
{
 int i;
 int data[10];
 for (i=0; i<10; i++)
  data[i]= i;
 i=0;
 while (i<10)
 {
  printf("Data[%d] = %d\n", i, data[i]);
  i++;
 }
 return 0;
```

"data refers to a block of 10 integer variables, data[0], data[1], ..., data[9]



## The result

#### Array size constant

#### int main()

{

```
int i;
int data[10];
for (i=0; i<10; i++) data[i]= i;
i=0;
while (i<10)
{
 printf("Data[%d] = %d\n", i, data[i]);
 i++;
}
Return 0;
```

#### Output

Data[0] = 0
Data[1] = 1
Data[2] = 2
Data[3] = 3
Data[4] = 4
Data[5] = 5
Data[6] = 6
Data[7] = 7
Data[8] = 8
Data[9] = 9

## **Declaring Arrays**

- Like variables, the arrays used in a program must be declared before they are used.
- General syntax:
  - type array-name [size]; type specifies the type of element that will be contained in the array (int, float, char, etc.)
  - size is an integer constant which indicates the maximum number of elements that can be stored inside the array.
- int marks [5]. marks is an array that can store a maximum of 5 integers.

•Examples: int x[10]; char word[10]; float distance[150]; char name[35];

• If we are not sure of the exact size of the array, we can define an array of a large size

int marks[50];

though in a particular run we may only be using, say, 10 elements.

## Accessing array elements

- A particular element of the array can be accessed by specifying two things:
  - Name of the array
  - Index (relative position) of the element in the array
- In C, the index of an array starts from <u>zero</u>
- Example:
  - An array is defined as int x[10];
  - The first element of the array x can be accessed as x[0], fourth element as x[3], tenth element as x[9], etc.

## Contd.

• The array index must evaluate to an integer between 0 and n-1 where n is the maximum number of elements possible in the array

a[x+2] = 25; b[3\*x-y] = a[10-x] + 5;

• Remember that each array element is a variable in itself, and can be used anywhere a variable can be used (in expressions, assignments, conditions,...) How is an array stored in memory?

• Starting from a given memory location, the successive array elements are allocated space in consecutive memory locations



- x: starting address of the array in memory
- k: number of bytes allocated per array element
- a[i] → is allocated memory location at address
   x + i\*k

## Storage

#### int main()

```
{
    int i;
    int data[10];
    for(i=0; i<10; i++)
    printf("&Data[%d] = %u\n", i, &data[i]);
    return 0;</pre>
```

#### Output

```
&Data[0] = 3221224480
&Data[1] = 3221224484
&Data[2] = 3221224488
&Data[3] = 3221224492
&Data[4] = 3221224496
&Data[5] = 3221224500
&Data[6] = 3221224504
&Data[7] = 3221224508
&Data[8] = 3221224512
&Data[9] = 3221224516
```

## Initialization of Arrays

• General form:

type array\_name[size] = { list of values };

• Examples:

int marks[5] = {72, 83, 65, 80, 76};

char name[4] = {'A', 'm', 'i', 't'};

• The size may be omitted. In such cases the compiler automatically allocates enough space for all initialized elements

int flag[] = {1, 1, 1, 0}; char name[] = {'A', 'm', 'i', 't'};

## How to read the elements of an array?

 By reading them one element at a time for (j=0; j<25; j++)</li>

scanf ("%f", &a[j]);

- The ampersand (&) is necessary
- The elements can be entered all in one line or in different lines