CS10003:
Programming & Data Structures

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

Autumn 2020

How to read the elements of an array?

- By reading them one element at a time

```
for  (j=0; j<25; j++)
    scanf  (“%f”, &a[j]);
```

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

A Warning

- In C, while accessing array elements, array bounds are not checked
- Example:

```
int   marks[5];
:
:
marks[8] = 75;
```

  - The above assignment would not necessarily cause an error
  - Rather, it may result in unpredictable program results

Reading into an array

```
int main()
{
    const int MAX_SIZE = 100;
    int i, size;
    float marks[MAX_SIZE];
    float total;
    scanf("%d",&size);
    for (i=0, total=0; i<size; i++)
    {
        scanf("%f",&marks[i]);
        total = total + marks[i];
     }
   printf("Total = %f \n Avg = %f\n", total,
total/size);
   return 0;
}
```

Output

```
4
2.5
3.5
4.5
5
Total = 15.500000
 Avg = 3.875000
```

How to print the elements of an array?

- By printing them one element at a time

```
for  (j=0; j<25; j++)
    printf  (“\n %f”, a[j]);
```

  - The elements are printed one per line

```
printf  (“\n”);
for  (j=0; j<25; j++)
    printf (“ %f”, a[j]);
```

  - The elements are printed all in one line (starting with a new line)

How to copy the elements of one array to another?

- By copying individual elements

```
for  (j=0; j<25; j++)
    a[j] = b[j];
```

- The element assignments will follow the rules of assignment expressions

- Destination array must have sufficient size

Example 1: Find the minimum of a set of 10 numbers

```
int main()
{
    int  a[10], i, min;

    for  (i=0; i<10; i++)
        scanf (“%d”, &a[i]);

    min = a[0];
    for  (i=1; i<10; i++)
    {
        if  (a[i] < min)
            min = a[i];
    }
    printf (“\n Minimum is %d”, min);
    return 0;
}
```

Alternate Version 1

Change only one line to change the problem size

```
const int size = 10;

int main()
{
    int  a[size], i, min;

    for  (i=0; i<size; i++)
        scanf (“%d”, &a[i]);

    min = a[0];
    for  (i=1; i<size; i++)
    {
        if  (a[i] < min)
            min = a[i];
    }
    printf (“\n Minimum is %d”, min);
    return 0;
}
```

Alternate Version 2

Change only one line to change the problem size

Used #define macro

```
#define size 10

int main()
{
    int  a[size], i, min;

    for  (i=0; i<size; i++)
        scanf (“%d”, &a[i]);

    min = a[0];
    for  (i=1; i<size; i++)
    {
        if  (a[i] < min)
            min = a[i];
    }
    printf (“\n Minimum is %d”, min);
    return 0;
}
```

#define macro

- #define X Y
- Preprocessor directive
- Compiler will first replace all occurrences of string X with string Y in the program, then compile the program
- Similar effect as read-only variables (const), but no storage allocated
- We prefer you use const instead of #define

Alternate Version 3

Define an array of large size and use only the required number of elements

```
int main()
{
    int  a[100], i, min, n;

    scanf (“%d”, &n);  /* Number of elements */
    for  (i=0; i<n; i++)
        scanf (“%d”, &a[i]);

    min = a[0];
    for  (i=1; i<n; i++)
    {
        if  (a[i] < min)
            min = a[i];
    }
    printf (“\n Minimum is %d”, min);
    return 0;
}
```