

S10_1

1 D - A r r a y s



Objectives

To learn and appreciate the following concepts:

- Declare, initialize and access 1D array.
- Write programs using common data structures namely arrays and strings and solve problems.

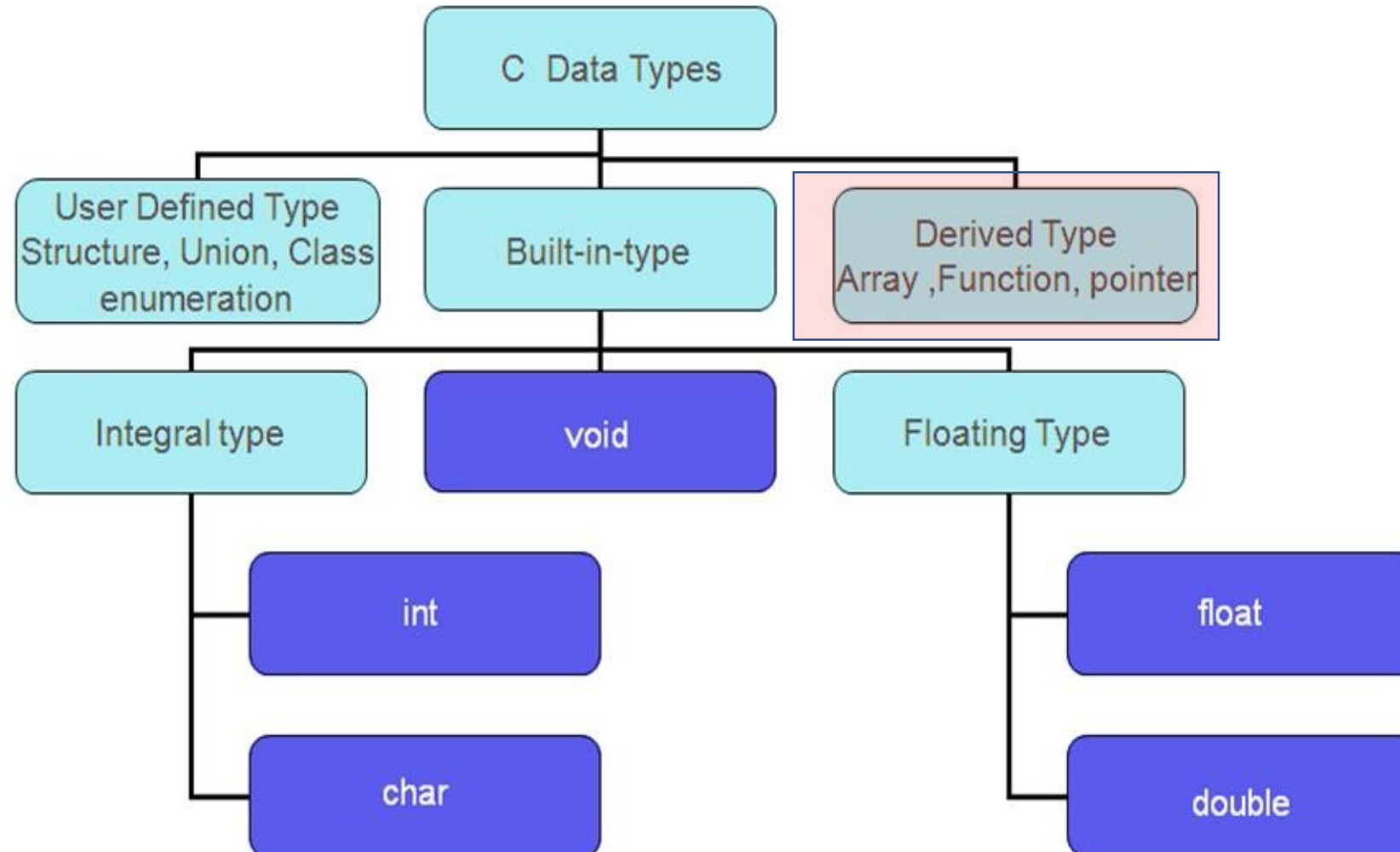


Session outcome

- At the end of session student will be able to
 - Declare, initialize and access 1D array
 - Write programs using 1D array



Data types - Revisit





Arrays

- **An array is a group of related data items that share a common name.**
- The array elements are placed in contiguous **memory locations**.
- A particular value in an array is indicated by writing an integer number called **index number** or **subscript** in **square brackets** after the array name.
- The least value that an index can take in array is 0..



Arrays

Array Declaration:

```
data-type name [size];
```

where data-type is a valid data type (like int, float, char...)

- ✓ name is a valid identifier
- ✓ size specifies how many elements the array has to contain.
 - size field is always enclosed in square brackets [] and takes static values.
- For example an array salary containing 5 elements is declared as follows

```
int salary [5];
```

Arrays - *One Dimensional*

Name	roll[0]	roll[1]	roll[2]	roll[3]	roll[4]	roll[5]	roll[6]	roll[7]
Values	12	45	32	23	17	49	5	11
Address	1000	1002	1004	1006	1008	1010	1012	1014

1-D Array memory arrangement

- A **linear list** of fixed number of data items of same type.
- These items are accessed using the same name using a single subscript. E.g. **roll[0]**, **roll[1]**.... or **salary [1]**, **salary [4]**
- A list of items can be given one variable name using only one subscript and such a variable is called a **single-subscripted variable** or a **one- dimensional array**.



Arrays - 1D

Total size:

The Total memory that can be allocated to 1Darray is computed as

Total size = size ***(sizeof(data_type));**

where size → number of elements in 1-D array

data_type → basic data type

sizeof() → is an unary operator which returns the size of data type in bytes.



Arrays - 1D

How to read & display the values of an array and store it !

```
int main() {  
    int arr[50],n; // declaration of 'arr'  
    printf(" enter value  of n\n"); // no of elements  
    scanf("%d", &n); // reading the limit into n  
    for(int i=0;i<n;i++)  
    {  
        scanf ("%d", &arr[i]) ; // reading n elements  
    }  
    for(int j=0; j<n;j++) //displaying n elements  
    {  
        printf ("%d",arr[j]) ;  
        printf ("\t") ;  
    }  
    return 0;  
}
```



Initializing one-dimensional array

```
int number[3] = {0,0,0}; or {0} ;
```

→ declares the variable number as an array of size 3 and will assign 0 to each element.

```
int age[ ] = {16,25,32,48,52,65} ;
```

→ declares the age array to contain 6 elements with initial values 16, 25, 32, 48, 52, 65 respectively



Initializing one-dimensional array

Initialize all the elements of an integer array 'values' to zero

```
int values[20];
```

Begin for loop

Initialize counter

Set limit for counter

```
for (int i=0; i<20; i++)
```

Initialize element in array 'values'

```
values[i]=0;
```

Increment counter



Printing one-dimensional array

For example

```
int x[3] = {9,11,13};
```

```
printf("%d\n",x[0]);
```

```
printf("%d\n",x[1]);
```

```
printf("%d\n",x[2]);
```

or

```
int x[3] = {9,11,13};
```

```
for (int i = 0; i<3; i++)
```

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

Output:

9

11

13



Program to read n elements into an array and print it

```
int a[10], i, n;  
  
printf("enter no of numbers");  
  
scanf("%d",&n);  
  
printf("enter n numbers \n");  
  
for (i=0;i<n;i++)  
    scanf ("%d\n", &x[i]) ;  
  
printf("\nNumbers entered are:\n");  
  
for (i=0;i<n;i++)  
    printf ("%d\n", a[i]) ;
```

Output:

enter no of numbers

3

enter n numbers

9

11

13

Numbers entered are:

9

11

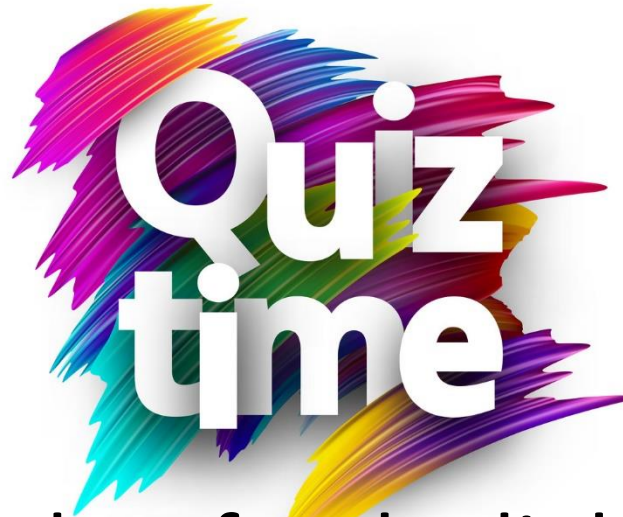
13



Program to add two array elements and store the corresponding sum elements in another array

```
int a[10], b[10], c[10], n, m, i;  
printf("enter no. of numbers in first array\n");  
scanf("%d",&n);  
//first array reading  
for (i=0 ; i<n ; i++)  
    scanf ("%d" , &a[i] ) ;  
printf("enter no of numbers in second array\n");  
scanf("%d",&m);  
//second array reading  
for (i=0 ; i<m ; i++)  
    scanf ("%d" , &b[i] ) ;
```

```
if(m==n)  
{    //addition  
    for (i=0 ; i<m ; i++)  
        c[i]=a[i]+b[i] ;  
  
    printf("Sum of given array elements\n");  
  
    for(i=0;i<n;i++)  
        printf("%d\n",c[i]);  
}  
else  
    printf("cannot add");  
}
```



Go to posts/chat box for the link to the question

submit your solution in next 2 minutes

The session will resume in 3 minutes



Write a program to reverse an array

```
int a[20], i, j, n, temp;  
  
printf("enter n \n");  
  
scanf("%d", &n);  
  
printf("\n Enter values for an array");  
  
for(i=0;i<n;i++)  
  
    scanf("%d", &a[i]);
```

Example : a[]={1, 2, 3, 4, 5}

Enter values

n=5

1 2 3 4 5

Reversed array

5 4 3 2 1

Array

Reversed

array

a[0]=1

a[0]=5

a[1]=2

a[1]=4

a[2]=3

a[2]=3

a[3]=4

a[3]=2

a[4]=5

a[4]=1

Contd...



Reversing an array

```
for(i=0, j=n-1; i<n/2; i++, j--)  
{  
    temp=a[i];  
    a[i]=a[j];  
    a[j]=temp;  
}  
  
printf("\n Reversed array: \n");  
for(i=0;i<n;i++)  
    printf("%d\t", a[i]);  
}
```

Example :

a[]={1, 2, 3, 4, 5}

Output:

Enter values for an array

n=5

1 2 3 4 5

Reversed array

5 4 3 2 1



Summary

- Arrays
- 1 Dimensional arrays (lists)
- Problems on 1D arrays