Page No.
Date

"Arrays"

Topics:-

- Concepts of array
- One & two dimensional Arrays
- Declaration & Initialization of arrays

↦ Introduction :-

↳ Need For Array :-

↦ As far as we know we have only used fundamental data types char, int, float, double, These all are useful to store only one value at a time.

↦ So, through these data-type we can handle only limited amount of data.

↦ In many application however we need to handle large amount of data in terms of reading & processing.

↦ Thats So that C supports derived data-type known as "Array". That can be used for such applications,

Ex:-
- Marks of Students
- Table
- Matrix
- List of Customer's name, telephone nums etc.

↳ What is an Array?

- An array is a fixed size sequence collection of elements of the same data-type.

↳ Types of an Array :-

1. One - dimensional Array
2. Two - dimensional Array
3. Multi dimensional Array

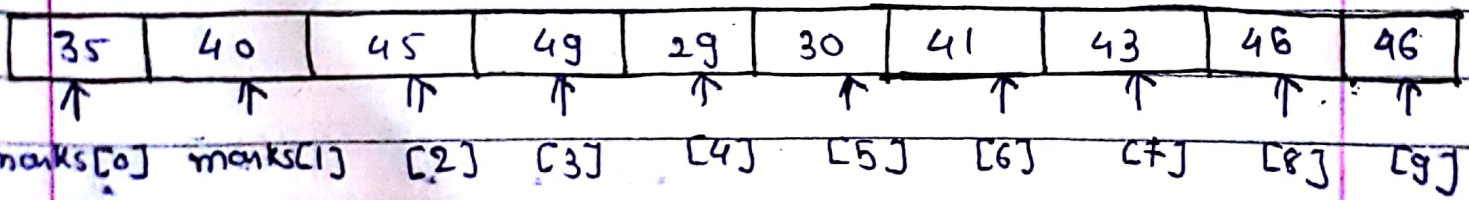
↳ Initialization of One - dimensional Array:-

Declaration Part :-

Data-type : Variable or Array Name [Size]

Example :- int marks [10];

- It represents the marks of ten students which contain numeric value.
- It always start from Index 0 to n-1.



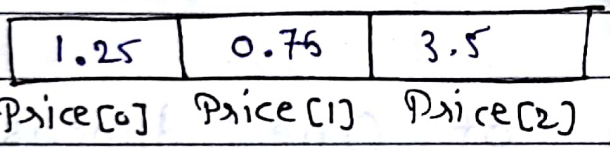
↳ Another method to Represent an Array :-

- Initialization :-

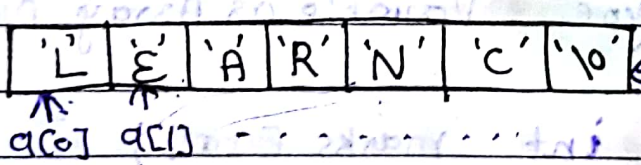
int marks[10] = {35, 40, 45, 99, 29, 30, 41, 43, 45, 46};

- Another Example :-

Float Price[3] = {1.25, 0.75, 3.5};



char a[8] = {'L', 'E', 'A', 'R', 'N', 'C', '\0'};



\0 - Null character (which indicates the termination of String)

↳ Example:-

1. Wsit a Program to Print & Read 10 numbess from user using an Array.

```
#include <stdio.h>
#include <conio.h>
void main()
{   int a[10], i;
    clrscr();
    Pfc"Enter any 10 numbers:");
    for(i=0; i<10; i++)
    {
        scanf("%d", &a[i]);
    }
    Pfc"User have entered following nums:");
    for(i=0; i<10; i++)
    {
        Pfc"%d\n", a[i]);
    }
    getch();
}
```

2. Write a Program to Print Sum of any 10 numbers using 1-D Array.

```
Void main()
{
```

```
int n[10], i, sum=0;
pf("Enter 10 numbers");
for(i=0; i<10; i++)
{
    scanf("%d", &n[i]);
    sum = sum + n[i];
}
pf("sum = %d", sum);
getch();
}
```

3. Write a program to find maximum element from 1-D Array.

```
Void main()
{
    int n[10], i, max=0, n;
    clrscr();
    pf("How many elements ?");
    sf("%d", &n);
    pf("Enter %d elements\n", n);
    for(i=0; i<n; i++)
    {
        scanf("%d", &n[i]);
        if(n[i] > max)
        {
            max = n[i];
        }
    }
```

```
    printf("Maximum num = %d", max);
    clrscr();
}
```

4. Write a program to sort given array in an ascending order.

```
Void main()
{
    int a[10], n, i, j, t;
    clrscr();
    pf("Enter n:");
    sf("%d", &n);
    pf("Enter %d elements", n);
    for(i=0; i<n; i++)
    {
        scanf("%d", &a[i]);
    }
    for(i=0; i<n; i++)
    {
        for(j=i+1; j<n; j++)
        {
            if(a[i]>a[j])
            {
                t=a[i];
                a[i]=a[j];
                a[j]=t;
            }
        }
    }
```

```
printf("Sorted List \n");
for(i=0; i<n; i++)
{
    printf("%d \n", a[i]);
}
getch();
}
```

↦ Write a C Program to read 10 numbers from user & find sum, maximum, minimum & average of them.

↦
```
Void main( )
{
    int i,a[10], sum=0, max, min;
    float avg;
    Pfc"Enter 10 elements of array:");
    for(i=0; i<10; i++)
    {
        ssc"%d", &a[i]);
    }
max = a[0];
min = a[0];

for(i=0; i<10; i++)
{
    sum = sum + a[i];

    if(a[i]>max)
    {
        max = a[i];
    }
    if(a[i]<min)
    {
        min = a[i];
    }
}
avg = (float)sum/10;
```

```
pfc" Sum of 10 elements = %d", sum);
pfc"\n Max num = %d", max);
pfc"\n min num = %d", min);
pfc"\n Average = %f", avg);

getch();
}
```

2. Write a Program which declares array of 10 integers, enter data & sum all the elements which are even. Also find maximum number from them.

```
Void main()
{
   int a[10], i, sum=0, max = 0;

  pfc"Enter 10 elements:");
  for(i=0; i<10; i++)
  {
        sfc"%d", &a[i]);
  }

  for(i=0; i<10; i++)
  {
        Sum = Sum * a[i];
        if(a[i]%2==0)
        {
            sum = sum + a[i];
            if(a[i] > max)
            {
                max = a[i];
            }
        }
  }
```

```
Pf c" Sum of even nums : %d", sum);
Pf c"\n max num = %d", max);
getch();
}
```

⇒ CWAP to read in array of intergers & Print its elements in reverse order.

⇒
```
void main()
{
    int n, i, a[10];

    Pf c"Enter your limit:");
    Sf c"%d", &n);

    for(i=0; i<n; i++)
    {
        sf c"%d", &a[i]);
    }

    Pf c"Reverse array given below:");

    for(i=n-1; i>=0; i--)
    {
        pf c"%d", a[i]);
    }

    getch();

}
```

⇒ WAP to accept array of N integers & find largest odd number as well as largest even number & display them.

⇒
```
void main()
{
    int i,n, a[100], max_even, max_odd;
    int even_count=0, odd_count=0;
    pf("Enter n:");
    sf("%d", &n);

    for(i=0; i<n; i++)
    {
        sf("%d", &a[i]);
    }

    for(i=0; i<n; i++)
    {
        if(a[i]%2==0)
        {
            if(even_count==0)
                max_even = a[i];
            else if(max_even<a[i])
                max_even = a[i];
            even_count++;
        }
        else
        {   if(odd_count==0)
                max_odd = a[i];
            else if(max_odd<a[i])
                max_odd = a[i];
            odd_count++;
        }
    }
}
```

↳ Two-dimensional Array :-

↦ In a two-dimensional array to store table of values in a forms of Rows & columns. Thus it reffered to as a matrix or tables.

↳ Declaration Syntax :-

Data-type variable-name [rows] [columns]

Ex :-

int table [2][3];

↳ Initialization Syntax :-

int table [2][3] = {2, 11, 3, 5, 1, 10};

	col 0	col 1	col 2
row 0	2	11	3
row 1	5	1	10

↳ Example :- 5

1. Write a Program to add two matrix.

↦
```
Void main()
{
    int a[3][3], b[3][3], c[3][3], i, j;
    clrscr();
    pf("Enter 3x3 matrix A \n");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            Scanf("%d", &a[i][j]);
        }
    }
    pf("Enter 3x3 matrix B \n");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            scanf("%d", &b[i][j];
        }
    }
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            c[i][j] = a[i][j] + b[i][j];
        }
    }
```

```
pf c" Resultant matrix c:");
for (i=0; i<3; i++)
{
    for (j=0; j<3; j++)
    {
        printf ("%d\t", c[i][j]);
    }
    printf("\n");
}
getch();
}
```

2. Write a program to multiply two 3x3 matrices.

```
Void main()
{
  int a[3][3], b[3][3], c[3][3], i, j, sum=0;

  pf c", Enter first matrix: ");
  for (i=0; i<3; i++)
  {
      for (j=0; j<3; j++)
      {
          scanf ("%d", &a[i][j]);
      }
  }
pf c" \n Enter second matrix ");
```

```
for (i=0; i<3; i++)
{
    for(j=0; j<3; j++)
    {
        scanf("%d", &b[i][j]);
    }
}

for (i=0; i<=2; i++)
{
    for (j=0; j<=2; j++)
    {
        sum=0;
        for (k=0; k<=2; k++)
        {
            sum=sum+a[i][k]*b[k][j];
        }
        c[i][j]=sum;
    }
}
printf("\n Multiplication:");
for (i=0; i<3; i++)
{
    for (j=0; j<3; j++)
    {
        printf("%d", c[i][j]);
    }
    printf("\n");
}
getch();
```

⇒ WAP to find maximum element from 3X3 matrix.

```
⇒ void main( )
{
    int i, j, a[3][3], max;
    Pf("Enter the elements of matrix:");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            sf("%d", &a[i][j]);
        }
    }
    max = a[0][0];
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            ~~if(a[i][j]>max)~~
            ~~max~~ if(a[i][j] > max)
                max = a[i][j];
        }
    }
```

```
Pf("Max element = %d", max);
getch();
}
```

⇒ Write a Program in C to count num of positive, negative & zero elements from 3x3 matrix.

⇒
```
Void main()
{
    int i, j, a[3][3], pos=0, neg=0, zero=0;
    Pf("Enter elements of matrix:");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            Sf("%d", &a[i][j]);
        }
    }
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            if(a[i][j] > 0)
            pos++;
            else if(a[i][j] < 0)
            neg++;
            else
```

```
            zero++;
        }
    }

    Pfc"Total positive num = %d", pos);
    pfc"\n Total Negative num = %d", neg);
    Pfc"\n Total zeros = %d", zero);

    getch();
}
```

⇒ WAP to read two matrices from the user & store multiplication of two matsies in the resultant matrix. i.e C = A*B.

↦

```
Void main()
{
    int a[10][10], b[10][10], c[10][10];
    int m,n,p,q;
    int i,j,k,sum;

    Pfc"Enter the size mXN for 1st Matrix";
    Sfc"%d %d", &m, &n);

    Pfc"Enter the size of pXq for 2nd matrix";
    Sfc"%d %d", &p, &q);

    if(n==p)
    {
        Pfc"Enter 1st Matrix:");
        for(i=0; i<m; i++)
        {
```

```
        for (j=0; j<n; j++)
        {
            sf("%d", &a[i][j]);
        }
    }
    pf("Enter 2nd matrix:");
    for (i=0; i<p; i++)
    {
        for(j=0; j<q; j++)
        {
            sf("%d", &b[i][j]);
        }
    }
    pf("Resultant matrix:\n");
    for (i=0; i<m; i++)
    {
        for (j=0; j<q; j++)
        {
            sum = 0;
            for (k=0; k<n; k++)
            {
                sum = sum + a[i][k] * b[k][j];
            }
            c[i][j] = sum;
            pf("%d", c[i][j]);
        }
        pf("\n");
    }
}
else
    pf("Matrix Multiplication not possible");
```

Page No
Date

```
    getch();
}
```

↳ Multidimensional Array :-

⇒ C allows arrays of three or more dimensions.

Ex :-

```
int table [3] [5] [12] ;
```

Suppose 1st index - Year
2nd index - city
3rd index - month

↳ Advantage of Arrays :-

1. Random Access - Any element in an element in an array can be accessed immediately if its exact position in the array is known.
2. Easy to use.

↳ Limitations of Arrays :-

1. Constant size.
2. Constant data-type.

3. Arrays do not support copy assignment (You can't write array marks = array scores).

Prepared By:-

Ms. Jinal Zalavadiya

M.E(C.E), LJIET