1. Array: An array can be defined as a collection of more than one similar values. Similar values imply that their data type should be same. The name of all these values will be same but their position will differ from each other. It is just like a class which contains more than one students but having different roll numbers. These roll numbers can be treated as their position in the class. In the same way, every pessanger in a train is having his own seat number. So this seat number can be treated as position of pessangers in the train. Elements of an array can have same value but their position must differ from each other. In case of C language, the position of elements of an array starts from zero.

2. To define an array: In order to define an array we need to provide its data type, name and either total number of elements or all elements values. The general syntax for defining an array is as follows:

```
data_type array_name[size];

data_type array_name[] = { element1, element2, element3, element4 };

int rates[10];
```

Here rates is an array of type int which can store rate of maximum 10 items.

```
int rolls[] = { 101, 201, 302, 405 };
```

The moment when we specify the elements of an array at the time of declaration, we can mention of we can not mention the size of the array within square bracket. It is optional in this case.

```
int rolls[4] = { 101, 201, 302, 405 };
```

Here the position of roll 101 is 0, the position of roll 201 is 1, the position of roll 302 is 2 and the position of roll 405 is 3. So if an array is having n elements then the position of elements will start from 0 and go up to n-1.

So whenever there is an array then we are having more than one elements and every element of the array is having two aspects associated with it - element value and element position.

3. To assign some value to a particular element of an array: In order to place some value in a particular element of an array we need to specify the array name and its position while putting some value in it.

```
array_name[position]=value;

rates[0]=5;
```

Here we are putting the value 5 at position 0. So the rate of first item will be 5.

```
rates[4]=10; //Rate of 5th item is 10.
```

4. To put value in all elements of an array: It can be done in two ways - with or without using loop. Then loop can be used either when there is some similarity in the vaues of elements or the value of elements are being taken input from the user. If there is no similarity in the values of elements then we need to assign them to individual elements of array using separate statements.

```
int marks[5];
```

Here in this array we can store marks of five students.

```
marks[0]=67;
marks[1]=87;
marks[2]=99;
marks[3]=89;
marks[4]=65;

for(int i=0; i<5; i++)
{
    printf("Enter marks of student#%d: ", (i+1));
    scanf("%d",&marks[i]);
}
```

5. To display the value of an element of an array:

```
printf("%d",marks[0]); //It will display 1st marks that is 67.
printf("%d",marks[1]); //It will display 2nd marks that is 87.
```

In order to display all elements of an array we need to write a looping statements wh ich will start loop from the 1st element and then it will go up to the last element.

```
i=0;
while(i<5)
{
    printf("Marks#%d = %d\n", i+1, marks[i]);
    i++;
}
```

April 10, 2020:

Task#1: Write a program in C to take input 10 integers from the user, then store them in an array. Then display those integers along with their nature that whether it is even or odd.

Task#2: Write a program in C to take input 10 years from the user and then store them in an array. Then display those years along with their nature that whether it is leap or non leap year.

April 11, 2020:

6. It is not necessary that always we need to store elements in an array equal to its siz e. The size of an array is the maximum number of elements that we can store in it. Genera lly we store less number of elements as compared to the size of the array. So in such cas es, we need to define another variable to keep track of how many elements have been actua lly stored in the array.

7. To find minimum/maximum value from an array of integers:

I. To find minimum out of two numbers:

```
int x, y, min;
x=12;
y=34;

if(x<y) min=x;
if(y<x) min=y;
```

II. To find minimum out of there numbers:

```
int a, b, c, min;
a=3;
b=2;
c=5;

if(a<b && a<c) min=a;
if(b<a && b<c) min=b;
if(c<a && c<b) min=c;
```

III. To find minimum out of n integers:

The element of the array for which there is no other element less than it, can be declared as the minimum element. The main feature of smallest element is that there is n o other element smaller than it. So in case of array we can write a loop and then check f or the element for which there is no other smaller element. That element can be declared as the smallest element.

```
int marks[]={ 60, 50, 70, 30, 80 };

marks smaller marks   count (how many elements are smaller than current element)
60    50, 30          :2
50    30              :1
70    60, 50, 30      :3
30                    :0 - smallest
80    60, 50, 70, 30  :4
```

```
        for(i=0; i<5; i++)
        {
            //loop to find total number of elements smaller that marks[i]
            count=0;
            for(j=0; j<5; j++)
            {
                if(marks[j]<marks[i]) count++;
            }
            if(count==0)
            {
                printf("Minimum marks is %d\n", marks[i]);
                break;
            }
        }
```

Task#1: WAP to find the sum and average of minimum and maximum elements from an array of integers.

Task#2: WAP to find minimum even number from an array of integers.

Task#3: WAP to find the sum of minimum odd and maximum even number from an array of integers.

```
#include<stdio.h>
int main()
{
        int years[100]; //10 is the maximum number of elements that we can store here.
        int i, total;

        printf("Enter total number of years to be cheked: ");
        scanf("%d",&total);

        if(total>100) printf("Maximum 100 years can be checked.\n");
        else
        {
                printf("Now enter %d years\n",total);
                for(i=0; i<total; i++)
                {
                        printf("Enter year#%d: ",(i+1));
                        scanf("%d",&years[i]);
                }
                printf("Year Nature:\n\n");
                for(i=0; i<total; i++)
                {
                        //When year is non century and divided by 4 OR year is century an
d divided by 400 then it is leap year
                        if((years[i]%100!=0) && (years[i]%4==0)||(years[i]%100==0) && (ye
ars[i]%400==0))
                                printf("%d is a leap year.\n",years[i]);
                        else
                                printf("%d is not a leap year.\n",years[i]);
                }
        }


        return 0;
}
//B2
```

```
//Program to check nature(even/odd) of an array of integers
#include<stdio.h>
int main()
{
        //Variables
        int array[10];
        int i;
        //Inpute
        printf("Enter ten no.\n");
        for(i=0; i<10; i++)
        {
                scanf("%d",&array[i]);
        }
        //Output
        printf("Number Nature\n");
        for(i=0; i<10; i++)
        {
                if(array[i]%2==0) printf("\nEven no. = %d", array[i]);
                else printf("\n Odd no. = %d", array[i]);
        }
        printf("\n");
        return(0);
}
//B2
```

```
#include<stdio.h>

int main(int argc, char const *argv[])
{
    //Variables
    int marks[10];
    int i;

    //Input
    for(i=0; i<10; i++)
    {
        printf("Enter marks of student#%i: ",(i+1));
        scanf("%i",&marks[i]); //input in ith marks
    }
    //Output
    printf("Marks of all students are as follows:\n");
    i=0;
    while (i<10)
    {
        if(marks[i]>=50) printf("Marks#%i: %i\tResult: Pass\n",(i+1), marks[i]);
        else printf("Marks#%i: %i\tResult: Fail\n",(i+1), marks[i]);
        i++;
    }

    return 0;
}
```

```
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char const *argv[])
{
    //Variables
    int marks[100];
    int i, j, min, total, count, position;

    //Input
    printf("Enter total number of students: ");
    scanf("%d",&total);
    if(total>100)
    {
        printf("Maximum 100 students can be taken.\n");
        exit(1);
    }
    printf("Now enter marks of %d students:\n", total);
    for(i=0; i<total; i++)
    {
        printf("Enter marks of student#%d: ", (i+1));
        scanf("%d",&marks[i]);
    }

    //Processing
    for(i=0; i<total; i++)
    {
        //loop to find total number of elements smaller that marks[i]
        count=0;
        for(j=0; j<total; j++)
        {
            if(marks[j]<marks[i]) count++;
        }
        if(count==0)
        {
            min=marks[i];
            position=i;
            break;
        }
    }

    //Output
    printf("Minimum marks for student#%d is %d\n", (position+1), min);

    return 0;
}
```