Arrays

An array is a collection of similar items stored in contiguous memory locations. In programming, sometimes a simple variable is not enough to hold all the data.

Declaring an array in C++

There are couple of ways to declare an array.
Method 1:

```
int arr[5];
arr[0] = 10;
arr[1] = 20;
arr[2] = 30;
arr[3] = 40;
arr[4] = 50;
```

Method 2:

```
int arr[] = {10, 20, 30, 40, 50};
```

Method 3:

```
int arr[5] = {10, 20, 30, 40, 50};
```

Accessing Array Elements

Array index starts with 0, which means the first array element is at index 0, second is at index 1 and so on. We can use this information to display the array elements. See the code below:

```
#include <iostream>
using namespace std;

int main(){
   int arr[] = {11, 22, 33, 44, 55};
   cout<<arr[0]<<endl;
   cout<<arr[1]<<endl;
   cout<<arr[2]<<endl;
   cout<<arr[3]<<endl;
   cout<<arr[4]<<endl;
   return 0;
}
```

2/3 Marks Questions

1. Write the definition of a function Alter(int A[], int N) in C++, which should change all the multiples of 5 in the array to 5 and rest of
the elements as 0. For example, if an array of 10 integers is as follows:

```
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7] A[8] A[9]
55   43   20   16   39   90   83   40   48   25
```

After executing the function, the array content should be changed
as follow:

```
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7] A[8] A[9]
5    0    5    0    0    5    0    5    0    5
```

Ans 1.

```
void Alter(int A[ ],int N)
{
for (int i=0;i<N;i++)
if(A[i]%5==0)
A[i]=5;
else
A[i]=0;
}
```

2. Write the definition of a function Alter(int A[], int N) in C++, which should change all the odd numbers in the array to 1 and even numbers as 0. For example, if an array of 10 integers is as
follows:

```
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7] A[8] A[9]
55   43   20   16   39   90   83   40   48   25
```

After executing the function, the array content should be changed
as follow:

```
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7] A[8] A[9]
1    1    0    0    1    0    1    0    0    1
```

Ans 2.

```
void Alter(int A[ ],int N)
{
for (int i=0;i<N;i++)
if(A[i]%2==0)
A[i]=0;
else
A[i]=1;
     }
```

3. Write code for a function void oddEven (int s[],int N) in C++, to add 5 in all the odd values and 10 in all the even values of the array S.
e.g. If the original content of the array S is: 50 11 19 24 28.
The modified content will be : 60 16 24 34 38

```
      Ans 3.    void oddEven(int S[ ],int N)
{
      for (int i=0;i<N;i++)
      {    if(S[i]%2==0)
      S[i]+=10;
      else
      S[i]+=5;
           }
      }
```

4. Write a function in C++ TWOTOONE() which accepts two array X[ ], Y[ ] and their size n as argument. Both the arrays X[ ] and Y[ ] have the same number of elements. Transfer the content from two arrays X[ ], Y[ ] to array Z[ ]. The even places (0,2,4...) of array Z[ ] should get the contents from the array X[ ] and odd places (1,3,5...) of array Z[ ] should get the contents from the array Y[ ].

Example : If the X[ ] array contains 30,60,90 and the Y[ ] array contains 10,20,50. Then Z[ ] should contain 30,10,60,20,90,50.

```
Ans 4  void TWOTOONE(int X[], int Y[], int m, int n)
{          int Z[30], i,k=0;
 for( i=0;i<=m-1,i++)
          { Z[k]=X[i];
          k=k+2;
          }
 k=1;
 for(i=0;i<=n-1;i++)
          { Z[k]=Y[i];
            k+=2;
          }
 cout<<"the resultant array is ";
 for(k=0;k<=m+n-1; k++)
 cout<<Z[k];
 }
```

5. Write a code in C++ for a function void Convert (int T[], int N) , which repositions all the elements of array by shifting each of them one to one position before and by shifting the fast element to last.

e.g. if the content of array is

0	1	2	3
22	25	14	30

The changed array content will be:

0	1	2	3
25	14	30	22

```
Ans 5.          void Covert (int T[], int N)
        { int temp=T[0];
         for( i=0;i<(N-1);i++)
        {
          T[i]=T[i+1];
```

```
}
T[i]=temp;
}
```

6. Write a for a function void ChangeOver (int P[], int N) in C++ , which repositions all the elements of array by shifting each of them to next position and shifting last element to first position.
e.g. if the content of array is

0	1	2	3
10	14	11	21

The changed array content will be:

0	1	2	3
21	10	14	11

Ans 6.
```
void Change (int P[], int N)
{ int temp;
 for( i=0;i<(N-1);i++)
{  temp=P[N-1];
   P[N-1]=P[i];
  P[i]=temp;
}
}
```

7. Write a function SWAP2BEST (int ARR[], int Size) in C++ to modify the content of the array in such a way that the elements, which are multiples of 10 swap with the value present in the very next position in the array.
For example :
if the content of array ARR is

90, 56, 45, 20, 34, 54

The content of array ARR should become

56, 90, 45, 34, 20, 54

Ans 7.
```
void SWAP2BEST (int ARR[], int Size)
 {
        for(int i=0;i<(Size-2);i++)
        if ( ARR[i]%10==0)
        { int temp= ARR[i];
          ARR[i]=ARR[i+1];
         ARR[i+1]=temp;
        i++;
        }
 }
```

8. Write a Get1From2( ) function in C++ to transfer the content from two arrays FIRST[ ] and SECOND[ ] to array ALL[ ].

The even places(0,2, 4,...) of array ALL[ ] should get the content from the array FIRST[ ] and odd places (1, 3, 5,) of the array ALL[] should get the content from the array SECOND[ ].
Example:
If the FIRST[ ] array contains
30, 60, 90
And the SECOND[ ] array contains
10, 50, 80
The ALL[ ] array should contain
30, 10, 60, 50, 90, 80

Ans 8.

```
void Get1From2 (int ALL[],int FIRST[],int SECOND[],int N,int M)
{
for(int I=0,J=0,K=0;i<N+M; I++)
if (I%2==0)
ALL[I]=FIRST[J++];
else
ALL[I]=SECOND[K++];
}
```

9. Write a Get2From1( ) function in C++ to transfer the content from one arrays ALL[ ] to two arrays Odd[ ]and Even[]. The Even should contain values from places (0,2,4,……) of ALL[] and Odd[] should contain values from places (1,3,5,…….).
Example:
If The ALL[ ] array contain
30, 10, 60, 50, 90, 80
Then the Even[ ] array should contains
30, 60, 90
And the ODD[ ] array should contains
10, 50, 80

Ans 9.

```
void Get2From1(int ALL[],int N)
  { int p,q,i,j=0,k=0;
      if (N%2==0)
      { p=N/2; q=N/2;}
      else
      { p=N/2; q=((n/2)+1); }
      int *Odd=new int[p];
      int *Even=new int[q];
      for(i=0;i<(p+q);i++)
      { if(i%2==0)
            Even[j++]=ALL[i];
            else
            Odd[k++]=ALL[i];
      }
   }
```

10. Write a function CHANGEO in C++,which accepts an array of integer and its size as parameters and divide all those array elements by 7 which are divisible by 7 and multiply other-array elements by 3.

Sample Input Data of the array

Content of the array after Calling CHANGE( ) function

Ans 10.
```
void CHANGE (int A[ ], int N)
{
for(int I = 0; I<N; I++)
{
        if (A[I]%7 = = 0)
        A [I] = A [I] /7;
        else
        A[I] = A[I] * 3;
}
}
```

11. Write a function REASSIGNO in C++, which accepts an array of integers and its size as parameters and divide all those array elements by 5 which are divisible by 5 and multiply other array elements by 2.

Sample Input Data of the array

Content of the array after calling REASSIGNO function

Ans 11.
```
void REASSIGN (int Arr[ ], int Size)
      {
for (int i=0;i<Size;i++)
if (Arr[i]%5==0)
Arr[i]/=5;
else
Arr[i]*=2;
}
```

12. Write a function in C++, which accepts an integer array and its size as arguments and swap the elements of every even location with its following odd location.

Example :

If an array of nine elements initially
contains the elements as
2,4,1,6,5,7,9,23,10
then the function should rearrange the
array as 4,2,6,1,7,5,23,9,10

Ans 12.
```
void SwapArray(int A[ ], int N)
{
int i,j,temp;
       for(i=0;i<N-1;i+=2)
{
temp=A[i];
A[i]=A[i+1];
A[i+1]=temp;
}
```

13. Write a function in C++, which accepts an integer array and its size as parameters and rearranges the array in reverse.

Example:

If an array of nine elements initially
contains the elements as 4, 2, 5, 1,6, 7, 8, 12, 10
Then the function should rearrange the array as
10, 12, 8, 7, 6, 1, 5, 2, 4

Ans 13.

```
void receive(int A[ ], int size)
{
int temp;
for(i=0,j=size-1;i<size/2;i++,j--)
{
temp=A[i];
A[i]=A[j];
A[j]=temp;
}
}
```

14. Write function in C++ which accepts an integer array and size as arguments and replaces elements having odd values with thrice its value and elements having even values with twice its value.

Example : if an array of five elements
initially contains elements as 3, 4, 5, 16, 9
The function should rearrange the content of the array as 9, 8, 75, 32,27

Ans 14.

```
void manipulate (int a[ ],int size)
{
for (i=0;i<size;i++)
{
if (a[i]%2= =1)
a[i]=a[i]*3;
else
a[i]=a[i]*2;
cout<<a[i]<<',';
}
}
```

15. Write a function in C++ which accepts an integer array and its size as arguments and exchanges the values of first half side elements with the second half side elements of the array.

Example :

If an array of 8 elements initial content as 2, 4, 1, 6, 7, 9, 23, 10
The function should rearrange array as 7, 9, 23, 10, 2, 4, 1, 6

Ans 15.

```
void exchange(int a[],int n)
{
    int i, mid,t,pos=0;   mid=n/2;
   if(n%2!=0)
```

```
        pos=1;
         for(i=0;i<mid;i++)
 {         t=a[i];
         a[i]=a[mid+pos+i];
          a[mid+pos+i]=t;
        }
}
```

16. Write a Function to Search for an element from Array A by Linear Search.

Ans 16.
```
void Lsearch(int A[], int n, int Data)
{     int I;
      for(I=0; I<n; I++)
      {        if(A[I]==Data)
            {
                  cout<<"Data Found at : "<<I;
            }
      }
      cout<<"Data Not Found in the array"<<endl;
}
```

17. Write a Function to Search for an element from Array A by Binary Search.

Ans 17.
```
int BsearchAsc(int A[], int n, int data)
{      int Mid,Lbound=0,Ubound=n-1,Found=0;
       while((Lbound<=Ubound) && !(Found))
       {     Mid=(Lbound+Ubound)/2;        //Searching The Item
             if(data>A[Mid])
                    Lbound=Mid+1;
             else if(data<A[Mid])
                    Ubound=Mid-1;
             else
                    Found++;
       }       if(Found)
             return(Mid+1);        //returning 1ocation, if
present
               else
             return(-1);        //returning -1,if not present
}
```

18. Write a function to Sort the array A by Bubble Sort.

Ans 18.
```
      void BSort(int A[], int n)
     {
    int I,J,Temp;
    for(I=0;I<n-1;I++) //sorting
     {
       for(J=0;J<(n-1-I);J++)
            if(A[J]>A[J+1])
           {
               Temp=A[J]; //swapping
               A[J]=A[J+1];
               A[J+1]=Temp;
           }
```

```
    }
  }
```

19. Write a function to Sort the array A by Selection Sort.

Ans 19.
```
void SSort(int A[], int n)
{   int I,J,Temp,Small;
    for(I=0;I<n-1;I++)
        {                                   Small=I;
for(J=I+1;J<n;J++) //finding the smallest element
if(A[J]<A[Small])
     Small=J;
if(Small!=I)
        {
    Temp=A[I]; //Swapping
    A[I]=A[Small];
    A[Small]=Temp;
        } } }
```

20. Write a function to Sort the array A by Insertion Sort.

Ans 20.
```
void ISort(int A[], int n)
{
    int I,J,Temp;
    for(I=1;I<n;I++) //sorting
    {
       Temp=A[I];
       J=I-1;
       while((Temp<A[J]) && (J>=0))
       {
          A[J+1]=A[J];
          J--;
       }
       A[J+1]=Temp;
    }
}
```

21. What will be the status of the following list after fourth pass of bubble sort and fourth pass of selection sort used for arranging the following elements in descending order?

14, 10, -12, 9, 15, 35

Ans 21.

Bubble Sort

```
14,10,-12,9,15,35 (Original Content)
i. 14,10,9,15,35,-12
ii. 14,10,15,35,9,-12
iii. 14,15,35,10,9,-12
iv. 15,35,14,10,9,-12   (Unsorted status after 4th pass)
```

Selection Sort

```
14,10,-12,9,15,35    (Original Content)
i. 35,10,-12,9,15,14
ii. 35,15,-12,9,10,14
iii. 35,15,14,9,10,-12
iv. 35,15,14,10,9,-12
```

22. A two dimensional array P[20] [50] is stored in the memory along
the row with each of its element occupying 4 bytes, find the address of the element P[10] [30],if the element P[5] [5] is stored at the memory location 15000.

Ans 22. Loc(P[I][J]) along the row =BaseAddress+W [(I-LBR)*C+(J-LBC)]
(where C is the number of columns, LBR=LBC=0)
LOC(P[5][5])= BaseAddress + W*[I*C + J]
15000 = BaseAddress + 4*[5*50 + 5]
= BaseAddress + 4*[250 + 5]
= BaseAddress + 4*255
= BaseAddress + 1020
BaseAddress = 15000-1020
= 13980
LOC(P[10][30])= 13980 + 4*[10*50+30]= 13980 + 4*530
= 13980 + 2120
= 16100

23. A two dimensional array ARR[50][20] is stored in the memory along the row with each of its elements occupying 4 bytes. Find the address of the element ARR[30][10], if the element ARR[10] [5] is stored at the memory location 15000.

Ans 23. Loc(ARR[I][J]) along the row =BaseAddress + W[(I - LBR)*C+(J - LBC)]
(where C is the number of columns, LBR = LBC = 0
LOC(ARR[10][5])= BaseAddress + W [I*C + J]
15000 = BaseAddress + 4[10*20 + 5]
= BaseAddress + 4[200 + 5]
= BaseAddress + 4 x 205
= BaseAddress + 820
BaseAddress = 15000-820
= 14180
LOC(ARR[30][10])= 14180 + 4[30 * 20 + 10]
= 14180 + 4 * 610 = 14180 + 2440
= 16620
OR
LOC(ARR[30][10])= LOC(ARR[10][5])+ W[(I-LBR)*C + (J-LBC)]
= 15000 + 4[(30-10)*20 + (10-5)]
= 15000 + 4[20*20 + 5]
= 15000 + 4 *405
= 15000 + 1620
= 16620

24. An array T [25][20] is stored along the row in the memory with each element requiring 2 bytes of storage. If the base

address of array T is 42000, find out the location of T[l0][15]. Also, find the total number of elements present in this array.

Ans 24 T[i][j]=Base Addr + [i * number of columns+j]* size of each element

T[10][15]= 42000+[(10*20)+15]*2
= 42000+215*2
=42000+430=42430

Total number of elements in array is = 25*20=500

25. An array A[20][30] is stored along the row in the memory with each element requiring 4 bytes of storage. If the base address of array A is 32000, find out the location of A[15][10]. Also, find the total number of elements present in this array.

Ans 25. T[i][j]=Base Addr + [i * number of columns+j]* size of each element

A[15][10]= 32000+[(15*30)+10]*4
= 32000+460*4
=32000+1840=33840

Total number of elements in array is = 20*30=600

26. Given an array A[10][12] whose base address is 10000. Calculate the memory location of A[2][5] if each element occupies 4 bytes and array is stored column-wise.

Ans 26. B=10000, W=4, N=10, I=2, J=5

A[I][J]=B+W[(I-LBR)+(J-LBC)*N]
A[2][5]=10000+4(2+10*5)
=10000+4(52)
=10000+208=10208

27. An array P[15][10] is stored along the column in the memory with each element requiring 4 bytes of storage. If the base address of array P is 14000, find out the location of P[8][5].

Ans 27. B=14000, W=4, N=15, I=8, J=5

P[I][J]=B+W[(I-LBR)+(J-LBC)*N]
P[8][5]=14000+4(8+5*15)
=14000+4(83)
=14000+332=14332

28. An array T[15][10] is stored along the row in the memory with each element requiring 8 bytes of storage. If the base address of array T is 14000, find out the location of T[10][7].

Ans 28. Address of T[i][j]=address of T[0][0]+(i*number of columns present in array +j)*sizeof(element)

Address of T[10][7]=14000+(10*10+7)*8

=14000+(107)*8
=14000+856
=14856

29. An array T[20][10] is stored in the memory along the column with each of the element occupying 2 bytes, find out the memory location of T[10][5], if an element T[2][9] is stored at location 7600.

Ans 29. T[2][9]=Base addr+2[2+9*20]
7600=Base addr+2*(182)
Base addr=7600-364=7236
T[10][5]=7236+2(10+5*20)
= 7236+110*2
= 7456

30. An array P[20] [50] is stored in the memory along the column with each of its element occupying 4 bytes, find out the 1ocation of P[15][10], if P[0][0] is stored at 5200.

Ans 30. Assuminq LBR=LBC=0
B=5200
W=4 bytes
Number of Rows(N)=20
Number of Columns(M)=50
LOC(Arr[I] [J]) = B +(I + J*N)*W
LOC(Arr[15][10]) = 5200+(15+10*20)*4
= 5200 + (215*4)
= 5200 + 860
= 6060

31. An array G[50] [20] is stored in the memory along the row with each of its element occupying 8 bytes, find out the 1ocation of G[10][15], if P[0][0] is stored at 4200.

Ans 31. Assuminq LBR=LBC=0
B=4200
W=8 bytes
Number of Rows(N)=50
Number of Columns(M)=20
LOC(Arr[I] [J]) = B +(I*M + J)*W
LOC(Arr[10][15]) = 4200+(10*20+15)*8
= 4200 + (215*8)
= 4200 + 1720
= 5920

32. An array P[50] [60] is stored in the memory along the column with each of the element occupying 2 bytes, find out the memory location for the element P[10][20],if the Base Address of the array is 6800.

Ans 32. Loc(P[I] [J]) = Base(P)+W(I+J*M)i Loc(P[10][20])
= Base(P)+2(10+20*50)
Loc(P[10] [20]) = 68OO + 2(10+20*50)
= 6800 + 2 (10+1000)
= 6800 + 2*1010
= 6800 + 2020
= 8820
OR
Address of P[i] [j] = BaseAddress+ W((i-L1)+(j-L2)*M)
Address of P[10] [20]= 6800 +2((10-0)+(20-0)x50)
= 6800 + 2 x 1010
= 6800 + 2020
= 8820

33. An array T[90][100] is stored in the memory along the column with each of the elements occupying 4 bytes. Find out the memory location for the element T[10][40],if the Base Address of the array is 7200.

Ans 33. Loc(T[I][J))=Base(T)+W(I+J*N)where N is the number of rows,LBR= LBC = 0)
= 7200 + 4[10 + 40 x 90]
= 7200 + 4[10+3600]
= 7200 + 4 x 3610
= 7200 + 14440
= 21640

34. An array S[40][30] is stored in the memory along the column with each of the element occupying 4 bytes, find out the base address and address of element S[20] [15], if an element S[15][10] is stored at the
memory location 7200.

Ans 34.
Loc(S[I][J]) = Base(S)+W(I+J*N)
Loc(S[15][10]) = Base(S)+4(15+10*40)
Base(S) = 7200-4*415
Base(S) = 7200-1660
Base(S) = 5540
Loc(S[20][15]) =Base(S)+4(20+15*40)
Loc(S[20][15]) = 5540 + 4(20+15*40)
= 5540 + 4(20+600)
= 5540 + 4*620
= 5540 + 2480
= 8020

35. An array Arr[50][10] is store in the memory along the row with each element occupying 2 bytes. Find out the Base address

of the location Arr[20][50], if the location Arr[10][25] is stored at the address
10000.
Ans 35.
Assuming LBR=LBC=0 S=2 bytes
Number of Rows (N)=50
Number of Columns (M)=10
LOC (Arr [I] [J]) = B + (I*M+J)*S
LOC (Arr [10] [25]) = B +(10*10+25)*2
10000 = B +(100+25)*2
B = 10000-250
B = 9750
LOC (Arr [20] [50]) = 9750+(20*10+50)*2
= 9750 + (250*2)
= 9750+500
= 10250

36. An array VAL[1…15][1…10] is stored in the memory with each element requiring 4 bytes of storage. If the base address of the array VAL is 1500, determine the location of VAL[12][9] when the array VAL is stored Row wise.
Ans 36. Given Data: VAL[1…15][1…10]
Word Length (W) = 4 Bytes
Base Address of VAL(B) = 1500
VAL[12][9] = ?
C = Total No of Columns=10
R = Total No of Rows=15
Lr = Least Row=1
Lc = Least Column=1
Row Major:
Address of an element (I,J) in row major = B + W (C (I-Lr) + (J – Lc))
VAL [12][9] = 1500 + 4 (10 * (12-1) + (9-1))
= 1500 + 4 (10 * 11+8)
= 1500 + 4 (118)
= 1500 + 472
= 1972.

37. An array VAL[1…15][1…10] is stored in the memory with each element requiring 4 bytes of storage. If the base address of the array VAL is 1500, determine the location of VAL[12][9] when the array VAL is stored Column wise.
Ans **37.** Given Data: VAL[1…15][1…10]
Word Length (W) = 4 Bytes
Base Address of VAL(B) = 1500
VAL[12][9] = ?
C = Total No of Columns=10

R = Total No of Rows=15
Lr = Least Row=1
Lc = Least Column=1
Column Major:
Address of an element (I,J) in column major
= B + W ((I-Lr) + R(J – Lc))
VAL [12][9] = 1500 + 4 ((12-1) +15 * (9-1))
= 1500 + 4 (11 + 15 * 8)
= 1500 + 4 (11+ 120)
= 1500 + 4 * 131
= 1500 + 524
= 2024.

38. An array S[40][30] is stored in the memory along the **row** with each of the element occupying 2 bytes, find out the memory location for the element S[20][10], if an element S[15][5] is stored at the memory location 5500.
Ans 38. Given, W(Word Length) = 2 R(Number of Rows) =40
C(Number of Columns) =30
Lr = Least Row = 0
Lc = Least Column = 0
Loc(S[15][5]) =5500
Address of an element (I,J) in row major = B + W (C (I-Lr) + (J – Lc))
Loc(S[15][5]= Base(S)+2*(30*(15-0)+(5-0))
5500 = Base(S)+2*(30*15+5)
5500 =Base(S)+2*(450+5)
Base(S) =5500- 910
Base(S) =4590
Loc(S[20][10]) =4590+2*(30*(20-0)+(10-0))
=4590+2*(30*20+10)
=4590+2*(600+10)
=4590+1220
= 5810

39. Write a function REVCOL (int P[][5], int N, int M) in C++to display the content of a two dimensional array, with each column content in reverse order.
Note: Array may contain any number of rows.
For example, if the content of array is as follows:
15 12 56 45 51
13 91 92 87 63
11 23 61 46 81
The function should display output as:
11 23 61 46 81
13 91 92 87 63

15 12 56 45 51

Ans 39.

```
     void REVCOL(int P[][5],int N,int M)
{
for(int I=N1;I>=
0;I)
{
for(int J=0;J<M;J++)
cout<<P[I][J];
cout<<endl;
}
}
```

40. Write a function REVROW(int P[][5],int N, int M) in C++ to display the content of a two dimensional array, with each row content in reverse order.

For example, if the content of array is as follows:

15 12 56 45 51

13 91 92 87 63

11 23 61 46 81

The function should display output as:

51 45 56 12 15

63 87 92 91 13

81 46 61 23 81

Ans 40.

```
  void REVROW(int P[][5],int N,int M)
{
for(int I=0; I<N; I++)
{ for(int J=M1;
J>=0; J)
cout<<P[I][J];
cout<<endl;
}
}
```

41. Write a user-defined function SumLast3(int A[][4], int N, int M) in C++ to find and display the sum of all the values, which are ending with 3 (i.e. Unit place is 3).For example if the content of array is:

33	13	92
99	3	12

The output should be 49

Ans 41.

```
  void SumLast3(int A[][4], int N,int M)
  { int S=0;
       for(int i=0;i<N;i++)
             {
             for( int j=0;j<M;j++)
```

```
                    { int r=A[i][j]%10;
                         if(r==3)
                         S+=A[i][j];
                    }
                    }
                    cout<<S;
               }
```

42. Write a user-defined function AddEnd2(int A[][4],int N,int M) in C++ to find and display the sum of all the values, which are ending with 2 (i.e., units place is 2).

For example if the content of array is:

12	16	32
19	5	2

The output should be 46

Ans 42.
```
     void AddEnd2(int A[][4], int N,int M)
     { int S=0;
          for(int i=0;i<N;i++)
               {
               for( int j=0;j<M;j++)
               { if((A[i][j]%10)==2)
                    S+=A[i][j];
               }
            }  cout<<S;
          }
```

43. Write a function in C++ which accepts a 2D array of integers and its size arguments and displays the elements which lie on minor diagonal. [Top right to bottom left diagonal] [Assuming the 2D array to be square matrix with odd dimension i.e. 3 x 3, 5 x 5, 7, x 7, etc ...]

For example

If the 2D array is

6 7 8
1 3 6
7 9 3

The following should be displayed :

8
3
7

Ans 43.
```
     void show( int a[5][5],int r,int c)
     {
      for(int i=0;i<r;i++)
      { for(int j=0;j<c;j++)
       if((i+j==r-1))
     cout<<a[i][j];
     cout<<endl;
          }
      }
```

44. Write a user defined function display (int A[][4], int N,int M) in C++ to find and display all numbers, which are divisible by 10.
e.g. if the content of array is : 45 50 60
10 3 15
Then output should be 50 60 10

```
Ans 44.   void display(int A[][4], int N,int M)
     { int i,j;
        for( i=0;i<N;i++)
          for(j=0;j<M;j++)
           if( A[i][j]%10==0)
          cout<<A[i][j]<<" ";
     }
```

45. Write a function ALTERNATE (int A[][3],int N,int M) in C++ to display all alternate element from two-dimensional array A (starting from A[0][0]).
e.g. if the Array is containing:

```
                                    12   13   14
                                    15   59   53
                                    35   36   34
```

The output will be:
12 14 59 35 34

```
Ans 45.   void Alternate (int A[][3], int N,int M)
     { int i,j;
        for( i=0;i<N;i++)
         {
     if(i%2==0)
      j=0;
     else j=1;
           While(j<M)
           {   cout<<A[i][j]<<'\t';
               j+=2;
            }
          }
     }
```

46. Write a COLSUM( ) function in C++ to find sum of each column of a NxM Matrix.

```
Ans 46.    void COLSUM(int A[] [100], int N, int M)
{  int SUMC;
for (int j=0; j<M; j++)
{  SUMC = 0;
for (int i=0; i<N; i++)
SUMC = SUMC + A[i] [j] ;
cout<< "Sum of Column "<<j+1<<" = "<<SUMC<<endl ;
}
}
```

47. Write a ROWSUM( ) function in C++ to find sum of each row of a rxc Matrix.

Ans 47.
```
void ROWSUM(int a[] [100], int r, int c)
{ int i,j;
for(i=0;i<r;i++)
{  int s=0;
          for(j=0;j<c;j++)
 s+=a[i][j];
 cout<<"sum="<<s<<endl;
 }
}
```

48. Write a DSUM function in C++ to find the sum of diagonal element of a n*n matrix.

Ans 48.
```
void DSUM( int a[][3], int n)
{  int i,j;
     int s=0;
 for(i=0;i<n;i++)
    s+=a[i][i];
  cout<<"sum of upper diagonal="<<s<<endl;

int s1=0;
 for(i=0;i<n;i++)
 for(j=0;j<n;j++)
if((i+j==n-1))
     s1+=a[i][j];
  cout<<"sum of lower diagonal="<<s1;

}
```

49. Write a user defined function in C++ to display the multiplication of row elements of two dimensional array A[4][6] containing integers.

Ans 49.
```
void RowMulti(int A[4][6])
{ int Mul[4];
for(int i=0;i<4;i++)
{ Mul[i]=1;
for(int j=0;j<6;j++)
Mul[i]*=A[i][j];
cout<<"Product of row"<<i+1<<"="<<Mul[i]<<endl;
}
}
```

50. Write a user defined function in C++ to display the sum of row elements of two dimensional array A[5][6] containing integers.

Ans 50.
```
void RowSum(int A[5][6])
{ int SUMC[5];
```

```
for(int i=0;i<5;i++)
{ SUMC[i]=0;
for(int j=0;j<6;j++)
SUMC[i]+=A[i][j];
cout<<"Sum of row"<<i+1<<"="<< SUMC[i]<<endl;
}
}
```

51. Write a function in C++ to print the product of each column of a two dimensional integer array passed as the argument of the function.
Explain: if the two dimensional array contains
1 2 4
3 5 6
4 3 2
2 1 5
Then the output should appear as:
Product of Column 1 = 24
Product of Column 2 = 30
Product of Column 3 = 240

Ans 51.
```
void ColProd(int A[4][3],int r,int c)
{ int Prod[C],i,j;
for(j=0;j<c;j++)
{ Prod[j]=1;
for(i=0;i<r;i++)
Prod[j]*=A[i][j];
cout<<"Product of Column" <<j+1<<"="<<Prod[j]<<endl;
}
}
```

52. Write a function in C++ which accepts a 2D array of integers and its size as arguments and display the elements which lie on diagonals.
[Assuming the 2D Array to be a square matrix with odd dimension i.e., 3 x 3, 5 x 5, 7 x 7 etc….]
Example, if the array content is
5 4 3
6 7 8
1 2 9
Output through the function should be:
Diagonal One: 5 7 9
Diagonal Two: 3 7 1

Ans 52.
```
const int n=5;
void Diagonals(int A[n][n], int size)
{
int i,j;
cout<<"Diagonal One:";
for(i=0;i<n;i++)
```

```
cout<<A[i]ij]<<" ";
cout<<"\n Diagonal Two:"
for(i=0;i<n;i++)
cout<<A[i][n-(i+1)]<<" ";
}
```

53. Write a function in C++ which accepts a 2D array of integers and its size as arguments and display the elements of middle row and the elements of middle column.
[Assuming the 2D Array to be a square matrix with odd dimension i.e., 3 x 3, 5 x 5, 7 x 7 etc….]
Example, if the array content is
3 5 4
7 6 9
2 1 8
Output through the function should be:
Middle Row: 7 6 9
Middle Column: 5 6 1
Ans 53.

```
const int S=7; // or it may be 3 or 5
int DispMRowMCol(int Arr[S][S],int S)
{ int mid=S/2; int i; //Extracting middle row
cout<<"\n Middle Row:";
for(i=0;i<S;i++)
cout<<Arr[mid][i]<<" ";
//Extracting middle column
cout<<"\n Middle Column:";
for(i=0;i<S;i++)
cout<<Arr[i][mid]<<" ";
}
```

54. Write a function int ALTERSUM (int B[ ][5], int N, int M in C++ to find and return the sum of elements from all alternate elements of a two-dimensional array starting from B[0][0].
Ans 54.

```
     int ALTERSUM(int B[ ][5] ,int N,int M)
{
int Sum=0;
for (int I=0;I<N;I++)
for (int J=(I%2==0)?0:1;J<M;J+=2)
Sum+=B[I][J] ;
return Sum;
}
```