output--

The value of a[0] = 1
The value of a[1] = 2
The value of a[2] = 3
The value of a[3] = 4
The value of a[4] = 5

Input-

Program-1

```
// Program to print all elements of an array.
#include <stdio.h>
#include <conio.h>
void main()
{
    int  a[5] = {1, 2, 3, 4, 5};
    clrscr();
        for(i=0; i<=5-1; i++)
        {
            printf(" The value of a[%d] = %d\n", i, a[i]
        }
    getch();
}
```

Output-

The value of a[0] = 1
The value of a[1] = 2
The value of a[2] = 3
The value of a[3] = 4
The value of a[4] = 5
The array is given below :->>
1 2 3 4 5

Input

Program - 2

```
// Program to input / output in an array
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[5];
  clrscr();
    for(i=0; i<=5-1; i++)
    {
     printf(" The value of a[%d] = ", i);
        scanf(" %d", &a[i]);
    }
  printf(" The array is given below :->> \n ");
  for(i=0; i<=5-1; i++)
   {
    printf(" %d  ", a[i]);
   }
  getch();
}
```

output-

```
Enter the size of an array = 5
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
The array is given below :->>

1
2
3
4
5
```

Input

Program-3

```
/* Program to input/output elements of an array using
                                    function. */

#include <stdio.h>
#include <conio.h>
/* function declaration */
void inputarray(int a[],int n);
void outputarray(int a[],int n);
void main()
{
  int  n, a[10], i;
  clrscr();
    printf(" Enter the size of an array = ");
        scanf(" %d", &n);
  inputarray(a,n);          /* function call */
  printf(" The array is given below :->> \n\n");
  outputarray(a,n);         /* function call */
  getch();
}
/* function definition of inputarray */
void inputarray(int a[], int n)
{
  int i;
   /* logic for input data in an array */
   for(i=0; i<=n; i++)
   {
     printf(" Input a[%d] value = ", i);
        scanf(" %d ", &a[i]);
   }
}
/* function definition of an outputarray */
void outputarray(int a[], int n)
{
  int i;
```

```
/* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n %d", a[i]);
    }
}
```

Input

Program-4

```
/* Input & Output multiple array using function*/
#include <stdio.h>
#include <conio.h>
void inputarray(int [], int);
void outputarray(int [], int);
void main()
{
  int n, a[10], b[10], c[10], m, o;
  clrscr();
   printf(" Enter the size of Ist array = ");
      scanf(" %d", &n);
   printf(" Enter the size of IInd array = ");
      scanf(" %d", &m);
   printf(" Enter the size of IIIrd array = ");
      scanf(" %d", &o);
  inputarray(a,n);                /* function call */
  inputarray(b,m);                /* function call */
  inputarray(c,o);                /* function call */
  printf(" The first array is given below :->\n");
  outputarray(a,n);               /* function call */
  printf(" The second array is given below :-> \n");
  outputarray(b,m);               /* function call */
  printf(" The third array is given below :-> \n");
  outputarray(c,o);               /* function call */
  getch();
}
void inputarray(int a[], int n)
{
  int i;
   for(i=0 ; i <= n-1 ; i++)
   {
      printf(" Input a[%d] value = ", i);
         scanf(" %d", &a[i]);
   }
```

```
void outputarray(int a[],int n)
{
  int i;
    for(i=0; i<=n; i++)
    {
      printf("%d\n", a[i]);
    }
}
```

Output-

Enter the size of 1st array = 2
Enter the size of IInd array = 2
Enter the size of IIIrd array = 2
Input a[0] value = 1
Input a[1] value = 2
Input a[0] value = 1
Input a[1] value = 2
Input a[0] value = 1
Input a[1] value = 2
The first array is given below :->
1
2
The Second array is given below :->
1
2
The Third array is given below :->
1
2

Input-

Program-5

```
/* Program to sum all elements of an array using
                    function */

#include<stdio.h>
#include<conio.h>
void inputarray(int [],int);
void outputarray(int [],int);
void sumarray(int [],int);      /* function declaration */
void main()
{
 int n, a[10], i;
 clrscr();
   printf("Enter the size of array=");
      scanf("%d", &n);
 inputarray(a,n);                /*functionat call */
   printf(" The array is given below :->\n");
 outputarray(a,n);               /*function call */
 sumarray(a,n);                  /* function call */
 getch();
}

void inputarray(int a[],int n);
{
 int i;
   for(i=0; i<=n-1; i++)
   {
     printf(" Input a[%d] value = ", i);
       scanf("%d", &a[i]);
   }
}

void outputarray(int a[], int n)
{
 int i;
   for(i=0; i<=n-1; i++)
   {
```

```
        printf(" %d\n", a[i]);
    }
}
/* function definition of sum all elements of an array
void sumarray(int a[], int n)
{
  int i, sum=0;
      for(i=0; i<=n-1; i++)
      {
        sum = sum + a[i];
      }
    printf(" The sum of all elements of array =%d", sum)
}
```

Output-

```
Enter the size of an array = 5
 Input a[0] value = 1
 Input a[1] value = 2
 Input a[2] value = 3
 Input a[3] value = 4
 Input a[4] value = 5
The array is given below :->
 1
 2
 3
 4
 5
The sum of all elements of array = 15
```

Input

Program - 6

```
/* Program to find average of all elements of an
      array */
# include<stdio.h>
#include <conio.h>
/* function declaration */
void inputarray(int a[],int n);
void outputarray(int a[],int n);
int sumarray(int a[],int n);
int averagearray(int, int);
void main()
{
  int a[10], n, sum, average;
  clrscr();
    printf(" Enter the size of an array= ");
      scanf(" %d", &n);
inputarray(a,n);                    /*function call */
    printf(" The array is given below :->\n");
outputarray(a,n);                   /*function call */
sum= sumarray(a,n);                 /* function call */
average= averagearray(sum,n);       /* function call */
   printf("\n The average of an array= %d \n", average);
getch();
}

void inputarray(int a[],int n)
{
  int i;
    for(i=0; i<= n-1; i++)
    {
      printf(" Input a[%d] value= ");
        scanf(" %d", &a[i]);
    }
}
void outputarray(int a[],int n)
```

```
{
  int i;  for (i=0; i<=n-1; i++)
  { printf("%d\n", a[i]);
  }
}
/* function definition to finding average */
void averagearray(int n, int sum)
{
  int av;
      av = sum/n;
  return (av);
}
/* function definition to sum all elements of an array*/
void sumarray(int a[], int n)
{
  int i, j;
      for (i=0; i<=n-1; i++)
      {
        for (j=0; j<=c-1; j++)
        {
          sum = sum + a[i];
        }
      }
  return(sum);
}
```

Output-

```
Enter the size of an array = 5
Input a[0] = 1
Input a[1] = 2
Input a[2] = 3
Input a[3] = 4
Input a[4] = 5
The array is given below:->>
```

1
2
3
4
5

The average of an array = 15

Program-7

```
/* Program to find minimum element of an array */
#include <stdio.h>
#include <conio.h>
/* function declaration */
void inputarray(int a[],int n);
void outputarray(int a[],int n);
void min(int a[],int n);
void main()
{
    int n,a[10],i;
    clrscr();
      printf(" Enter the size of array = ");
        scanf(" %d",&n);
    inputarray(a,n);                /* function call */
    printf(" The array is given below :-> \n\n");
    outputarray(a,n);               /* function call */
    min(a,n);                       /* function call */
  getch();
}
/* function definition of array input */
void inputarray(int a[],int n)
{
    int i;
    /* logic for input data in an array */
     for(i=0; i<=n-1; i++)
     {
        printf(" Input a[%d] value = ",i);
         scanf(" %d",&a[i]);
     }
  }
/* function definition output array
void outputarray(int a[],int n)
{
```

```
/* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n %d", a[i]);
    }
}
/* function definition minimum element */
void min(int a[], int n)
{
  int i, min;
    min = a[0];
   for(i=1; i<=n-1; i++)
   {
       if( a[i] < min)
       {
           min = a[i];
       }
   }
   printf("\n The Minimum element of array
                                    = %d", min);
}
```

output:

```
Enter the size of an array = 5
 Input a[0] value = 1
 Input a[1] value = 2
 Input a[2] value = 3
 Input a[3] value = 4
 Input a[4] value = 5
The array is given below :->

 1
 2
```

3
4
5

The minimum element of array = 1

Program-8

```
// Program for linear search
/* Program to perform linear search in an array
   using function */
#include <stdio.h>
#include <conio.h>
/* function declaration */
void inputarray(int a[], int n);
void outputarray(int a[], int n);
int linearsearch(int a[], int n, int item);
void main()
{
  int n, a[10], item, loc;
  clrscr();
   printf(" Enter the size of array = ");
      scanf(" %d", &n);
  inputarray(a,n);        /* function call */
  printf(" The array is given below :-> \n ");
  outputarray(a,n);       /* function call */
   printf(" \n Enter the item to search=");
    scanf(" %d", &item);
  loc = linearsearch(a,n, item);  /* function call */
  if(loc!=1)
   printf(" \n Item is found at = %d location", loc);
  else
   printf(" \n Search is unsuccessful; item is not
                                        found ");
  getch();
}
/* function definition of arrayinput */
void inputarray(int a[], int n)
{
  int i;
  /* logic for input data in an array */
```

```
for(i=0 ; i<=n-1; i++)
{
    printf(" Input a[%d] value = ", i);
      scanf(" %d", &a[i]);
  }
}
/* function definition output array */
void outputarray(int a[], int n)
{
  /* logic for output data from an array */
  int i;
    for(i=0; i<=n-1; i++)
    {
      printf(" \n %d ", a[i]);
    }
}
/* function definition linear search */
int linearsearch (int a[], int n, int item)
{
  int i;
    for(i=0; i<=n-1; i++)
    {
      if (a[i]==item)
        return(i);
    }
  return(-1);
}
```

Output-

Enter the size of array = 5
Input [0] value = 1
Input [1] value = 2
Input [2] value = 3
Input [3] value = 4
Input [4] value = 5.
The array is given below :->
1
2
3
4
5
Enter the item to search = 3
Item is found at = 2 location

Program-9

```
// Program for Binary search
/* Program to perform binary search in an
   array using function */
#include <stdio.h>
#include <conio.h>
/* function definition */
void inputarray(int a[],int n);
void outputarray(int a[],int n);
int binarysearch(int a[],int lb,int ub,int item);
void main()
{
  int n,a[10],item, loc;
  clrscr();
   printf(" Enter the size of an array =");
     scanf("%d", &n);
 inputarray(a,n);              /* function call */
   printf(" The array is given below :-> \n");
 outputarray(a,n);             /* function call */
   printf(" \n Enter the item to search =");
     scanf("%d", &item);
 loc= binarysearch(a,0,n-1,item);   /* function call */
   if( loc != -1)
   printf(" \n Item is found at = %d location ", loc);
   else
   printf(" \n Search is unsuccessful, item is not
                                        found ");
 getch();
}
/* function definition of array input */
void inputarray(int a[], int n)
{
  int i;
/* logic for input data in an array */
```

```
    for(i=0; i<=n-1; i++)
        printf("Input a[%d] value=",i);
          scanf("%d", &a[i]);
}
/* function definition output array */
void outputarray(int a[], int n)
{
/* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("%d", a[i]);
    }
}
/* function definition binary search */
int binarysearch(int a[], int lb, int ub, int item)
{
 int mid;
 while (lb<=ub)
 {
   mid = (lb+ub)/2;
        if(a[mid]==item)
        return (mid);
        else
         if(item<a[mid])
          ub =mid-1;
         else
          lb = mid+1;
 }

 return (-1);
}
```

output -

Enter the size of an array = 5
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
The array is given below :->
1
2
3
4
5
Enter the item to search = 3
Item is found at = 2 location.

Program - 10

```
/* Program to find the occurances of item in an
   array using function */
#include <stdio.h>
#include <conio.h>
/* function declaration */
void inputarray(int a[], int n);
void outputarray(int a[], int n);
void occurance(int a[], int n, int item);
void main()
{
 int n, a[10], item, loc;
 clrscr();
      printf(" Enter the size of array=");
          scanf(" %d", &n);
inputarray(a,n);          /* function call */
printf(" The array is given below :->\n");
outputarray(a,n);         /* function call */
  printf("\n Enter the item to search = ");
    scanf(" %d", &item);
occurances(a,n,item);
getch();
 }
/* function definition of arrayinput */
void inputarray(int a[], int n)
{

 int i;
/* logic for input data in an array */
     for(i=0; i<=n-1; i++)
     {
       printf(" Input a[%d] value = ", i);
          scanf("%d ", &a[i]);
     }
}
```

```
/* function definition output array */
void outputarray(int a[],int n)
{
    /* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n%d", a[i]);
    }
}
/* function definition to find the occurance of item
    in an array */
void occurance (int a[], int n,int item)
{
    int i, count=0;
        for(i=0 ; i<= n-1; i++)
        {
            if(a[i] == item)
            {
                count++;
                printf("\n item is found at loc/index =%d\n",
            }
        }
    if (count==0)
    {
        printf(" Item is not found ");
    }
    else
    {
    printf(" The occurance of item is = %d", count);
    }
}
```

Date
Page

output

Enter the size of an array = 5
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 2
Input a[4] value = 2
The array is given below :->
1
2
3
2
2
Enter the item to search = 2
Item is found at loc/index = 1
Item is found at loc/index = 3
Item is found at loc/index = 4
The occurance of item is = 3.

Program-11

```
/* Program to perfor insertion operation in an
   array using function.
#include <stdio.h>
#include <conio.h>
# define MAX 10
/* function definition */
void inputarray (int a[], int n);
void outputarray (int a[], int n);
int insert (int a[], int n, int item, int loc);
void main()
{
  int n, a[MAX], item, loc;
  clrscr();
     printf(" Enter the size of an array = ");
        scanf(" %d", &n);
inputarray (a,n);              /* function call */
  printf(" The array is given below :->>\n");
outputarray (a,n);             /* function call */
  printf(" Enter the item to insert= ");
     scanf(" %d", &item);
printf(" \n Enter the loc/index at w/c you want to insert
     scanf(" %d", &loc);
  n= insert (a,n,item,loc);    /* function call */
printf(" \n After insertion array is:->");
  outputarray (a,n);
  getch();
}
/* function definition of array input */
void inputarray (int a[], int n)
{
  int i;
/* logic for input data in an array */
     for(i=0; i<=n-1; i++)
```

```
    printf(" Input a[%d] value = ");
        scanf(" %d ", &a[i]);
  }
/* function definition output array */
void outputarray ( int a[], int n)
{
 /* logic for output data from an array */
  int i;
      for(i=0; i<=n-1; i++)
          printf(" \n %d ", a[i]);
}
/* function definition for insertion item in an array*/
int insert( int a[], int n, int item, int loc)
{
 /* logic for insertion an item in an array. */
  int i;
      if(n== MAX)
      {
        printf(" \n Overflow , Array is full !!! Can't insert new
                                        element ");
        return(n);
      }
      else
      {
        for(i=n-1; i >= loc ; i--)
        {
          a[i+1]= a[i];
        }
          a[loc]= item;
          n=n+1;
        return(n);
}
```

Output-

Enter the size of an array = 5.
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
The array is given below :->
1
2
3
4
5
Enter the item to insert = 7
Enter the loc / index at w/c you want to insert = 3
After insertion array is :->
1
2
3
7
4
5

Program-12

```
// Insertion by searching
/* Program to perform insertion operation before
   an array element using function */
#include<stdio.h>
#include<conio.h>
#define MAX 10
/* Function declaration */
void inputarray(int [],int);
void outputarray(int [],int);
int linearsearch (int [],int,int);
int insert(int [],int,int,int);
void main()
{
 int a[MAX], n, item, loc, before;
 clrscr();
     printf(" Enter the size of an array = ");
        scanf("%d", &n);
 inputarray(a,n);    /* Function call */
    printf(" The array is given below :-> \n\n");
 outputarray (a,n);    /* Function call */
    printf(" Enter the item before you want to insert=");
      scanf("%d", &before);
    printf(" Enter the item to be insert in an array=");
        scanf("%d", &item);
    loc = linearsearch (a,n,before);    /* Function call */
    n= insert (a,n, item, loc);    /* Function call */
    printf("\n After insertion array is ");
 outputarray (a, n);    /* Function call */
getch();
}

/* Function definition of array input */
void inputarray( int a[],int n)
{
```

```
/* logic for input data in an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf(" Input a[%d] value= ", i);
        scanf(" %d", &a[i]);
    }
}

/* Function definition output array */
void outputarray(int a[], int n)
{
/* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n %d", a[i]);
    }
}

/* Function definition for linear search of location's item
   for inserting item */
int linearsearch(int a[], int n, int before)
{
/* logic for linear search of location's item for
   inserting item */
int i;
    for(i=0; i<=n-1; i++)
    {
        if(a[i]==before)
        {
            return(i);
        }
    }
    return(-1);
}
```

```
/* Function definition for insert item in an array
int insert(int a[],int n,int item,int loc)
{
/* logic for insertion item in an array */
 int i;
    if(n==MAX)
    {
      printf("\n overflow, Array is full!!! Can't
                insert new element");
    }
    else
    {
      for(i=n-1; i>=loc; i--)
      {
          a[i+1]= a[i];
      }
          a[loc] = item;
           n=n+1;
          return(n);
}
```

Output-

Enter the size of an array = 5
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
The array is given below :->

1
2
3
4
5

Enter the item before you want to insert = 4
Enter the item to be insert in an array = 7
After insertion the array is :
1
2
3
7
4
5

Program-13

```
/* Multiple insertion to check overflow */
#include<stdio.h>
#include<conio.h>
# define MAX 10
/* Function definition */
void inputarray(int [],int);
void outputarray(int [],int);
int insert(int [], int, int, int);
void main()
{
  int n,a[MAX], item, loc;
  char ch;
  clrscr();
    printf(" Enter the size of an array = ");
      scanf("%d", &n);
inputarray(a,n);                /* Function call */
   printf(" The array is given below :-> \n");
outputarray(a,n);               /* Function call */
do {
  printf("\n Enter the item to insert = ");
    scanf("%d",&item);
printf(" Enter the location/index at w/c you want to
                                insert=");
      scanf("%d", &loc);
  n= insert(a,n,item,loc);      /*Function call*/
printf("\n After insertion array is =");
outputarray(a,n);               /* Function call */
printf("\n Do you want to insert more item = ");
        fflush(stdin);
        scanf("%c", &ch);
} while(ch=='y');
getch();
}
```

```
/* Function definition of array input */
void inputarray(int a[],int n)
{
/* logic for input data in an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf(" Input a[%d] value = ");
            scanf(" %d", &a[i]); }
}
/* Function definition of output array */
void outputarray(int a[], int n)
{
/* logic for output data from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n%d", a[i]);
    }
}
/* Function definition for insertion item in an
    array */
int insert(int a[],int n,int item,int loc)
{
/* logic for insertion an item in an array */
int j;
    if(n==MAX)
    {
    printf("\n overflow, Array is full!!! Can't
                insert new element");
    return(n);
    }
    else
```

```
{
  for (i=n-1; i>=loc; i--)
   {
     a[i+1] = a[i];
   }
   a[loc] = item;
    n=n+1;
   return(n);
}
```

output-

```
Enter the size of an array = 8
Input a[0] value= 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value= 4
Input a[4] value = 5
Input a[5] value = 6
Input a[6] value = 7
Input a[7] value = 8
The array is given below :->
1
2
3
4
5
6
7
8
Enter the item to insert = 11
Enter the location/index at w/c you want to insert = 3
After insertion array is =
```

1
2
3
11
4
5
6
7
8
Do you want to insert more item = y
Enter the item to insert = 12
Enter the location/index at w/c you want to insert = 6
After insertion array is =
1
2
3
14
4
5
12
6
7
8
Do you want to insert more item = y
Enter the item to insert = 13
Enter the location/index at w/c you want to insert = 2
After insertion array is =
1
2
13
3
11
4

5
12
6
7
8

Do you want to insert more item = y
Enter the item to insert = 14
Enter the location/index at w/c you want to insert = 1
Overflow, Array is full !!! Can't insert new elemen
After insertion array is =
1
2
13
3
11
4
5
12
6
7
8

Do you want to insert more item = n

Program-14

```
/* Deletion item in an array */
/* Program to perform deletion operation in an
   array using function */
#include<stdio.h>
#include<conio.h>
#define MAX 10
/* Function definition */
void inputarray(int [],int);
void outputarray(int [],int);
int deletionarray(int [], int, int);
void main()
{
  int n, a[MAX], loc;
  clrscr();
     printf(" Enter the size of an array = ");
       scanf("%d", &n);
inputarray(a,n);                    /* Function call*/
  printf(" The array is given below :-> \n");
outputarray(a,n);                   /* Function call */
  printf("\n Enter the loc/index at wlc you want to delete=");
  scanf("%d", &loc);
  n = deletionarray(a,n,loc);       /* Function call */
  printf("\n After deletion array is = ");
outputarray(a,n);
  getch();
}
/* Function definition of array input */
void inputarray(int a[], int n)
{
  /* logic for input in an array */
  int i;
     for(i=0; i<=n-1; i++)
     {
```

```
        printf(" Input a[%d] value = ", i);
            scanf(" %d ", &a[i]);
        }
}
/* Function definition of output array */
void outputarray(int a[], int n)
{
/* logic for output from an array */
int i;
        for(i=0; i<=n-1; i++)
        {
            printf("\n%d ", a[i]);
        }
}
/* Function for deletion */
int deletionarray(int a[], int n, int loc)
{
  int i;
        if(n==0)
        {
          printf("\n Underflow, Array is empty!! Can't delete
          return (n);                                  element ");
        }
        else
        {
            for(i=loc; i<=n-1; i++)
            {
                a[i]=a[i+1];
            }
            n=n-1;
            return(n);
        }
}
```

output-

Enter the size of an array = 6
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
Input a[5] value = 6
The array is given below :->
1
2
3
4
5
6
Enter the loc/index at w/c you want to delete = 2
After deletion array is =
1
2
4
5
6

Program- 15

```
/* Delete an item by choose an array elem
#include<stdio.h>
#include<conio.h>
#define MAX 10
/* Function definition */
void inputarray(int [],int);
void outputarray(int [],int);
int linearsearch(int [], int,int);
int deletionarray(int [],int,int);
void main()
{
  int n,a[MAX],item,loc;
  clrscr();
   printf(" Enter the size of an array = ");
      scanf("%d", &n);
  inputarray(a,n);          /* Function call*/
  printf(" The array is given below :-> \n");
  outputarray(a,n);         /* Function call */
  printf(" Enter the array element to be delete=");
      scanf("%d", &item);
loc = linearsearch(a,n,item);      /* Function call*/
      printf("After delete an item, The array is = ");
  n= deletionarray(a,n,loc);     /* Function call*/
  outputarray(a,n);              /* Function call*/
getch();
}

/* Function definition of input array */
void inputarray(int a[],int n)
{
/* logic for input in an array */
int i;
      for(i=0; i<=n-1; i++)
      {
```

```
        printf("~~%d~~ Input a[%d] value = ",i);
            scanf("%d", &a[i]);
    }
}
/* Function definition of output array */
void outputarray(int a[], int n)
{
/* logic for output from an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("\n%d", a[i]);
    }
}
/* Function definition for deletion an item. */
int linearsearch(int a[], int n, int item)
{
/* logic for delete an item in an array */
int i;
    for(i=0; i<=n-1; i++)
    {
        if(a[i]==item)
        {
            return(i);
        }
    }
    return(-1);
}
/* Function definition for deletion
int deletionarray(int a[], int n, int loc)
{
/* logic for deletion
int i;
```

```
if(n==0)
{
 printf("\n Underflow, Array is empty!!! Can't delete element");
 return(n);
}
else
{
   for(i=0; i<=n-1; i++)
   {
      a[i]= a[i+1];
   }
   n=n-1;
   return(n);
}
}
```

Output-

```
Enter the size of an array = 6
Input a[0] value = 1
Input a[1] value = 2
Input a[2] value = 3
Input a[3] value = 4
Input a[4] value = 5
Input a[5] value = 6
The array is given below :->
1
2
3
4
5
6
Enter the element to be delete = 5
After delete an item, the array is =
1
2
3
4
6
```

Program - 16

```
/* Multiple deletion to check underflow */
#include <stdio.h>
#include <conio.h>  include
#define <process.h>
define MAX 10
/* function declaration */
void inputarray (int [], int );
void outputarray(int [], int);
int deletearray(int [], int, int);
void main( )
{
int a[MAX], n, loc;
char ch;
clrscr( );
printf(" Enter the size of an array = ");
    scanf("%d", &n);
inputarray(a,n);          /* function call */
printf(" The array is given below:->\n");
outputarray (a,n);        /* function call*/
do{
printf(" Enter the loc \index to delete = ");
  scanf("%d ", &loc);
n= deletionarray(a,n,loc);     /* function call*/
  printf(" After deletion array is = ");
outputarray (a, n);
printf(" Do you want to delete another item = ");
        fflush (stdin);
        scanf("%c ", &ch);
} while (ch == 'y');
getch( );
}
/* function definition of array input */
void inputarray(int a[], int n)
{
```

```
/* logic for input data in an array */
int i=0;
    for(i=0; i<=n-1; i++)
    {
    printf("Input a[%d] value = ", i);
      scanf("%d", &a[i]);
    }
}
/* function definition output array */
void outputarray(int a[], int n)
{
  /* logic for output data from an array */
  int i=0;
      for(i=0; i<=n-1; i++)
      {
        printf("%d", a[i]);
      }
}
/* function definition for deletion */
int deletion (int a[], int n, int loc)
{
    int i;
      if(n==0)
      {
      printf("\n Underflow, Array is empty!! Can't
                    delete element");
      return(n);
    }
    else
      for (i=loc; i<=n-1; i++)
          a[i] = a[i+1];
    n=n-1;
    return(n);
}
```

Output-

Enter the size of an array = 2
Input a[0] value = 1
Input a[1] value = 2
The array is given below :->
1
2

Enter the loc \index to delete = 1
After deletion array is =
1

Do you want to delete another item = y
Enter the loc \index to delete = 0
After deletion array is =
Underflow, Array is empty!! Can't delete element.

Program - 17

```
/* Program to sort an array by Selection sort
        using function */
#include <stdio.h>
#include <conio.h>
#define MAX 10
/* function declaration */
void inputarray(int [],int);
void outputarray(int [],int);
void selectionsort(int [],int);
void main()
{
  int n, a[MAX];
  clrscr();
    printf(" Enter the size of an array = ");
      scanf("%d", &n);
  inputarray(a, n);                    /*function call */
  printf(" The array before Sorting is given below:\n");
  outputarray(a, n);                   /* function call*/
  selectionsort(a,n);                  /*function call*/
  printf(" In After sorting array is = ");
  outputarray(a,n);                    /*function call */
  getch();
}
/* Function definition of array input */
void inputarray(int a[],int n)
{
/* logic for array input */
int i;
  for(i=0; i<=n-1; i++)
  {
    printf(" Input a[%d] value = ");
      scanf("%d", &a[i]);
  }
```

```
/* Function definition for array output */
void outputarray(int a[], int n)
{
/* logic for output array */
int i;
    for(i=0; i<= n-1; i++)
    {
        printf("\n %d", a[i]);
    }
}
/* Function definition for selection sort */
void selectionsort(int a[], int n)
{
    int i, j, min, loc, temp;
    for(i=0; i<= n-2; i++)
    {
        min = a[i]; loc = i;
        for(j=i+1; j<= n-1; j++)
        {
            if(a[j] < min)
            {
                min = a[j];
                loc = j;
            }
        }
    temp = a[i];
    a[i] = a[loc];
    a[loc] = temp;
    }
}
```

Output -

Enter the size of an array = 5
Input a[0] value = 3
Input a[1] value = 4
Input a[2] value = 1
Input a[3] value = 5
Input a[4] value = 2
The array before sorting is given below :->
3
4
1
5
2
After sorting array is =
1
2
3
4
5

Program-18

```
/* Program to sort an array by bubble sort
                using function */
#include <stdio.h>
#include <conio.h>
# define MAX 10
/* Function declaration */
void inputarray (int [ ], int );
void outputarray (int [ ], int );
void bubblesort (int [ ], int);
void main( )
{
  int n, a[MAX];
  clrscr( );
  printf(" Enter the size of an array = ");
      scanf("%d", &n);
  inputarray(a, n);                    /*function call */
  printf(" The array before sorting is given below => \n")
  outputarray(a, n);                   /*Function call */
  bubblesort(a, n);                    /* Function call */
  printf("\n After sorting array is = ");
  outputarray(a, n);                   /* Function call */
  getch( );
}
/* Function definition of array input */
void inputarray (int a[ ], int n)
{
/* logic for array input */
  int i;
    for (i=0; i<=n-1; i++)
    {
      printf(" Input value a[%d] = ", i);
      scanf("%d", &a[i]);
    }
}
```

```
/* Function definition for array output */
void outputarray(int a[], int n)
{
/* logic for array output */
int i;
    for(i=0; i<=n-1; i++)
    {
        printf("%d\n", a[i]);
    }
}
/* Function for bubble sort */
void bubblesort(int a[], int n)
{
int i, j, temp;
    for(i=0; i<=n-2; i++)
    {
        for(j=0; j<n-i-1; j++)
        {
            if(a[j+1] < a[j])
            {
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
}
```

output-

Enter the size of an array = 5
Input value a [0] = 7
Input value a [1] = 3
Input value a [2] = 1
Input value a [3] = 4
Input value a [4] = 2
The array before sorting is given below :->
7
3
1
4
2
After sorting array is =
1
2
3
4
7

Program-19

```
/* Program to sort an array by insertion Sort
                using function */
#include<stdio.h>
#include <conio.h>
#define MAX 10
void inputarray(int [],int);          /*Function Declartion*/
void outputarray(int [],int);
void insertionsort(int [],int);
void main()
{
   int a[MAX],n;
   clrscr();
        printf(" Enter the size of an array=");
            scanf("%d",&n);
   inputarray(a,n);              /* function call */
      printf(" Before sorting, The array is given below:->\n")
   outputarray(a,n);             /*Function call*/
      printf("\nArray after sorting ");
   insertionsort(a,n);           /*Function call */
   outputarray(a,n);             /* Function call*/
   getch();
}

/* Function definition for array input */
void inputarray(int a[],int n)
{
/* logic for input array */
  int i;
        for(i=0; i<=n-1; i++)
        {
            printf(" Enter a[%d] value= ",i);
                scanf("%d", &a[i]);
        }
}
```

```
/* Function definition for array output */
void outputarray(int a[],int n)
{
/* logic for outputarray */
int i;
    for(i=0; i<=n-1; i++)
    {
      printf("\n %d", a[i]);
    }
}
/* Function definition for insertion sort */
void insertionsort(int a[],int n)
{
/* logic for insertion sort */
int i, j, temp;
    For(i=0 ; i<=n-1; i++)
    {
      temp = a[i]
      j= i-1;
      while ((temp < a[j]) && (j >=0))
      {
        a[j+1] = a[j];
        j--;
      }
      a[j+1] = temp;
    }
}
```

Output-

Enter the size of an array = 5
Enter a [0] value = 7
Enter a [1] value = 9
Enter a [2] value = 1
Enter a [3] value = 0
Enter a [4] value = 2
Before sorting, The array is given below :->
7
9
1
0
2
Array after sorting
0
1
2
7
9

Program - 20

```
/* Program to perform merging of two array
   using function */
#include <stdio.h>
#include <conio.h>
 #define MAX 10
/* Function declaration */
void inputarray(int a[], int n);
void outputarray(int a[], int n);
void mergingarray(int a[], int n, int b[], int m, int c[]);
void main()
{
  int n, a[MAX], m, b[MAX], c[MAX+MAX], item, loc;
  clrscr();
    printf(" Enter the size of first array = ");
      scanf(" %d", &n);
  inputarray(a, n);          /* Function call */
    printf(" Enter the size of second array = ");
  ~~inputarray(b~~ scanf(" %d", &m);
  inputarray(b, m);          /* Function call */
   printf("\n The first array is given below :->\n");
  outputarray(a, n);         /* Function call */
   printf("\n The second array is given below :->\n");
  outputarray(b, m);         /* Function call */
   printf("\n After merging merged array is\n");
  mergingarray(a, n, b, m, c);  /* Function call */
  outputarray(c, n+m);       /* Function call */
  getch();
}
/* Function definition of array input */
void inputarray(int a[], int n)
{
  int i;
   /* logic for input data in an array */
```

```
    for(i=0; i<=n-1; i++)
    {
      printf(" Input a[%d] value = ", i);
        scanf("%d", &a[i]);
    }
}
/* Function definition output array */
void outputarray(int a[],int n)
{
 /*logic for outputarray */
 int i;
    for(i=0; i<=n-1; i++)
    {
      printf("\n%d", a[i]);
    }
}
/* Function for merging */
void merging(int a[],int n,int b[],int m,int c[])
{
 int i=0, j=0, k=0, l;
    while((i<n) && (j<m))
    {
      if (a[i] < b[j])
      {
          c[k] = a[i];
          k++;
          i++;
      }
      else
      {
          c[k]= b[j];
          k++;
          j++;
```

```
        }
    }
    for(l=i; l<n; l++)
    {
        c[k] = a[l];
        k++;
    }
    for(l=j; l<m; l++)
    {
        c[k] = b[l]
        k++;
    }
}
```

<u>output</u>-

Enter the size of first array = 5
Input [0] value = 1
Input [1] value = 3
Input [2] value = 5
Input [3] value = 7
Input [4] value = 9
Enter the size of second array = 5
Input [0] value = 2
Input [1] value = 4
Input [2] value = 6
Input [3] value = 8
Input [4] value = 10

The first array is given below :->

1
3

5
7
9

The Second array is given below :->

2
4
6
8
10

After merging merge array is

1
2
3
4
5
6
7
8
9
10

Program - 21

```
/* Program to Matrix Input / Output using function*/
#include <stdio.h>
#include <conio.h>
#define MAX 5
/* Function declaration */
void inputmatrix(int [][MAX], int, int);
void outputmatrix(int [][MAX], int, int);
void main()
{
int a[MAX][MAX], r, c;
clrscr();
printf(" Enter the row of matrix = ");
        scanf("%d", &r);
printf(" Enter the column of matrix = ");
        scanf("%d", &c);
inputmatrix(a, r, c);   /*Function declaration */
outputmatrix(a, r, c);  printf(" The Matrix is given below:- ");
                        /*Function declaration */
getch();
}
/* Function definition of input matrix */
void inputmatrix(int a[][MAX], int r, int c)
{
int i, j;
    for(i=0; i<=r-1; i++)
    {
     for(j=0; j<=c-1; j++)
     {
     printf(" Enter [%d][%d] value = ", i, j);
        scanf("%d", &a[i][j]);
     }
    }
}
/* Function definition of output matrix */
```

```
void outputmatrix(int a[][MAX], int r, int c)
{
    int i, j;
        for(i=0; i<=r-1; i++)
        {
            for(j=0; j<=c-1; j++)
            {
                printf("  %d", a[i][j]);
            }
            printf("\n");
        }
}
```

<u>output-</u>

Enter the row of matrix = 2
Enter the column of matrix = 2
Enter [0] [0] value = 1
Enter [0] [1] value = 2
Enter [1] [0] value = 3
Enter [1] [1] value = 4
The Matrix is given below :->

```
1   2
3   4
```

Program - 22

```
/* Program to perform the sum of all elements
        of matrix */
#include <stdio.h>
#include <conio.h>
void inputmatrix(int m[][4], int row, int col);
void outputmatrix(int m[][4], int row, int col);
void sumall(int m[][4], int row, int col);
void main()
{
  int m[4][4], r, c;
  clrscr();
  printf(" Enter no. of rows of matrix=");
    scanf("%d", &r);
  printf(" Enter no. of columns of matrix=");
    scanf("%d", &c);
  inputmatrix(m, r, c);
  outputmatrix(m, r, c);
  sumall(m, r, c);
  getch();
}
void inputmatrix(int m[][4], int row, int col)
{
  int i, j;
   for(i=0; i<=row-1; i++)
   {
      for(j=0; j<=col-1; j++)
      {
      printf(" Enter [%d][%d] th values = ", i, j);
      scanf("%d", &m[i][j]);
      }
   }
void outputmatrix(int m[][4], int row, int col)
{
  int i, j;
```

printf(" The matrix is given below:->\n");

```
    for(i=0; i<=row-1; i++)
    {
    for(j=0; j<=col-1; j++)
    {
      printf(" %d", m[i][j]);
    }
    printf("\n");
    }
}
void sumall(int m[ ][4], int row, int col)
{
  int sum=0, i, j;
    for(i=0; i<=row-1; i++)
    {
      for(j=0; j<=col-1; j++)
      {
        sum=sum + m[i][j];
      }
    }
  printf("The sum of all elements of matrix = %d", sum);
}
```

output-

Enter no. of row of matrix = 2
Enter no. of column of matrix = 2
Enter [0][0] th value = 1
Enter [0][1] th value = 2
Enter [1][0] th value = 3
Enter [1][1] th value = 4
The matrix is given below :>

1	2
3	4

The sum of all elements of matrix = 10

Program - 23

```
/* Program to perform the sum of all
   elements of matrix diagonal */
#include <stdio.h>
#include <conio.h>
void inputmatrix(int m[][4], int row, int col);
void outputmatrix(int m[][4], int row, int col);
void sumdiagonal(int m[][4], int row, int col);
void main()
{
int m[4][4], r, c;
clrscr();
printf(" Enter no. of rows of matrix = ");
scanf("%d", &r);
printf(" Enter no. of columns of matrix = ");
scanf("%d", &c);
inputmatrix(m, r, c);
printf(" The matrix is given below :-> ");
outputmatrix(m, r, c);
sumdiagonal(m, r, c);
getche();
}

void inputmatrix(int m[][4], int row, int col)
{
int i, j;
for(i=0; i<=row-1; i++)
{
for(j=0; j<=col-1; j++)
{
printf(" Enter [%d][%d] th values = ", i, j);
scanf("%d", &m[i][j]);
}
}
}
```

```
void outputmatrix(int m[][4], int row, int col)
{
int i,j;
  for(i=0; i<=row-1; i++)
  {
  for(j=0; j<=col-1; j++)
  {
    printf(" %d", m[i][j]);
  }
  printf("\n");
  }
void sumdiagonal(int m[][4], int row, int col)
{
int sum=0, i,j;
  for(i=0; i<=col-1; i++)
  {
    for(j=0; j<=row-1; j++)
    {
      if (i==j)
      {
        sum=sum+m[i][j];
      }
    }
  }
  printf("The sum of all elements of main diagonal =%d
                                    \n", sum);
}
```

Output-

Enter no. of rows of matrix = 3
Enter no. of columns of matrix = 3
Enter [0] [0] th value = 1
Enter [0] [1] th value = 2
Enter [0] [2] th value = 3
Enter [1] [0] th value = 4
Enter [1] [1] th value = 5
Enter [1] [2] th value = 6
Enter [2] [0] th value = 7
Enter [2] [1] th value = 8
Enter [2] [2] th value = 9
The Matrix is given below :->

1	2	3
4	5	6
7	8	9

The sum of all elements of main diagonal = 15

Program - 24

```
/* Program to perform sum of all elements of
   matrix rowwise */
#include <stdio.h>
#include <conio.h>
void inputmatrix(int m[][4], int row, int col);
void outputmatrix(int m[][4], int row, int col);
void sumrow (int m[], int row, int col);
void main()
{
int m[4][4], r, c;
clrscr();
printf(" Enter no. of row of matrix = ");
scanf("%d", &r);
printf(" Enter no. of column of matrix = ");
scanf("%d", &c);
inputmatrix(m, r, c);
printf(" The matrix is given below :->\n");
outputmatrix(m, r, c);
sumrow(m, r, c);
getch();
}
void inputmatrix(int m[][4], int row, int col)
{
int i, j;
for(i=0; i<=row-1; i++)
{
for(j=0; j<=col-1; j++)
{
printf(" Enter [%d][%d] th values = ", i, j);
scanf("%d", &m[i][j]);
}
}
}
```

```
void outputmatrix(int m[][4], int row, int col)
{
 int i,j;
 for(i=0; i<=row-1; i++)
 {
   for(j=0; j<=col-1; j++)
   {
     printf("  %d", m[i][j]);
   }
     printf("\n");
 }
}
void sumrow(int m[][4], int row, int col)
 {
 int sum,i, j;
 for(i=0; i<=row-1; i++)
 { sum=0;
   for(j=0; j<=col-1; j++)
   {
     sum = sum+m[i][j];
   }
 printf("\n The sum of all elements %dth row=%d
                                  \n", i,sum);
}
```

Output-

Enter no. of row of matrix = 3
Enter no. of column of matrix = 3
Enter [0] [0] th values = 1
Enter [0] [1] th values = 2
Enter [0] [2] th values = 3
Enter [1] [0] th values = 4
Enter [1] [1] th values = 5
Enter [1] [2] th values = 6
Enter [2] [0] th values = 7
Enter [2] [1] th values = 8
Enter [2] [2] th values = 9
The matrix is given below :->

1	2	3
4	5	6
7	8	9

The sum of all elements 1th row = 6
The sum of all elements 2th row = 15
The sum of all elements 3th row = 24

Program-25

```
/* Program to perform the sum of all elements
   of matrix colwise */
#include <stdio.h>
# include <conio.h>
void inputmatrix(int m[][4], int row, int col);
void outputmatrix(int m[][4], int row, int col);
void sumcol(int m[][4], int row, int col);
void main()
{
int m[4][4], r, c;
clrscr();
printf(" Enter no. of row of matrix =");
  scanf(" %d", &r);
printf(" Enter no. of col of matrix =");
  scanf(" %d", &c);
inputmatrix(m, r, c);
printf(" The matrix will be given below :->\n");
outputmatrix(m, r, c);
sumcol(m, r, c);
getch();
}
void inputmatrix(int m[][4], int row, int col)
{
int i, j;
for(i=0; i <= row-1; i++)
{
  for(j=0; j<= col-1; j++)
  {
  printf(" Enter [%d][%d]th values =", i, j);
   scanf(" %d", &m[i][j]);
  }
}
}
```

```
void outputmatrix( int m[][4], int row, int col)
{
  int i,j;
 for(i=0; i<= row-1; i++)
 {
    for(j=0; j<= col-1; j++)
    {
      printf(" %d", m[i][j]);
    }
      printf("\n");
  }
}
void sumcol( int m[][4], int row, int col)
{
 int sum, i, j;
 for(i=0; i<= col-1; i++)
 {
  sum=0;
    for(j=0; j<= row-1; j++)
    {
      sum = sum+ m[j][i];
    }
printf("\n The sum of all elements %d th col %d\n"
                                    , j, sum);
}
```

Output:-

Enter no of row of matrix = 3
Enter no of column of matrix = 3
Enter [0][0] th value = 1
Enter [0][1] th value = 2
Enter [0][2] th value = 3
Enter [1][0] th value = 4
Enter [1][1] th value = 5
Enter [1][2] th value = 6
Enter [2][0] th value = 7
Enter [2][1] th value = 8
Enter [2][2] th value = 9
The matrix is given below :->

1	2	3
4	5	6
7	8	9

The sum of all elements 1th col = 12
The sum of all elements 2th col = 15
The sum of all elements 3th col = 18

Program - 26

```
/* Program to transpose a matrix colwise */
#include <stdio.h>
#include<conio.h>
void inputmatrix(int m[][4], int row, int col);
void outputmatrix(int m[][4], int row, int col);
void transpose(int m[][4], int row, int col);
void main()
{
int m[4][4], r,c;
clrscr();
printf(" Enter no. of rows of matrix=");
scanf(" %d", &r);
printf("Enter no. of cols of matrix =");
  scanf(" %d", &c);
inputmatrix(m,r,c);
printf(" The matrix is given below :->\n");
outputmatrix(m,r,c);
transpose(m,r,c);
printf(" \n After transposition matrix is \n");
outputmatrix(m,r,c);
getch();
}
void inputmatrix(int m[][4], int row, int col)
{
int i,j;
for(i=0; i<=row-1; i++)
{
  for(j=0; j<=col-1; j++)
  {
    printf(" Enter [%d][%d] th values =", i,j);
      scanf(" %d", &m[i][j])
  }
}
}
```

```
void outputmatrix(int m[][4], int row, int col)
{
int i,j;
for(i=0;i<=row-1;i++)
{
    for(j=0;j<=col-1;j++)
    {
        printf("%d", m[i][j]);
    }
    printf("\n");
}
}
void transpose(int m[][4], int row, int col)
{
int t, temp, i,j;
if(row<col)
  t=col;
else
  t=row;
for(i=0;i<=t-1;i++)
  {
   for(j=0;j<t;j++)
   {
     temp = m[i][j];
     m[i][j]= m[j][i];
     m[j][i] = temp;
   }
  }
}
```

Output-

Enter no of row of matrix = 3
Enter no of col of matrix = 3
Enter [0][0] th value = 1
Enter [0][1] th value = 2
Enter [0][2] th value = 3
Enter [1][0] th value = 4
Enter [1][1] th value = 5
Enter [1][2] th value = 6
Enter [2][0] th value = 7
Enter [2][1] th value = 8
Enter [2][2] th value = 9

The matrix is given below :->

1	2	3
4	5	6
7	8	9

After transposition matrix is

1	4	7
2	5	8
3	6	9

Program-27

```
/* Program to store and restore sparse matrix in
      sparse array */
#include <stdio.h>
#include <conio.h>
typedef struct sparse
{
  int row;
  int col;
  int ele;
}sparse;
void storesparse (sparse s[], int row, int col);
void restoresparse (sparse s[]);
void displaysparse (sparse s[]);
void main()
{
  sparse s[5];
  int r,c, ele;
  clrscr();
  printf("Enter row and col no. of a matrix=");
    scanf("%d", &r);
    scanf("%d", &c);
  storesparse(s, r, c)
  restoresparse(s);
  displaysparse(s);
  getch();
}
void storesparse(sparse s[], int row, int col)
{
  int i,j, ele, k=1;
  s[0].row = row ; s[0].col = col;
  for(i=0; i<=row-1; i++)
  {
      for(j=0; j<=col-1 ; j++)
```

```
{
printf(" Enter [%d][%d]th value of sparse matrix=", i, j);
  scanf("%d", &ele);
  if (ele != 0)
  {
    s[k].row = i;
    s[k].col = j;
    s[k].ele = ele;
    k++;
  }
  }
s[0].ele = k-1;
}
}
void restoresparse(sparse s[])
{
int i, j, k=1, zero=0;
for(i=0; i<=s[0].row-1; i++)
{
    for(j=0; j<=s[0].col-1; j++)
    {
      if((i==s[k].row) && (j==s[k].col))
      {
       printf(" %d ", s[k].ele);
       k++;
      }
      else
      {
       printf(" %d ", zero);
      }
    }
  printf("\n");
  }
}
```

```
void displaysparse(sparse s[])
{
int i;
printf("\n Row    Col    Nonzero element \n");
  for(i=0; i<= s[0].ele ;i++)
  {
  printf("%d  %d  %d ", s[i].row, s[i].col, s[i].ele);
    printf("\n");
  }
}
```

Output:

Enter row and col no. of matrix = 3 3
Enter [0][0]th value of sparse matrix = 0
Enter [0][1]th value of sparse matrix = 0
Enter [0][2]th value of sparse matrix = 3
Enter [1][0]th value of sparse matrix = 0
Enter [1][1]th value of sparse matrix = 2
Enter [1][2]th value of sparse matrix = 0
Enter [2][0] th value of sparse matrix = 1
Enter [2][1] th value of sparse matrix = 0
Enter [2][2] th value of sparse matrix = 0

row	column	Nonzero element
3	3	3
0	2	3
1	1	2
2	0	1

0 0 3
0 3 0
2 0 0

Program - 28

```
/* Diagonal sparse matrix (only input non
        zero element) */
#include <stdio.h>
#include<conio.h>
void storesparse(int [],int [])
void restoresparse(int [],int [])
void main()
{
 int s[5];
 int r,c;
 clrscr();
    printf(" Enter the row of the matrix= ");
      scanf("%d", &r);
    printf(" Enter the column of the matrix
      scanf("%d", &c);
    if(r!=c)
    {
   printf(" This is not square matrix ");
    exit();
    getch();
    }
storesparse(s,r,c);
restorespare(s,r,c);
getch();
}
void storesparse(int s[], int r,int c)
 {
  int i, k=1;
   s[0]=r;
    for(i=0;i<=r-1;i++)
    {
  printf(" Input non zero [%d][%d] element of
            diagonal sparse matrix =", i,
```

```
    scanf(" %d", &s[k]);
    k++;
  }
}
void restoresparse (int s[])
{
  int i, j, k=1, z=0;
  for(i=0; i<=s[0]-1; i++)
  {
   for(j=0; j<=s[0]-1; j++)
   {
     if(i==j)
     {
       printf(" %d", s[k]);
       k++;
     }
     else
     {
       printf(" %d", z);
     }
   }
   printf("\n");
  }
}
```

Output -

Enter the row of the matrix = 3
Enter the col of the matrix = 3
Input non zero [0][0] element of diagonal sparse matrix = 1

Input non zero [1][1] element of diagonal sparse matrix = 2

Input non zero [2][2] element of diagonal sparse matrix = 3

1	0	0
0	2	0
0	0	3

Program- 29

```
/* Diagonal sparse matrix */
#include <stdio.h>
#include< conio.h>
void storesparse( int [], int, int );
void restoresparse( int [] );
void main( )
{
 int s[5];
 int r,c;
 clrscr( );
    printf(" Enter the row of the matrix =");
     scanf("%d" ,&r);
    printf(" Enter the col of the matrix=");
     scanf("%d", &c);
    if( r!=c )
    {
     printf(" This is not square matrix");
      exit( );
      getch( );
    }
 storesparse( s, r, c );
 restoresparse (s, r, c);
 getch( );
}

void storesparse (int s[], int r, int c)
{
 int i, k=1, j, el;
  s[0] = r;
    for (i=0; i<= r-1; i++)
    {
     for (j=0; j<= c-1; j++)
      {
    printf(" Input [%d][%d] element of dagonal
```

```
            sparse matrix = ", i, j);
        scanf("%d", &ele);
        if(ele != 0)
        {
         s[k] = ele;
         k++;
        }
      }
      }
  }
void restoresparse(int s[])
{
 int i,j, k=1, s=0;
   for(i=0; i<=s[0]-1; i++)
   {
   for(j=0; j<=s[0]-1; j++)
   {
     if(i==j)
     {
       printf(" %d ", s[k]);
     k++;
     }
     else
     {
       printf(" %d", z);
     }
   }
   printf("\n");
 }
}
```

Output-

Enter the row of the matrix = 3
Enter the col of the matrix = 3
Input [0][0] element of diagonal sparse matrix = 1
Input [0][1] element of diagonal sparse matrix = 0
Input [0][2] element of diagonal sparse matrix = 0
Input [1][0] element of diagonal sparse matrix = 0
Input [1][1] element of diagonal sparse matrix [redacted]
Input [1][2] element of diagonal sparse matrix = 0
Input [2][0] element of diagonal sparse matrix = 0
Input [2][1] element of diagonal sparse matrix = 0
Input [2][2] element of diagonal sparse matrix = 3

1	0	0
0	2	0
0	0	3

Program-30

```
/* Tridiagonal sparse matrix */
#include <stdio.h>
#include <conio.h>
void storesparse(int [],int,int);
void restoresparse(int [] );
void main()
{
  int s[5];
  int r,c;
  clrscr();
  printf(" Enter the row of the matrix=");
  scanf("%d",&r);
  printf(" Enter the col of the matrix=");
  scanf("%d",&c);
  if(r!=c)
  {
  printf(" This is not square matrix=");
  exit();
  getch();
  }
storesparse(s,r,c);
restoresparse(s);
getch();
}
void storesparse(int s[],int r,int c)
{
int i,k=1,j;
s[0]=r;
  for(i=0; i<=r-1;i++)
  {
  for(j=0; j<=c-1;j++)
  {
  if((i==j)||(i==j+1)||(i==j-1))
  {
```

```
printf("Input nonzero [%d][%d] element of tridiagonal
                                sparse matrix = ");
    scanf("%d", &s[k]);
    k++;
  }
 }
 }
}
void restoresparse(int s[])
{
 int i,j,k=1,z=0;
  for(i=0;i<=s[0]-1;i++)
  {
   for(j=0;j<=s[0]-1;j++)
   {
   if((i==j)||(i==j-1)||(i==j+1))
   {
    printf(" %d", s[k]);
    k++;
   }
   else
   {
    printf(" %d", z);
   }
  }
  printf("\n");
 }
}
```

Output-

Enter the row of the matrix = 4
Enter the column of the matrix = 4
Input nonzero [0][0] element of tridiagonal sparse matrix = 1
Input nonzero [0][1] element of tridiagonal sparse matrix = 2
Input nonzero [1][0] element of tridiagonal sparse matrix = 3
Input nonzero [1][1] element of tridiagonal sparse matrix = 4
Input nonzero [1][2] element of tridiagonal sparse matrix = 5
Input nonzero [2][1] element of tridiagonal sparse matrix = 6
Input nonzero [2][2] element of tridiagonal sparse matrix = 7
Input nonzero [2][3] element of tridiagonal sparse matrix = 8
Input nonzero [3][2] element of tridiagonal sparse matrix = 9
Input nonzero [3][3] element of tridiagonal sparse matrix = 10

1	2	0	0
3	4	5	0
0	6	7	8
0	6	9	10

Program - 31

```
/* Upper triangular sparse matrix */
#include<stdio.h>
#include<conio.h>
void storesparse(int [],int ,int);
void restoresparse(int []);
void main( )
{
 int s[10];
 int r,c;
 clrscr();
 printf(" Enter the row of the matrix= ");
    scanf("%d", &r);
 printf(" Enter the col. of the matrix=");
    scanf("%d", &c);
storesparse(s, r, c);
restoresparse(s);
getch();
}
void storespare(int s[],int r,int c)
{
 int i, k=1, j;
 s[0] = r;
    for( i=0; i <= r-1; i++)
    {
      for(j=0; j<=c-1; j++)
      {
        if( i<= j)
        {
          printf(" Input nonzero [%d][%d] element
                    of upper triangular sparse matrix=");
          scanf("%d", &s[k]);
        k++;
      }
```

```
        }
        }
    }
void restoresparse(int s[])
{
    int i, j, k=1, z=0;
    for(i=0; i<=s[0]-1; i++)
    {
        for(j=0; j<=s[0]-1; j++)
        {
            if (i<=j)
            {
                printf("  %d", s[k]);
                k++;
            }
            else
            {
                printf("  %d", z);
            }
        }
        printf("\n");
    }
}
```

output-

Enter the row of the matrix = 3
Enter the column of the matrix = 3
Input nonzero [0][0] element of upper triangular sparse matrix = 1
Input nonzero [0][1] element of upper triangular sparse matrix 2
Input nonzero [0][2] element of upper triangular sparse matrix = 3
Input nonzero [1][1] element of upper triangular sparse matrix = 4
Input nonzero [1][2] element of upper triangular sparse matrix = 5
Input nonzero [2][2] element of upper triangular sparse matrix = 6

1	2	3
0	4	5
0	0	6

Program - 32

```
/* Lower triangular sparse matrix */
#include< stdio.h>
#include< conio.h>
void storesparse(int [],int,int);
void restoresparse(int []);
void main()
{
 int s[10], r,c;
 clrscr();
 printf(" Enter the row of the matrix = ");
      scanf("%d", &r);
 printf(" Enter the column of the matrix = ");
      scanf("%d", &c);
 storesparse(s,r,c);
 restoresparse(s);
 getch();
}

void storesparse(int s[], int r,int c)
{
 int i,k = 1,j;
 s[0]=r;
   for(i= 0;i<=r-1; i++)
   {
     for(j=0;j<=c-1; j++)
     {
       if(i >=j)
       {
 printf(" Input nonzero [%d][%d] element of lower
                triangular sparse matrix = ",i,j);
      scanf("%d", &s[k]);
      k++;
      }
    }
  }
}
```

```
void restoresparse(int s[])
{
  int i, j, k=1, z=0;
  for(i=0; i<=s[0]-1; i++)
  {
    for(j=0; j<=s[0]-1; j++)
    {
      if(i>=j)
      {
        printf("  %d", s[k]);
        k++;
      }
    }
    else
    {
      printf("  %d", z);
    }
    }
    printf("\n");
  }
}
```

output-

Enter the row of the matrix = 3
Enter the column of the matrix = 3
Input nonzero [0][0] element of lower triangular sparse matrix: 1
Input nonzero [1][0] element of lower triangular sparse matrix = 2
Input nonzero [1][1] element of lower triangular sparse matrix = 3
Input nonzero [2][0] element of lower triangular sparse matrix = 4
Input nonzero [2][1] element of lower triangular sparse matrix = 5
Input nonzero [2][2] element of lower triangular sparse matrix = 6

1	0	0
2	3	0
4	5	6

sparse matrix = 1
matrix = 2
matrix = 3
matrix = 4
matrix = 5
= 6

Program-33

```
/* Program to demonstrate structure and pointer
   and use of dot operator */
#include <stdio.h>
#include <conio.h>
typedef struct student
{
  int rollno;
  int age;
} student;
void main()
{
  Student s1, *s2;
  clrscr();
   printf("------- using variable -------");
   printf("\n Enter the rollno. of student = ");
    scanf("%d", &s1.rollno);
   printf(" Enter the age of student = ");
    scanf("%d", &s1.age);
   printf" Roll no. = %d  Age = %d\n", s1.rollno, s1.age);
   printf("------- using pointer -------");
   printf("\n Enter the rollno. of student = ");
    scanf("%d", &(*s2).rollno);
   printf(" Enter the age of student = ");
    scanf("%d", &(*s2).age);
   printf" Roll no. = %d  Age = %d\n", (*s2).rollno,
                                       (*s2).age);
   printf("\n ------ using Arrow operator ------");
   printf("\n Enter the rollno. of student = ");
    scanf("%d", &s2->rollno);
   printf(" Enter the age of student = ");
    scanf("%d", &s2->age);
   printf" Roll no = %d  Age = %d\n", s2->rollno, s2->age);
  getch();
}
```

Output-

--------using variable------

Enter the roll no. of student = 23

Enter the age of student = 1

Roll no. = 23 Age = 1

--------using pointer------

Enter the roll no. of student = 23

Enter the age of student = 1

Roll no. = 23 Age = 1

--------using Arrow operator---

Enter the roll no. of student = 23

Enter the age of student = 1

Roll no. = 23 Age = 1

Program-34

```
/* Program to demonstrate input and output
   in linked list using function */
#include<stdio.h>
#include<conio.h>
typedef struct node {
 int info;
 struct node *link;
}node;
// function declaration
void createlist(node *);
void displaylist(node *);
void releaselist(node *);
void main()
{
  node *start = NULL;
  start = (node *)malloc (sizeof(node));
  clrscr();
  createlist(start);
  displaylist(start);
  releaselist(start);
  getch();
}
/* function definition for input linked list */
void createlist(node *current)
{
char ch;
    do{
    printf("Enter the value of node = ");
        scanf("%d", &current->info);
    printf(" Do you want more nodes y/n = ");
        fflush (stdin);
        scanf("%c", &ch);
    if(ch == 'y')
```

```
    {
    current->link = (node *) malloc(sizeof(node));
    current=current->link;
    }
    else
    {
      current->link = NULL;
    }
  }
  while(ch=='y');
}
/* function definition for output linked list */
void displaylist(node * current)
{
    while (current != NULL)
    {
      printf("\n %d", current->info);
      current = current->link;
    }
}
/* Function definition for release memory */
void release_list(node * current)
{
  node *temp;
      while(current != NULL)
      {
        temp= current;
        current = current->link;
        free(current);
      }
  free(temp);
}
```

output

Enter the value of node = 3
Do you want more nodes y/n = y
Enter the value of node = 5
Do you want more nodes y/n = y
Enter the value of node = 7
Do you want more nodes y/n = n

3
5
7

Program-35

```
/* Program to search an item in linked list
   and return locations */
#include <stdio.h>
#include <conio.h>
typedef struct node {
int info;
struct node *link;
}node;
void createlist(node * current);
void displaylist(node *current);
node *search(node * current, int item);
void releaselist(node * current);
void main()
{
node *start= NULL, *loc= NULL;
int item;
clrscr();
start= (node *) malloc (sizeof(node));
createlist(start);
displaylist(start);
printf("\n Enter the item to search =");
scanf("%d", &item);
loc= search(start, item);
if(loc != NULL)
printf("\n item is found at address= %u", loc);
else
printf("\n item is not found");
releaselist(start);
getch();
}
/* Createlist */
void createlist(node *current)
{
```

```
char ch;
do
{
  printf("\n Enter info part of node = ");
    scanf("%d", &current->info);
  printf("\n Do you want another node y/n = ");
    fflush(stdin);
    scanf("%c", &ch);
  if( ch == 'y')
  {
  current->link = (node *)malloc (sizeof(node));
  current = current->link;
  }
  else
  {
      current->link = NULL;
  }
}
while( ch == 'y');
}
/* Display list */
void displaylist( node *current)
{
  while (current != NULL)
  {
    printf("\n %d", current->info);
    current = current->link;
  }
}
/* Searching */
node *search (node * current, int item)
{
  while(current != NULL)
```

```
        {
            if(current->info ==item)
            {
                return(current);
            }
            current = current -> link;
        }
        return(NULL);
    }
/* release list */
void  releaselist(node * current)
{
    node *temp;
        while(current != NULL)
        {
            temp = current;
            current = current->link;
            free(temp);
        }
}
```

output —

Enter the value of node = 3
Do you want another node = y
Enter the value of node = 4
Do you want another node = y
Enter the value of node = 5
Do you want another node = y
Enter the value of node = 6
Do you want another node = y
Enter the value of node = 7
Do you want another node = n

3
4
5
6
7

Enter the item to search = 5

Item is found at address = 601

Program-36

```
/* Program to search an item in linked list
   and print location */
#include<stdio.h>
#include<conio.h>
typedef struct node{
 int info;
 struct node *link;
 }node *;
/* Function declaration */
void createlist(node *current);
void displaylist(node *current);
void search(node *current);
void releaselist(node *current);
void main()
{
node *start = NULL, *loc = NULL;
int item;
clrscr();
start = (node *) malloc(sizeof(node));
createlist(start);
displaylist(start);
printf("\n Enter the item to search=");
  scanf("%d", &item);
search(start,item);
releaselist(start);
getch();
}

/* createlist */
void createlist(node *current)
{
char ch;
do
{
```

```
printf("\n Enter info part of node = ");
 scanf("%d", &current->info);
printf("\n Do you want another node y/n = ");
 fflush(stdin);
scanf("%c", &ch);
    if(ch == 'y')
    {
    current->link = (node *)malloc (sizeof (node));
    current = current->link;
    }
    else
    {
    current->link = NULL;
    }
  } while (ch == 'y');
}
/* displaylist */
void  displaylist (node *current)
{
 printf("\n The list is given below \n");
   while (current != NULL)
   {
   printf("  %d", current->info);
    current = current->link;
   }
}
/* Searching */
void  search (node *current, int item)
{
    while (current != NULL)
    {
      if (current->info == item)
      {
```

```
printf("\n Search is successful, item is found
                    at %u", current);

    getch();
    return;
  }
    current = current->link;
}
printf("\n Search is unsuccessful, item is not
                    found \n");

}
/* Release list */
void    releaselist(node *current)
{
  node  *temp;
      while(current != NULL)
      {
          temp= current;
          current = current -> link;
          free(temp);
      }
}
```

output -

Enter the value of node = 1
Do you want another node y/n = y
Enter the value of node = 2
Do you want another node y/n = y
Enter the value of node = 3
Do you want another node y/n = y
Enter the value of node = 4
Do you want another node y/n = n

The list is given below
1
2
3
4

Enter the item to search = 3

Search is successful, item is found at 601

Program-37

```
/* program to insert a node in begining of
   linked list using return value concept */
#include<stdio.h>
#include<conio.h>
typedef struct node{
int info;
struct node *link;
}node;
/* Function declaration */
void createlist(node *current);
void displaylist(node *current);
node *insertbegin(node** start, int item);
void releaselist(node *current);
void main()
{
node *start=NULL;
int item;
start=(node*) malloc (sizeof(node));
clrscr();
createlist(start);
displaylist(start);
printf("\n Enter the item to insert=");
scanf("%d", &item);
start=insertbegin(start, item);
displaylist(start);
releaselist(start);
getch();
}
void createlist(node *current)
{
char ch;
do
{
```

```
printf(" Enter value of node = ");
  scanf(" %d ", &current->info);
printf(" Do you want more nodes y/n=");
   fflush (stdin);
scanf(" %c", &ch);
  if(ch==='y')
  {
current->link = (node *) malloc (sizeof(node));
  current = current->link;
  }
  else
  current -> link = NULL;
}
while (ch==='y');
}
void   displaylist(node *current)
{
    while(current != NULL)
    {
     printf(" \n %d", current->info);
       current = current -> link;
    }
}
node *Insertbegin(node *start, int item)
{
 node *newnode;
newnode = (node *) malloc (sizeof(node));
    if(newnode ==NULL)
    {
       printf(" Overflow");
         getch();
           exit(0);
    }
```

```
    else
    {
    newnode -> info = item;
    newnode -> link = start;
    start = newnode;
    }
return(start);
}
void releasestart(node *current)
{
    node *temp;
    while (current != NULL)
    {
        temp = current;
        current = current->link;
        free(current);
    }
    free(temp);
}
```

output -

Enter value of node = 1

Do you want ~~another item~~ more nodes y/n = y

Enter value of node = 2

Do you want more nodes y/n = y

Enter value of node = 3

Do you want more nodes y/n = y

Enter value of node = 4

Do you want more nodes y/n = n

1
2
3
4

Enter the item to insert = 5

5
1
2
3
4

Program-30

```
/* program to insert a node in begining of
   linked list without return value */
#include<stdio.h>
#include<conio.h>
typedef struct node{
  int info;
  struct node *link;
 }node;
/* Function declaration */
void  createlist(node *current);
void  displaylist(node *current);
void  insertbegin(node **start, int item);
void  releaselist(node *current);
void  main()
{
  node  *start=NULL;
  int  item;
  start=(node *) malloc(sizeof(node));
  clrscr();
  createlist(start);
  displaylist(start);
   printf("\n Enter the item to insert= ");
      scanf("%d", &item);
  insertbegin(&start, item);
  display(start);
  releaselist(start);  getch();
}
/* Createlist */
void  creatlist(node *current)
{
  char  ch;
  do
  {
```

```
printf(" Enter value of node = ");
    scanf("%d", &current->info);
printf(" Do you want more node y/n :");
    fflush(stdin);
    scanf("%c", &ch);
    if (ch==='y')
    {
    current->link = (node*)malloc(sizeof(node));
      current = current->link;
    }
    else
    current->link = NULL;
}
while (ch=='y');
}
/* Displaylist */
void displaylist(node *current)
{
    while(current != NULL)
    {
    printf("\n %d", current->info);
      current = current->link;
    }
}
/* Insertion at begin without return value */
void insertbegin(node **start, int item)
{
  node *newnode;
  newnode=(node*) malloc(sizeof(node));
    if(newnode==NULL)
    {
        printf(" Overflow ");
        getch();
          exit(0);
    }
```

```
        else
        {
            newnode -> info = item;
            newnode -> link = *start;
            *start = newnode;
        }
    }
/* Release list */
void    releaselist(node *current)
{
    node *temp;
    while(current != NULL)
    {
        temp = current;
        current = current -> link;
        free(current);
    }
    free(temp);
}
```

Output

Enter value of node = 1
Do you want more nodes y/n = y
Enter value of node = 2
Do you want more nodes y/n = y
Enter value of node = 3
Do you want more nodes y/n = y
Enter value of node = 4
Do you want more nodes y/n = y
Enter value of node = 5
Do you want more nodes y/n = n

1
2
3
4
5

Enter the item to insert = 6

6
1
2
3
4
5

Program 39

```
/* program to insert a node in middle of linked
   list after an element */
#include <stdio.h>
#include <conio.h>
typedef struct node
{
   int info;
   struct node *link;
} node;
void createlist (node * current);        /*function declaration*/
void displaylist (node * current);       node *search(node
void insertmiddle(node *loc, int item);  *loc, int after);
void releselist (node *current);
void main()
{
   int item, after;
   node *start = NULL, *loc = NULL;
   clrscr();
   start = (node *) malloc (sizeof (node));
   createlist(start);
   displaylist(start);
   printf(" Enter the item to insert = ");
     scanf(" %d ", &item);
   printf(" Enter the item after wic you want to insert = ");
     scanf(" %d ", &after);
   loc = search (start, after);
   insertmiddle (loc, item);
   printf("\n The linked list after insertion \n ");
   displaylist(start);
   releselist(start);
   getch();
}
```

```
/* Creating list */
void createlist(node *current)
{
char ch;
do
{
printf("\n Enter info part of node = ");
  scanf("%d ", &current->info);
printf("\n Do you want another nodes y/n = ");
      fflush(stdin);
 scanf(" %c", &ch);
 if(ch == 'y')
 {
   current->link = (node*)malloc(sizeof(node));
     current = current->link;
 }
 else
 {
  current->link = NULL;
 }
}while(ch == 'y');
}   /* Display list */
void displaylist(node *current)
{
 printf("\n The list is given below \n");
   while(current != NULL)
   {
    printf(" %d", current->info);
    current = current->link;
   }
/* Insertion at middle */
void insertmiddle(node *loc, int item)
{
```

```
node *newnode;
newnode = (node *) malloc(sizeof(node));
    if(newnode == NULL)
    {
      printf(" overflow ");
        getch();
        exit(0);
      }
    else
      {
      newnode -> info = item;
      newnode -> link = loc -> link;
      loc -> link = newnode;
      }
}
/* Searching element and return location */
node *search(node *current, int after)
{
  while( current != NULL)
    {
      if (current -> info == after)
        return (current);
  current = current -> link;
    }
}
/* Release list */
void releaselist(node *current)
{
  node *temp;
    while( current != NULL)
    {
      temp = current; current = current -> link; free(temp
    }
}
```

Output-

Enter info part of node = 1
Do you want another nodes y/n = y
Enter info part of node = 2
Do you want another nodes y/n = y
Enter info part of node = 3
Do you want another nodes y/n = y
Enter info part of node = 4
Do you want another nodes y/n = n
The list is given below

1
2
3
4

Enter the item to insert = 7
Enter the item after w/c you want to insert = 3
The linked list after insertion

The list is given below

1
2
3
7
4

Program-40

```
/* Program to insert a node at last of
      linked list */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct node
{
    int info;
    struct node *link;
}node;
/* Function declaration */
void createlist(node *);
void displaylist(node *);
void insertmiddle(node *, int);
void releaselist (node *);
void main()
{
    node *start = NULL;
    int item;
    start = (node *) malloc (sizeof (node));
    clrscr();
    createlist (start);
    displaylist(start);
  printf("\n Enter the item to insert");
    scanf("%d", &item);
  insertlast (start, item);
  displaylist (start);
  releaselist(start);
  getch();
}
/* Createlist */
void  createlist (node * current)
{
```

```
char ch;
do
{
printf(" Enter value of node = ");
  scanf("%d ", &current->info);
printf(" Do you want more nodes y/n = ");
      fflush(stdin);
  scanf(" %c ", &ch);
  if(ch == 'y')
  {
  current->link = (node *) malloc (sizeof(node));
   current = current->link;
  }
  else
  {
     current -> link = NULL;
  }
}
while( ch == 'y');
}
/* Display list */
void display list(node *current)
{
    while (current != NULL)
    {
       printf(" \n %d ", current->info);
       current = current -> link;
    }
}
/* Insertion at last */
void insertlast (node *current, int item)
{

node *newnode;
```

```
newnode = (node *) malloc (sizeof( node ));
        if(newnode = = NULL)
        {
        printf("Overflow");
          getch();
          exit(0);
        }
        else
        {
        newnode -> info = item;
        while( current->link != NULL)
        {
            current = current -> link;
        }
        newnode -> link = NULL;
        current -> link = newnode;
        }
return;
}
/* Release list */
void release list (node * current)
{
  node *temp;
    while ( current != NULL)
    {
        temp = current;
        current = current -> link;
        free ( current );
    }
  free(temp);
}
```

Output →

Enter value of node = 1

Do you want more nodes y/n = y

Enter value of node = 2

Do you want more nodes y/n = y

Enter value of node = 3

Do you want more nodes y/n = y

Enter value of node = 4

Do you want more nodes y/n = n

1

2

3

4

Enter the item to insert = 5

1

2

3

4

5

Program-41

```
/* Program to delete a node from begining
   of linked list using return value */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct node
{
    int info;
    struct node *link;
} node;
/* Function declaration */
void createlist (node *current);
void displaylist (node * current);
node * deletebegin (node * current);
void release list( node * current);
void main( )
{
  node *start = NULL;
  int item;
  start = (node *) malloc (sizeof (node));
  clrscr( );
  create list (start);
  display list (start);
  start = deletebegin (start);
  printf("\n After deletion list is ");
  displaylist (start);
  release list (start);
  getch( );
}
/* Create list */
void create list (node * current)
{
  char ch;
```

```
do
{
printf(" Enter value of node= ");
scanf("%d", &current->info);
printf("Do you want another node y/n= ");
fflush(stdin);
scanf("%c", &ch);
if(ch=='y')
{
current->link= (node *)malloc(sizeof(node));
current= current->link;
}
else
{
current->link=NULL;
}
}
while(ch=='y');
}
/* Display list */
void displaylist(node *current)
{
while(current!=NULL)
{
printf("\n %d ", current->info);
current = current->link;
}
}
/* Deletion at begin with return value */
node *deletebegin(node *start)
{
node *temp;
if(start==NULL)
{
```

```
        printf(" Underflow");
          getch();
          exit(0);
        }
        else
        {
          temp = start;
          start = start->link;
          free(temp);
        }
    return(start);
  }
/* Release list */
void releaselist(node * current)
{
  node *temp;
    while(current!=NULL)
    {
        temp = current;
        current = current->link;
          free(current);
    }
  free(temp);
}
```

output -

Enter value of node = 1
Do you want another node y/n = y
Enter value of node = 2
Do you want another node y/n = y
Enter value of node = 3
Do you want another node y/n = y
Enter value of node = 4
Do you want another node y/n = y
Enter value of node = 5
Do you want another node y/n = n

1
2
3
4
5

After Deletion list is

2
3
4
5

Program-42

```
/* Program to delete a node from beginning
   of linked list without return value */
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
typedef struct node
{
  int info;
  struct node *link;
}node;
/* Function declaration */
void createlist(node *);
void displaylist(node *);
void deletebegin(node **);
void releaselist(node *);
void main()
{
  node *start=NULL;
  int item;
  start=(node*)malloc(sizeof(node));
  clrscr();
  createlist(start);
  display(start);
  deletebegin(&start);
  printf("\n After deletion list is ");
  displaylist(start);
  releaselist(start);
  getch();
}
/* Createlist */
void createlist(node *current):
{
  char ch;
```

```
do
{
 printf(" Enter value of node = ");
   scanf("%d", & current->info);
 printf(" Do you want another node y/n = ");
    fflush(stdin);
    scanf("%c", &ch);
    if (ch == 'y')
    {
    current -> link = (node *) malloc (sizeof(node));
     current = current -> link;
     }
     else
     {
       current -> link = NULL;
     }
  }
  while (ch == 'y');
}
  /* Display list */
void displaylist(node * current)
{      while (current != NULL)
  { printf(" \n %d", current->info);
       current = current->link;
  }

}/* Deletion at begin without return value */
void deletebegin(node ** start)
{
   node *temp;
     if (start == NULL)
     {
```

```
        printf(" Underflow");
          getch();
          exit(0);
      }
      else
      {
          temp = *start;
        *start = *start -> link;
          Free(temp);
      }
  }
/* Release list */
void release list( node *current)
{
    node * temp;
    while( current != NULL)
    {
        temp = current;
        current = current -> link;
        free(current);
    }
  free(current);
}
```

output
Enter value of node = 1
Do you want another node y/n = y
Enter value of node = 2
Do you want another node y/n = y
Enter value of node = 3
Do you want another node y/n = y
Enter value of node = 4
Do you want another node y/n = y
Enter value of node = 5
Do you want another node y/n = n

1
2
3
4
5

After Deletion ~~the~~ list is

2
3
4
5

Program-43

```
/* Program to delete an element from the middle
   of linked list */
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
typedef struct node
{
    int info;
    struct node *link;
}node;
/*Function declaration */
void createlist (node *);
void displaylist (node *);
void deletemiddle (node *, int);
void release list(node *);
void main()
{
node *start = NULL;
int item;
start = (node *) malloc (sizeof (node));
clrscr();
create list (start);
display list (start);
printf("\nEnter the item to delete from middle");
  scanf("%d", &item);
deletemiddle(start, item);
printf("\n After deletion list is \n");
displaylist (start);
releaselist(start);
getch();
}
/* Createlist */
void createlist (node *current)
```

from the middle

middle");


```
{
 char ch;
 do
  {
   printf(" Enter value of node = ");
     scanf("%d", &current->info);
 printf(" Do you want another node y/n = ");
        fflush(stdin);
     scanf("%c", &ch);
     if ( ch == 'y' )
     {
     current -> link = (node *) malloc (sizeof (node));
      current = current -> link;
      }
      else
      {
      current -> link = NULL;
      }
   } while (ch == 'y');
}
  /* Display list */
void displaylist(node *current)
 {
     while(current != NULL)
     {
       printf("\n %d", current -> info);
       current = current -> link;
     }
}
 /* Deletion at middle */
void deletemiddle(node *current, int item)
{
  node * prev;
```

```
        if(current == NULL)
        {
            printf(" Underflow");
            getch();
            exit(0);
        }
        else
        {
            while(current != NULL)
            {
                if(current->info == item)
                {
                    prev->link = current->link;
                    free(current);
                }
                prev=current;
                current=current->link;
            }
        }
    return;
}
/* Release list */
void releaselist(node *current)
{
    node *temp;
    while(current != NULL)
    {
        temp = current;
        current = current->link;
        free(current);
        ~~delete node~~;
    }
    free(temp);
}
```

Output-

Enter value of node = 1
Do you want another node y/n = y
Enter value of node = 2
Do you want another node y/n = y
Enter value of node = 3
Do you want another node y/n = y
Enter value of node = 4
Do you want another node y/n = y
Enter value of node = 5
Do you want another node y/n = n

1
2
3
4
5

Enter the item to delete from middle = 3
After deletion the list is
1
2
4
5

Program-44

```
/* Program to delete an element from the last of
        linked list */
#include stdio.h>
#include conio.h>
#include< stdlib.h>
typedef struct node
  {
      int info;
      struct node *link;
  }node;
/* Function declaration */
void creatlist (node *current);
void displaylist (node *current);
void delete last( node * current);
void releaselist( node *current);
void main( )
  {
    node *start = NULL;
    int item;
    start = (node *) malloc (sizeof(node));
    clrscr( );
    creatlist (start);
    displaylist(start);
    deletelast(start);
  printf("\n After deletion list is \n");
    displaylist (start);
    releaselist(start);
  getch( );
}
/* Creatlist */
void creatlist( node * current)
{
   char ch;
```

```
do
{
printf(" Enter value of node = ");
  scanf(" %d ", &current->info);
printf(" Do you want another node y/n = ");
  fflush(stdin);
  scanf(" %c ", &ch);
  if(ch=='y')
  {
   current->link = (node*) malloc(sizeof(node));
   current = current->link;
  }
  else
  {
   current->link = NULL;
  }
 }
 while(ch=='y')
}
/* Display list */
void  displaylist(node *current)
{
   while (current != NULL)
   {
     printf(" \n %d ", current->info);
     current = current->link;
   }
}

/* Deletion at last */
void deletelast(node *current)
{
 node *prev;
      if(current == NULL)
```

```
        {
            printf(" underflow ");
            getch();
            exit(0);
        }
        else
        {
            while (current -> link != NULL)
            {
                prev = current;
                current = current -> link;
            }
            prev -> link = NULL;
            free(current);
        }
    return;
}
/* Release list */
void releaselist(node * current)
{
    node * temp;
    while( current != NULL)
    {
        temp = current;
        current = current -> link;
        free(current);
    }
    free(temp);
}
```

Output -

Enter value of node = 1
Do you want another node y/n = y
Enter value of node = 2
Do you want another node y/n = y
Enter value of node = 3
Do you want another node = y
Enter value of node = 4
Do you want another node = y
Enter value of node = 5
Do you want another node = n

1
2
3
4
5

After deletion list is
1
2
3
4

Program-45

```
/* Program to delete a node in middle of
   linked list by searching */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct node
{
    int info;
    struct node *link;
}node;
/*Function declaration */
void createlist(node *current);
void displaylist(node * current);
node ** search(node *start, int item);
void deletemiddle(node *current, node *loc);
void releaselist(node *current);
void main()
{
 int item;
 node *start = NULL, *loc = NULL;
 clrscr();
 start = (node*)malloc(sizeof(node));
 createlist(start);  displaylist(start);
printf("\n Enter the item to delete \n");
scanf("%d", &item);
loc = search(start, item);
deletemiddle(start, loc);
printf("\n The linked list after deletion \n");
 displaylist(start);
releaselist(start);
 getch();
}
```

```
/* Create list */
void createlist(node *current)
{
char ch;
do
{
printf("\n Enter into part of node=");
  scanf("%d", & current->info);
printf("Do you want another node y/n = ");
    fflush(stdin);
    scanf("%d", &ch);
    if( ch==('y')
  {
  current->link = (node *) malloc (sizeof(node));
    current = current->link;
  }
  else
   {
    current->link = NULL;
   }
 }
 while(ch=='y');
}
/* Display list */
void displaylist(node *current)
{
 printf("\n The list is given below\n");
 while(current != NULL)
  {
   printf(" %d", current->info);
   current = current->link;
  }
}
```

```
/* Deletion at middle */
void deletemiddle(node *current, node *loc)
{
    if(current == NULL)
    {
        printf("underflow");
        getch();
        exit(0);
    }
    else
    {
    while(current -> link != loc)
    {
      current = current -> link;
    }
    current -> link = loc -> link;
    free(loc);
    }
}
/* Searching an item */
node **search(node *current, int item)
{
  while(current != NULL)
  {
      if(current -> info == item)
      {
          return(current);
      }
    current = current -> link;
  }
}
```

```
/* Release list */

void releaselist(node * current)
{
  node * temp;
    while (current != NULL)
    {
      temp = current;
      current = current->link;
      free(temp);
    }
free(temp);
}
```

output -

Enter info part of node = 1
Do you want another node y/n = y
Enter info part of node = 2
Do you want another node y/n = y
Enter info part of node = 3
Do you want another node y/n = y
Enter info part of node = 4
Do you want another node y/n = y
Enter info part of node = 5
Do you want another node y/n = n

The list is given below

1
2
3
4
5

Enter the item to delete = 4

The linked list after deletion

The list is given below

1
2
3
5

Program - 46

```
/* Program to demonstrate input and
   output in circular linked list */
#include <stdio.h>
#include <conio.h>
typedef struct node
{
    int info;
    struct node *link;
} node;

/* Function declaration */
void createlist (node * start);
void displaylist (node *current);
void releaselist (node *current);

void main()
{
  node *start = NULL;
  start = (node *) malloc (sizeof(node));
  clrscr();
  createlist (start);
  displaylist (start);
  releaselist (start);
  getch();
}


/* Create list */
void createlist( node *start)
{
  node *current = start;
  char ch;
  do
  {
```

```
        printf(" Enter value of node = ");
          scanf(" %d ", & current->info);
        printf(" Do you want more node y/n =");
            fflush(stdin);
          scanf(" %c " &ch);
        if (ch == 'y')
        {
            current -> link = (node *)malloc (sizeof(node));
            current = current -> link;
        }
        else
        {
            current -> link = start;
        }
    } while (ch == 'y');
}

/* Display list */
void displaylist(node *current)
{
        while(current != NULL)
        {
            printf("\n %d", current->info);
            current = current -> link;
        }
}


/* Release list */
void releaselist(node *current)
{
    node *temp;
        while( current != NULL)
        {
```

```
        temp = current;
        current = current->link;
        free(current);
    }
    free(temp);
}
```

output-

```
Enter value of node = 1
Do you want more node y/n = y
Enter value of node = 2
Do you want more node y/n = y
Enter value of node = 3
Do you want more node y/n = n

1
2
3
```

Program -43

```
/* Program to create and display doubly/
   two way linked list using function.
#include <stdio.h>
#include <conio.h>
#include <stlib.h>
typedef struct node
{ struct node *prev;
    int info;
    struct node {} *link;
}node;

// function declaration */
node *create(node *current);
void display(node *current);
void displayrev(node *);
void releaselist(node *current);

void main()
{
  int item;
  node *start = NULL, *end = NULL;
  clrscr();
  start = (node *) malloc (sizeof(node));
   end = create(start);
printf("\n Forward order traversal \n");
   display(start);
printf("\n Reverse order traversal \n");
  ~~display(rev);~~ displayrev(end);
releaselist(start);
getch();

}
```

y doubly/
function.


```
/* create doubly linked list */
node * create (node *current)
{
  char ch;
  current -> prev = NULL;
  do
  {
  printf(" Enter the info = ");
  scanf("%d", &current -> info);
  printf(" do you want another node(y/n)");
    fflush(stdin);
    scanf("%c", &ch);
        if (ch == 'y')
  {
  current -> link = (node *) malloc (sizeof(node));
    (current -> link) -> prev = current;
      current = current -> link;
    }
    else
    {
      current -> link = NULL;
    }
  }
  while (ch == 'y');
  return (current);
}

/* display function */
void display(node *current)
{
  while (current != NULL)
  {
    printf("%d ", current -> info);
```

```
        current = current -> link;
      }
    }

void    displayrev(node *current)
{
    while (current != NULL)
    {
        printf(" %d ", current->info);
        current = current -> prev;
    }
}

/* Releaselist */
void    releaslist(node *current);
{
    node *temp;
        while( current != NULL)
        {
            temp = current;
            current = current -> link;
            free(temp);
        }
    free(temp);
}
```

output-

Enter the info = 6
do you want another node y/n = y
Enter the info = 7
do you want another node y/n = y
Enter the info = 8
Do you want another node y/n = y
Enter the info = 9
Do you want another node y/n = n

Forward order traversal
6 7 8 9
Reverse order traversal
9 8 7 6

Program-48

```
/* Factorial using recursion */
#include <stdio.h>
#include <conio.h>
int fact (int n)
void main( )
{
  int n,f;
  clrscr( );
    printf(" Enter a number = ");
      scanf("%d", &n);
    f=fact (n);
  printf(" The factorial is= %d ", f );
    getch( );
}

int fact (int m)
{
  if(m==0)
  return (1);
  else
  {
   return (n * fact (m-1));
  }
}
```

Output -

Enter a number = 5

The factorial is = 120

Program-49

```
/* Program to compute factorial using PUSH and
      POP operations in stack.
   #include <stdio.h>
   #include <conio.h>
   #define MAX 6

   /* Function declaration */
void  push(int [], int *, int);
int   pop(int [], int *);

void  main()
{
int  stack[MAX], fact = 1, top = -1, n;
clrscr();
   printf("Enter the no to calculate factorial=");
     scanf("%d", &n);
    while( n>0)
    {
        push (stack, &top, n);
        n--;
    }
    while(top != 1)
    {
        fact = fact *pop (stack, &top);
    }
printf("The Factorial is = %d ", fact);
getch();
}


/* Function for push */
void  push( int stack[], int *top, int)
{
   if(*top == MAX)
```

```
{
    printf(" Overflow");
    return;
}
else
{
    *top = *top + 1;
    stack [*top] = item;
}
}

/* Function for pop */
int pop(int stack[], int *top)
{
    int item;
    if (*top == -1)
    {
        printf("Underflow");
        return(-1);
    }
    else
    {
        item = stack[*top];
        *top = *top - 1;
        return(item);
    }
}
```

Output-

Enter the no. to calculate factorial = 5
The Factorial is = 120

Program - 50

```
/* Program to perform PUSH, POP & PEEK operations in
   stack.
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
  /* Function declaration */
void push(int [],int *, int);
int pop(int [],int *);
~~int peek~~ void main()
{
int stack[MAX], top = -1, c, item;
char ch;
do{
printf("Your choice are as \n");
printf(" 1. PUSH \n 2. POP \n 3. PEEK \n 4. EXIT \n");
printf(" Which operation you want to perform, Enter your
        choice = ");
scanf(" %d", &c);
switch(c)
{
case 1:
        printf(" Enter the item to push= ");
          scanf(" %d", &item);
        push(stack, &top, item);
        break;
case 2:
        item = pop(stack, &top);
        break;
case 3:
        PEEK is =
        printf(" %d", stack[top]);
        break;
case 4:
        getch();
        exit(0);
```

```
default:
        printf(" Wrong choice , try again\n");
}
printf(" Do you want another operations = ");
  fflush(stdin);
  scanf(" %c ", &ch);
} while(ch == 'y');
getch();
}
/* Function definition for push operation */
void push(int stack[], int *top, int item)
{
    if(*top == MAX)
    {
     printf(" \nOverflow");
     return;
    }
    else
    {
      (*top)++;
     stack [*top] = item;
    }
}

/* Function definition for pop operation */
~~voi~~ int pop(int stack[], int *top)
{
    int item;
    if( *top == -1)
     {
        printf("\n Underflow ");
        return;
     }
    else
```

```
{
    item = stack [*top];
    (*top)--;
  }
return(item);
}
```

output

Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 1
Enter the item to push = 2
Do you want another operations = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 1
Enter the item to push = 3
Do you want another operations = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 3
PEEK is = 3
Do you want another operations = y
Your choice are as
1. PUSH
2. POP

3. PEEK
4. EXIT

Which operation you want to perform, Enter your choice=2
Do you want another operation = y
Your choice are as:
1. PUSH
2. POP
3. PEEK
4. EXIT

Which operation you want to perform, Enter your choice = 3
PEEK is = 2
Do you want another operation = y
Which operation you want to perform, Enter your choice=4

Program - 51

```
/*Power using recursion */
#include <stdio.h>
#include<conio.h
int powl(int n,int m)
{
    if (m==0)
    return(1);
    else
    return(n* powl(n,m-1));
}
void main()
{
    int n,m;
    clrscr();
printf("Enter the no. and power=");
    scanf("%d %d", &n,&m);
printf("Result is %d ", powl(n,m));
getch();
}
```

choice=2

ice=3

hoice=4

output -

Enter the no. and power = 5 3
Result is 125

Program - 52

```
/* Program to demonstrate linked stack using function */
#include<stdio.h>
#include<conio.h>
typedef struct node {
int info;
struck node *link;
}node;
// function declaration
void push(node **, int);
int pop(node **);
int peek(node *);
void releaselist(node *);
void main()
{
  int item, c;
  char ch;
  node *top = NULL;
  clrscre();
  do
  {
  printf("your choice are as\n");
  printf("1. PUSH \n 2. POP \n 3. PEEK \n 4. EXIT \n");
printf("which operation you want to perform, Enter your choice
    scanf("%d", &c);
=");
switch(c)
{
case 1:
    printf("Enter item to push=");
      scanf("%d", &item);
    push(&top, item);
      break;
case 2:
      item = pop(&top);
```

```
        if(item != -1)
        printf("\n The pop item is = %d \n", item);
        break;
case 3:
        item = peek(top);
        if (item != -1)
        printf" The peek item is = %d \n", item);
        break;
case 4:
        exit(0);
        getch();
default:
        printf" Wrong choice, try again\n";
}
printf" Do you want another operations =");
  fflush(stdin);
  scanf("%c", &ch);
}while(ch=='y');
getch();
}
/*function definition for push operations in linked stack
void push(node **top, int item)
{
 node *newnode = NULL;
 newnode = (node *) malloc (sizeof(node));
 if(newnode == NULL)
 {
   printf" Overflow");
   return;
 }
 else
 {
   newnode -> info = item;
```

```
    newnode->link = *top;
    *top = newnode;
  }
}
/* Function definition for pop operation in linked stack */
int pop(node **top)
{
  int item;
  node *temp;
  if (top==NULL){
  printf("\n underflow");
   return;
  }
  else
  {
  item = (*top)->info;
  temp = *top;
  *top = (*top)->link;
  free(temp);
  return(item);
  }
return(0);
}
/* Function for peek operation in linked stack */
int peek(node *top)
{
  int item;
  if(top==NULL)
  return(-1);
  else
  return(item);
}
```

```
/* Function definition for release memory */
void releaselist(node *top)
{
  node *temp;
      while( top!= NULL)
      {
        temp=top;
        top = top->link;
          free(top);
      }
  ~~return~~ free(temp);
}
```

Output -

Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 1
Enter the item to push = 2
Do you want another operations = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 1
Enter the item to push = 3
Do you want another operations = y
Your choice are as
1. PUSH
2. POP

3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 2
The pop item is = 3
Do you want another operation = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 3
The peek item is = 2
Do you want another operation = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, Enter your choice = 4

choice = 1

choice = 1

Program-53

```
//Perform ENQUEUE and DEQUEUE operation on linear
    Queue-
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define MAX 9
void enqueue(int queue[], int *front, int *rear, int item);
int dequeue(int queue[], int *front, int *rear);
void main()
{
int queue[MAX], front=-1, rear=-1, c, item;
char ch;
do{
printf(" your choice are as \n");
printf(" 1. ENQUEUE \n 2.DEQUEUE\n 3.PEEK\n 4. EXIT \n");
printf(" which operation do you want to perform, Entry your choice");
scanf("%d", &ch);
switch(c){
{
case 1:
    printf" Enter the item to enqueue="";
      scanf("%d", &item);
    enqueue(queue, &front, &rear, item);
    break;
case 2:
    item=dequeue(queue, &front, &rear);
     if(item!=-1)
    printf(" \n Dequeue item is %d \n", queue[front]);
    break;
case 4:
    getch();
     exit(0);
case 3:
    if(item!=-1)
```

```
        printf(" PEEK is %d ",queue[front]);
        break;
    default:
        printf(" Wrong choice, Try again\n");
    }
    printf(" Do you want another operations = ");
        fflush(stdin);
        scanf("%c", &ch);
    }while(ch == 'y')
    getch();
    }
// ENQUEUE OPERATION
void enqueue(int queue[], int *front, int *rear, int item)
{
        if(*rear == MAX-1) {
        printf(" Overflow! QUEUE IS FULL...\n");
        return;
        }
        else
        {
          if(*front == -1)
          {
            *front = 0;
            *rear = 0;
          }
          else
          {
            *rear = *rear +1;
          }
        queue[*rear] = item;
    }
}
```

```
//DEQUEUE OPERATION
int dequeue(int queue[],int *front,int *rear)
{
  int item;
  if(*front==-1)
  {
   printf("Underflow!!! QUEUE IS EMPTY\n");
   return(-1);
  }
 else
  {
    item = queue[*front];
    if(*front==*rear)
    {
    *front = *rear = -1;
    }
    else
    {
     *front = *front +1;
    }
   return(item);
  }
}
```

output=
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice=2
Underflow !!! QUEUE IS EMPTY
Do you want another operations = y
Your choice are as
1. ENQUEUE
2 DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice=1
Enter the item to enqueue = 1
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice = 1
Enter the item to enqueue = 2
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice=1
Enter the item to enqueue = 3
Overflow ! QUEUE IS FULL...

Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform = 2
Dequeue item is 1
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you wan to perform = 3
PEEK is 2
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform = 1
Enter the item to enqueue = 2
Overflow! QUEUE IS FULL.
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice = 4

Program-54

```
// Perform ENQUEUE and DEQUEUE operation on
   circular queue -
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#define MAX 2
/* function declaration */
void enqueue(int queue[], int *front, int *rear, int item);
int dequeue(int queue[], int *front, int *rear);
void main()
{
int queue[MAX], front=-1, rear=-1, c, item;
char ch;
do
{
printf("Your choice are as \n");
printf("1. ENQUEUE \n 2. DEQUEUE \n 3. PEEK \n 4. EXIT \n");
printf("Which operation do you want to perform, enter your choice=");
scanf("%d", &ch);
switch(c)
{
case 1:
    printf("Enter the item to enqueue=");
        scanf("%d", item);
enqueue(queue, &front, &rear, item);
    break;
case 2:
    item = dequeue(queue, &front, &rear);
    if(item != -1)
printf("\n Dequeue item is %d \n", queue[item]);
    break;
case 3:
    if(item != -1)
    printf("PEEK is %d \n", queue[front]);
    break;
```

```
case 4:
        getch();
        exit(0);
default:
        printf("Wrong choice, Try again\n");
}
printf(" Do you want another operations = ");
        fflush(stdin);
    scanf("%c", &ch);
} while(ch == 'y');
getch();
}
```

ENQUEUE OPERATION

```
void enqueue(int queue[], int *front, int *rear, int item)
{
if(((*front == 0) && (*rear == MAX-1)) || (*front == *rear+1))
{
  printf(" overflow ");
  return;
}
else
{
    if(*front == -1)
    *rear = *front = 0;
    else
        if (*rear == MAX-1)
          *rear = 0;
        else
          *rear = *rear+1;
queue[*rear] = item;
}
}
```

```
// DEQUEUE Operation
int dequeue (int queue[], int *front, int *rear)
{
 int item;
 if(*front == -1)
 {
  printf(" Underflow\n");
  return(-1);
 }
 else
 {
     item = queue [*front];
          if (*front == *rear)
           *front = *rear = -1;
          else
           if (*front == MAX -1)
            *front = 0;
           else
            *front = *front +1;
 return(item);
 }
}
```

~~int peek(int queue[], int front)~~

Output -

Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you wan to perform, Enter your choice = 1
Enter the item to enqueue = 1
~~Which~~ Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice = 1
Enter the item to enqueue = 2
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice = 1
~~Do you want another operations~~
Enter the item to enqueue = 3
Overflow
Do you want another operations = y
Your choice are as
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT
Which operation do you want to perform, Enter your choice = 2

your choice = 1

Enter your choice = 1

your choice = 1

your choice = 2

Dequeue item is 1

Do you want another operation = y

Your choice are as

1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT

Which operation do you want to perform, enter your choice = 1

Enter the item to enqueue = 3

Do you want another operations = y

Your choice are as

1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT

Which operations do you want to perform, enter your choice = 3

PEEK is 2

Do you want another operations = y

Your choice are as

1. ENQUEUE
2. DEQUEUE
3. PEEK
4. EXIT

Which operations do you want to perform, enter your choice = 4

Program-55

```
/* Binary Search Recursive */
#include<stdio.h>
#include<conio.h>
#define MAX 10
/* function declaration */
void input(int a[],int n);
void display(int a[],int n);
int binarysearchrecursive(int a[],int lb,int ub,int item);
void main()
{
int a[MAX],lb,ub,item,n,loc;
clrscr();
printf("\n Enter the size of array=");
scanf("%d",&n);
input(a,n);
display(a,n);
printf("\n Enter the item to search=");
scanf("%d",&n);
loc= binarysearchrecursive(a,0,n-1,item);
if(loc!=-1)
printf("\n item %d is found at location %d",item,loc);
getche();
}
/* Function for input */
void input(int a[],int n)
{
int i;
for(i=0;i<=n-1;i++)
{
printf("\n Enter the value a[%d]=",i);
scanf("%d",&a[i]);
}
}
```

```
/* Function for display */
void display (int a[], int n)
{
  int i;
  for (i=0; i<=n-1; i++)
  {
    printf(" %d", a[i]);
  }
}
/* Function for binary search recursion */
int binarysearchrecursive (int a[], int lb, int ub, int item)
{
  int mid;
  if (lb > ub)
    return (-1);
  else {
    mid = (lb + ub) / 2;
  if (a[mid] == item)
      return (mid);
  else
    if (a[mid] > item)
      binarysearchrecursive (a, lb, mid-1);
    else
      binarysearchrecursive (a, mid+1, ub);
  }
}
```

Output

Enter the size of an array = 5
Enter the value a[0] = 1
Enter the value a[1] = 2
Enter the value a[2] = 3
Enter the value a[3] = 4
Enter the value a[4] = 5

1
2
3
4
5

Enter the item to search = 4
Item 4 is found at location 3

~~in C++~~ Program - 56

```
/* Program to push peek into stack and pop from
      stack.
#include< iostream.h >
#include< conio.h >
#include< process.h >
#define MAX 5
class st
{
 int stack [MAX], top;
 public:
 st()
 {
  top=-1;
 }
void push(int item);
 int pop;
 int peek;
};
void st:: push(int item)
 {
      if( top==MAX)
       {
         cout<<"overflow";
      }
      else
       {
         top++;
      stack [top]= item;
      }
return;
}
int st:: pop( )
{
```

```
{
    int    item;
    if(top==-1)
     {
        cout<<"Underflow";
        return(-1);
     }
    else
    {
        item= stack [top];
        top--;
    }
  return (item);
}
int st::peek()
{
    int item;
    if(top==-1)
    {
    cout<<" stack is empty";
      return(-1);
    }
    else
    item =stack [top];
      return(item);
}
void main()
{
int c,item;
char ch;
clrscr();
do
{
```

```
cout<<"Your choices :->\n";
cout<<"1->PUSH \n 2->POP\n 3-> PEEK \n 4-> EXIT";
cout<<"Which operation you want to perform, enter your choice";
  cin>>c;
 switch(c)
{
 case 1:
 cout<<"\n Enter the item to push=";
    cin >> item;
    s1.push(item);
    break;
 case 2:
    item = s1.pop();
    if(item != -1)
    cout<<"\n\n The item popped from stack "<<item;
    break;
 case 3:
    item = s1.peek();
    if (item != -1)
     cout <<"The peeked item is "<<item;
     break;
 case 4:
     getch();
     exit(0);
 default:
    cout<<"\n Wrong choice! Try again \n";
}
cout<<"\n Do you want another operation =";
 cin>>ch;
}
while (ch=='y');
getch();
}
```

Output-

Your choice are as
1 -> PUSH
2 -> POP
3 -> PEEK
4 -> EXIT
Which operation you want to perform, enter your choice = 1
Enter the item to push = 1
Do you want another operations = y
Your choice are as
1 -> PUSH
2 -> POP
3 -> PEEK
4 -> EXIT
Which operation you want to perform, enter your choice = 1
Enter the item to push = 2
Do you want another operations = y
Your choice are as
1 -> PUSH
2 -> POP
3 -> PEEK
4 -> EXIT
Which operation you want to perform, enter your choice = 3
The peeked item is = 2
Do you want another operations = y
Your choice are as
1 -> PUSH
2 -> POP
3 -> PEEK
4 -> EXIT
Which operation you want to perform, enter your choice = 2
The item popped from stack 2
Do you want another operations = y

Your choice are as
1 → PUSH
2 → POP
3 → PEEK
4 → EXIT
Which operation do you want to perform, enter your choice = 3
The peeked item is = 1
Do you want another operation = y
Your choice are as
1. PUSH
2. POP
3. PEEK
4. EXIT
Which operation you want to perform, enter your choice = 4.
—

Program - 57

```
/* WAP in C++ to perform enqueue and dequeue operation
      on circular queue */
#include <iostream.h>
#include <stdlib.h>
#include <process.h>
#define MAX 4
class q
{
  int front, rear, queue[MAX];
  public:
    q()
    {
    front = -1;
    rear = -1;
    }
  void enqueue(int item);
  int dequeue();
  int peek();
};
void q:: enqueue (int item)
{
if (((rear == MAX-1) && (front == 0)) || (front == rear+1))
{
  cout << endl << "overflow, queue is full" << endl;
return;
}
else
{
  if (from == -1)
  {
    front = 0;
    rear = 0;
}
```

```
else if( rear == MAX -1)
rear = 0;
else
{
 rear = rear +1;
}
queue[rear] = item;
}
}
int q::dequeue()
{
 int item;
  if( front = = -1)
  {
   cout << endl << "Underflow, queue is empty" << endl;
   return(-1);
  }
 else
  {
   item = queue[front];
    if( front == rear)
    {
     front = rear = -1;
    }
   else if (front == MAX-1)
    front = 0;
   else
    {
     front = front + 1;
    }
   return(item);
  }
}
```

```
int q::peek()
{
    if (front != -1)
    {
        return(queue[front]);
    }
    return(-1);
}
void main()
{
q ob;
int c;
int item;
char ch;
clrscr();
do
{
cout<<endl<<"1 -> ENQUEUE";
cout<<endl<<"2 -> DEQUEUE";
cout<<endl<<"3 -> PEEK";
cout<<endl<<"4 -> EXIT";
cout<<endl<<"which operation do you want to perform";
    cin>>c;
switch(c);
{
case 1:
    cout<<endl<<"enter the item to enqueue=";
        cin>>item;
    ob.enqueue(item);
        break;
case 2:
    item = ob.dequeue();
    if(item != -1)
```

```
cout << " Dequeue item is = " << item << endl;
break;
case 3:
cout << " PEEK of queue is = " << ob. peek() << endl;
break;
case 4:
getch();
exit(0);
default:
cout << end << "Wrong choice, try again" << endl;
}
cout << endl << "Do you want more operation y/n = ";
fflush(stdin);
cin >> ch;
}
while(ch == 'y');
getch();
}
```

Output -

1 -> ENQUEUE
2 -> DEQUEUE
3 -> PEEK
4 -> EXIT

Which operation do you want to perform = 1

Enter the item to enqueue = 1

Do you want more operations y/n = y

1. -> ENQUEUE
2 -> DEQUEUE
3 -> PEEK
4 -> EXIT

Which operation do you want to perform = 1

Enter the item to enqueue = 2

Do you want more operations y/n = y

1 -> ENQUEUE
2 -> DEQUEUE
3 -> PEEK
4 -> EXIT

Which operation do you want to perform = 3

PEEK of queue is = 1

Do you want more operations y/n = y

1 -> ENQUEUE
2 -> DEQUEUE
3 -> PEEK
4 -> EXIT

~~(you wo~~
Which operation do you want to perform = 2
Dequeue item is 1
Do you want more operation y/n = y

1 -> ENQUEUE
2 -> DEQUEUE
3 -> PEEK
4 -> EXIT

Which operation do you want to perform = 4