

NATIONAL INSTITUTE OF TECHNOLOGY KURUKSHETRA



Data Structures

Lab Programs

Submitted To:

Smruti Rekha Mam

Assistant Professor

Department of Computer Engineering B.Tech 2nd year

NIT KURUKSHETRA (COT-213)

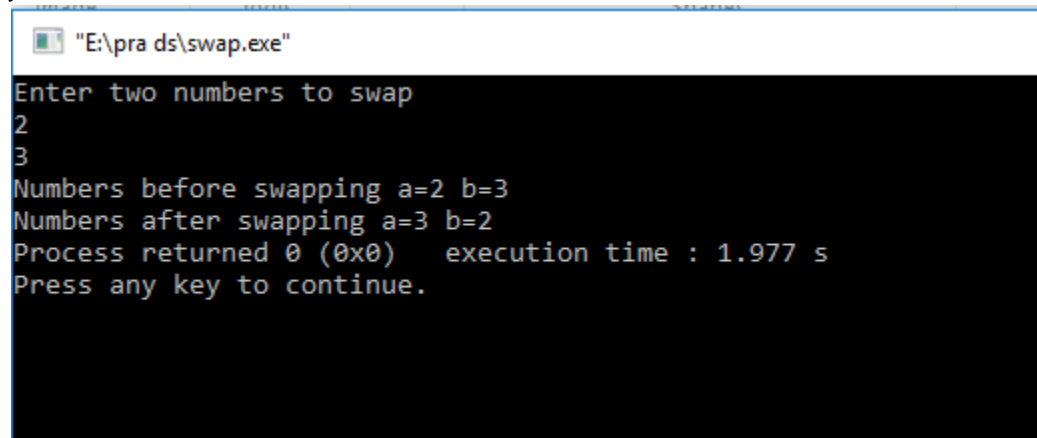
Submitted By:

Ginni Garg

11610559

(CO-4)

```
/*Swap to Two Numbers*/  
#include<stdio.h>  
void swap(int *,int * );  
int main()  
{  
    int a,b;  
    printf("Enter two numbers to swap\n");  
    scanf("%d%d",&a,&b);  
    printf("Numbers before swapping a=%d b=%d\n",a,b);  
    a=a^b;  
    b=a^b;  
    a=a^b;  
    printf("Numbers after swapping a=%d b=%d",a,b);  
    return 0;  
}
```



```
"E:\pra ds\swap.exe"  
Enter two numbers to swap  
2  
3  
Numbers before swapping a=2 b=3  
Numbers after swapping a=3 b=2  
Process returned 0 (0x0)   execution time : 1.977 s  
Press any key to continue.
```

```
/*Program to calculate Factorial*/
```

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

```
int main()
```

```
{int n;
```

```
printf("Enter n\n");
```

```
scanf("%d",&n);
```

```
printf("%d",fac(n));
```

```
return 0;
```

```
}
```

```
int fac(int a)
```

```
{static int b=1;
```

```
if(a==0)
```

```
return 1;
```

```
else
```

```
b=b*a;
```

```
if(a==2)
```

```
return b;
```

```
fac(a-1);
```

```
}
```



```
Enter n
```

```
5
```

```
120
```

```
Process returned 0 (0x0) execution time : 1.441 s
```

```
Press any key to continue.
```

```
_
```

```
/*Here sum of n elements of array is find in O(1)space complexity*/
```

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

```
int main()
```

```
{printf("Enter array size to find sum of all elements\n");
```

```
int sz;
```

```
scanf("%d",&sz);
```

```
int a,b,sum=0;
```

```
for(a=0;a<sz;a++)
```

```
{
```

```
scanf("%d",&b);
```

```
sum+=b;
```

```
}
```

```
printf("Sum of N numbers is:%d",sum);
```

```
return 0;
```

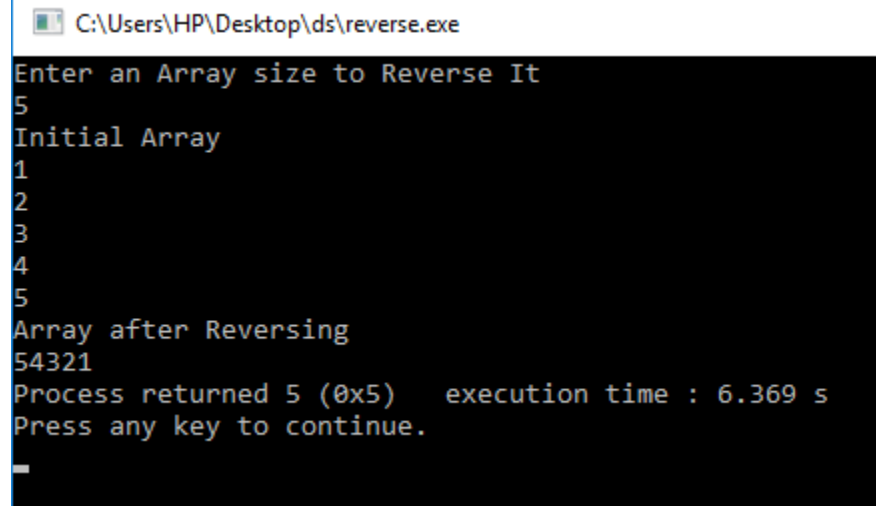
```
}
```

```
"C:\Users\HP\Desktop\ds\sum arr.exe"
Enter array size to find sum of all elements
5
1
2
3
4
5
Sum of N numbers is:15
Process returned 0 (0x0)   execution time : 6.200 s
Press any key to continue.
```

```
/*program to find the Trace of Matrix*/
#include<stdio.h>
int main()
{printf("Enter matrix order :row and column to find Trace of it\n");
int r,c;
scanf("%d%d",&r,&c);
int a,b,sum=0,matrix[r][c];
for(a=0;a<r;a++)
for(b=0;b<c;b++)
{scanf("%d",&matrix[a][b]);
if(a==b)
sum+=matrix[a][b];
}
printf("Trace of matrix:%d",sum);
return 0;
}
```

```
C:\Users\HP\Desktop\ds\trace.exe
Enter matrix order :row and column to find Trace of it
2
3
1
2
3
4
5
6
Trace of matrix:6
Process returned 0 (0x0)   execution time : 6.177 s
Press any key to continue.
```

```
/*Program to Reverse an Array*/
#include<stdio.h>
int main()
{printf("Enter an Array size to Reverse It\n");
int sz;
scanf("%d",&sz);
int arr[sz],i,j;
printf("Initial Array\n");
for(i=0;i<sz;i++)
scanf("%d",&arr[i]);
int tmp;
for(i=0,j=sz-1;i<j;i++,j--)
{tmp=arr[i];
arr[i]=arr[j];
arr[j]=tmp;
}
printf("Array after Reversing\n");
for(i=0;i<sz;i++)
printf("%d",arr[i]);
}
```



The screenshot shows a Windows command prompt window titled "C:\Users\HP\Desktop\ds\reverse.exe". The program prompts the user to "Enter an Array size to Reverse It", and the user enters "5". The program then displays "Initial Array" followed by the numbers "1", "2", "3", "4", and "5" on separate lines. After reversing the array, it displays "Array after Reversing" followed by the numbers "5", "4", "3", "2", and "1" on separate lines. The program concludes with the message "Process returned 5 (0x5) execution time : 6.369 s" and "Press any key to continue.", with a single underscore character visible on the line following the prompt.

```

/*program to find the row having minimum sum of elements of a Matrix*/
#include<stdio.h>
int main()
{printf("Enter matrix order :row and column to find Trace of it\n");
int r,c,tmp=0;
scanf("%d%d",&r,&c);
int a,min_r=0,b,sum=0,matrix[r][c];
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
{scanf("%d",&matrix[a][b]);
sum+=matrix[a][b];
}if(!a)
tmp=sum;
else
{tmp=tmp<sum?tmp:sum;
if(tmp==sum)
min_r=a;
}}
printf("\nIndex of Row having min sum of elements is:%d and sum is :%d",min_r,tmp);
return 0;
}

```

```

"C:\Users\HP\Desktop\ds\min row.exe"
Enter matrix order :row and column to find min row sum of it
2
3
1
2
3
4
5
6

Index of Row having min sum of elements is:0 and sum is :6
Process returned 0 (0x0)   execution time : 11.051 s
Press any key to continue.
_

```


```
/*Program to Find the Non-Repeating Elements of Array*/
#include<stdio.h>
int main()
{printf("Enter size of array\n");
int sz,count=0;
scanf("%d",&sz);
int a,b,arr[sz];
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
printf("Non repeating elements is:\n");
for(a=0;a<sz;a++)
{for(b=0;b<sz;b++)
if( a!=b && arr[a]==arr[b])
++count;
if(!count)
printf("%d ",arr[a]);
count=0;
}
}
```

```
"C:\Users\HP\Desktop\ds\non repetitine.exe"
Enter size of array
5
1
1
2
3
4
Non repeating elements is:
2 3 4
Process returned 5 (0x5)   execution time : 6.834 s
Press any key to continue.
```

```

/*program to find the Transpose of Matrix*/
#include<stdio.h>
int main()
{printf("Enter matrix order :row and column to find Transpose of matrix\n");
int r,c;
scanf("%d%d",&r,&c);
int a,b,matrix[r][c];
for(a=0;a<r;a++)
for(b=0;b<c;b++)
scanf("%d",&matrix[a][b]);
printf("Real Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
printf("%d ",matrix[a][b]);
printf("\n");
}
printf("Transpose of matrix\n");
int transpose[c][r];
for(a=0;a<r;a++)
for(b=0;b<c;b++)
transpose[b][a]=matrix[a][b];
for(a=0;a<c;a++)
{for(b=0;b<r;b++)
printf("%d ",transpose[a][b]);
printf("\n");
}
}

```

 "C:\Users\HP\Desktop\ds\matrix operation.exe"

```

Enter matrix order :row and column to find Transpose of matrix
2
3
1
2
3
4
5
6
Real Matrix
1 2 3
4 5 6
Transpose of matrix
1 4
2 5
3 6

Process returned 3 (0x3)   execution time : 10.391 s
Press any key to continue.

```



```

/*Program to do addition of Matrix*/
#include<stdio.h>
int main()
{printf("Enter matrix order :row and column to find addition it\n");
int r,c,r1,c1;
scanf("%d%d%d%d",&r,&c,&r1,&c1);
if(r!=r1 || c!=c1)
{
    printf("Matrix addition is not possible\n");
    return 0;
}
int a,b,mat1[r][c],mat2[r1][c1];
printf("Enter First Matrix\n");
for(a=0;a<r;a++)
for(b=0;b<c;b++)
scanf("%d",&mat1[a][b]);
printf("Enter Second Matrix\n");
for(a=0;a<r;a++)
for(b=0;b<c;b++)
scanf("%d",&mat2[a][b]);
int nmat[r][c];
for(a=0;a<r;a++)
for(b=0;b<c;b++)
nmat[a][b]=mat1[a][b]+mat2[a][b];
printf("First Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
printf("%d\t",mat1[a][b]);
printf("\n");
}
printf("Second Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
printf("%d\t",mat2[a][b]);
printf("\n");
}
printf("Final Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
printf("%d\t",nmat[a][b]);
printf("\n");
}
}

```

"C:\Users\HP\Desktop\ds\add matrix.exe"

Enter matrix order :row and column to find addition it

2

2

2

2

Enter First Matrix

1

2

3

4

Enter Second Matrix

1

2

3

4

First Matrix

| | |
|---|---|
| 1 | 2 |
|---|---|

| | |
|---|---|
| 3 | 4 |
|---|---|

Second Matrix

| | |
|---|---|
| 1 | 2 |
|---|---|

| | |
|---|---|
| 3 | 4 |
|---|---|

Final Matrix

| | |
|---|---|
| 2 | 4 |
|---|---|

| | |
|---|---|
| 6 | 8 |
|---|---|

Process returned 1 (0x1) execution time : 10.597 s

Press any key to continue.

```

/*program to do matrix mul of Matrix*/
#include<stdio.h>
int main()
{printf("Enter matrix order :row and column to find mul of it\n");
int r,c,r1,c1;
scanf("%d%d%d%d",&r,&c,&r1,&c1);
if(c!=r1)
{
    printf("Matrix Multiplication is not possible\n");
    return 0;
}
int a,b,mat1[r][c],mat2[r1][c1];
printf("Enter First Matrix\n");
for(a=0;a<r;a++)
for(b=0;b<c;b++)
scanf("%d",&mat1[a][b]);
printf("Enter Second Matrix\n");
for(a=0;a<r1;a++)
for(b=0;b<c1;b++)
scanf("%d",&mat2[a][b]);
int nmat[r][c1],d;
for(a=0;a<r;a++)
{
    for(b=0;b<c1;b++)
    {nmat[a][b]=0;
    for(d=0;d<c;d++)
    nmat[a][b]+=mat1[a][d]*mat2[d][b];
    }}
printf("First Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c;b++)
printf("%d\t",mat1[a][b]);
printf("\n");
}
printf("Second Matrix\n");
for(a=0;a<r1;a++)
{for(b=0;b<c1;b++)
printf("%d\t",mat2[a][b]);
printf("\n");
}
printf("Final Matrix\n");
for(a=0;a<r;a++)
{for(b=0;b<c1;b++)
printf("%d\t",nmat[a][b]);
printf("\n");
}
}

```

"C:\Users\HP\Desktop\ds\mul matrix.exe"

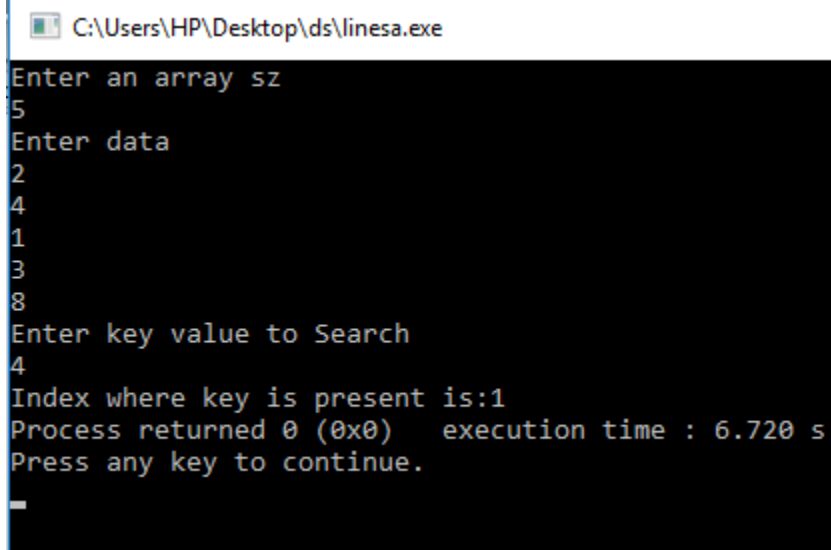
```
3
2
Enter First Matrix
1
2
3
4
5
6
Enter Second Matrix
3
7
8
9
5
6
First Matrix
1      2      3
4      5      6
Second Matrix
3      7
8      9
5      6
Final Matrix
34      43
82      109

Process returned 1 (0x1)   execution time : 11.069 s
Press any key to continue.
```

```

/*Program of Linear Search*/
#include<stdio.h>
int main()
{
printf("Enter an array sz\n");
int sz;
scanf("%d",&sz);
int a,arr[sz];
printf("Enter data\n");
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
printf("Enter key value to Search\n");
int key;
scanf("%d",&key);
for(a=0;a<sz;a++)
    if(arr[a]==key)
{printf("Index where key is present is:%d",a);
return 0;
}
printf("Key is Not Present\n")
return 0;
}

```



The screenshot shows a Windows command prompt window titled "C:\Users\HP\Desktop\ds\linesa.exe". The program prompts the user to "Enter an array sz", where the value 5 is entered. It then prompts "Enter data", where the values 2, 4, 1, 3, and 8 are entered on separate lines. Next, it prompts "Enter key value to Search", where the value 4 is entered. The program then outputs "Index where key is present is:1". Below this, it shows "Process returned 0 (0x0) execution time : 6.720 s" and "Press any key to continue." followed by a single underscore character.

```

C:\Users\HP\Desktop\ds\linesa.exe
Enter an array sz
5
Enter data
2
4
1
3
8
Enter key value to Search
4
Index where key is present is:1
Process returned 0 (0x0)   execution time : 6.720 s
Press any key to continue.
_

```

```

/*Program of Binary Search*/
#include<stdio.h>
int b_search(int,int*,int);
int main()
{printf("Enter an array sz\n");
int sz;
scanf("%d",&sz);
int a,arr[sz];
printf("Enter data in sorted ordered\n");
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
printf("Enter key value to Search\n");
int key;
scanf("%d",&key);
int value=b_search(key,arr,sz);
if(value!=-1)
printf("Index Where key is present is:%d",value);
else
printf("key is not present\n");
return 0;
}
int b_search(int key,int *a,int sz)
{int mid,l=0,r=sz-1;
while(l<=r)
{mid=(l+r)/2;
if(a[mid]==key)
return mid;
else if(a[mid]>key)
r=mid-1;
else
l=mid+1;
}
return -1;
}

```

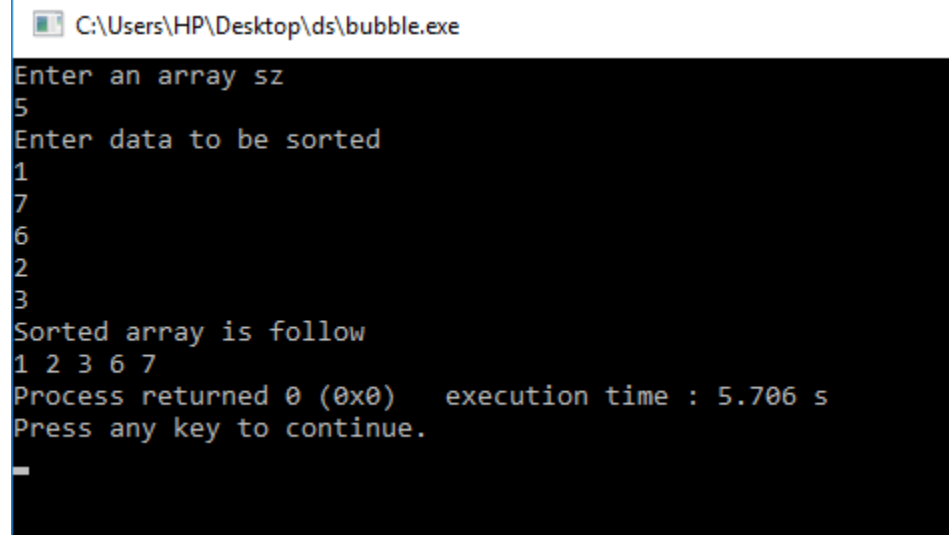
```

"C:\Users\HP\Desktop\ds\binary.exe"
Enter an array sz
5
Enter data in sorted ordered
1
2
6
7
8
Enter key value to Search
9
key is not present

Process returned 0 (0x0)   execution time : 6.469 s
Press any key to continue.

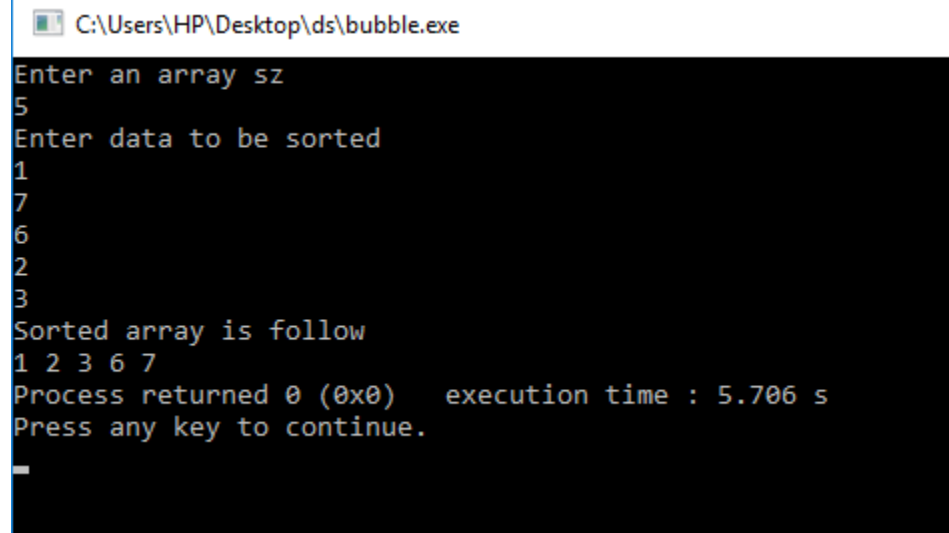
```

```
/*Program of selection sort*/
#include<stdio.h>
int main()
{printf("Enter an array sz\n");
int sz;
scanf("%d",&sz);
int a,b,tmp,arr[sz];
printf("Enter data to be sorted\n");
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
for(a=0;a<sz-1;a++)
    for(b=a+1;b<sz;b++)
        if(arr[a]>arr[b])
            tmp=arr[a],arr[a]=arr[b],arr[b]=tmp;
printf("Sorted array is follow\n");
for(a=0;a<sz;a++)
    printf("%d ",arr[a]);
return 0;
}
```



```
C:\Users\HP\Desktop\ds\bubble.exe
Enter an array sz
5
Enter data to be sorted
1
7
6
2
3
Sorted array is follow
1 2 3 6 7
Process returned 0 (0x0)   execution time : 5.706 s
Press any key to continue.
_
```

```
/*program of Bubble Sort*/
#include<stdio.h>
int main()
{printf("Enter an array sz\n");
int sz;
scanf("%d",&sz);
int a,b,tmp,arr[sz];
printf("Enter data in to be sorted\n");
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
int count =0;
for(a=0;a<sz-1;a++)
{
for(b=0;b<sz-a-1;b++)
{if(arr[b]>arr[b+1])
{tmp=arr[b],arr[b]=arr[b+1],arr[b+1]=tmp;
++count;
}}
if(!count)
break;
count=0;
}
printf("Sorted array is as Follow\n");
for(a=0;a<sz;a++)
printf("%d ",arr[a]);
return 0;
}
```



The screenshot shows a Windows command prompt window with the title bar "C:\Users\HP\Desktop\ds\bubble.exe". The program's output is displayed in a black terminal window with green text. The user enters '5' for the array size and '1 7 6 2 3' for the data. The program outputs the sorted array '1 2 3 6 7' and shows the execution time as 5.706 seconds.

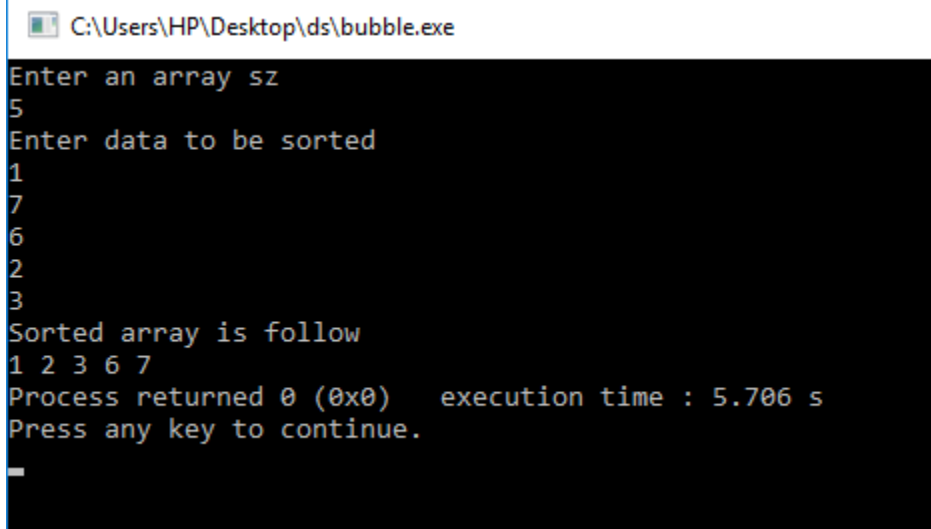
```
C:\Users\HP\Desktop\ds\bubble.exe
Enter an array sz
5
Enter data to be sorted
1
7
6
2
3
Sorted array is follow
1 2 3 6 7
Process returned 0 (0x0)   execution time : 5.706 s
Press any key to continue.
```



```

/*Program of Insert Sort*/
#include<stdio.h>
int main()
{printf("Enter an array sz\n");
int sz;
scanf("%d",&sz);
int a,b,tmp,arr[sz];
printf("Enter data in to be sorted\n");
for(a=0;a<sz;a++)
scanf("%d",&arr[a]);
for(a=0;a<sz-1;a++)
{
    tmp=arr[a+1];
    for(b=a+1;b>0;b--)
    {
        if(arr[b]<arr[b-1])
            tmp=arr[b-1],arr[b-1]=arr[b],arr[b]=tmp;
        else break;
    }
}
printf("Sorted array is as Follow\n");
for(a=0;a<sz;a++)
    printf("%d ",arr[a]);
return 0;
}

```



```

C:\Users\HP\Desktop\ds\bubble.exe
Enter an array sz
5
Enter data to be sorted
1
7
6
2
3
Sorted array is follow
1 2 3 6 7
Process returned 0 (0x0)   execution time : 5.706 s
Press any key to continue.

```

```

/*Program to sort a array using Quick Sort*/
#include<stdio.h>
void quick(int *,int ,int );
int partition(int *,int ,int );
int main()
{printf("Enter Size of Array\n");
int s;
scanf("%d",&s);
int a,arr[s];
for(a=0;a<s;a++)
scanf("%d",&arr[a]);
quick(arr,0,s-1);
for(a=0;a<s;a++)
printf("%d ",arr[a]);
return 0;
}
void quick(int *a,int first,int last)
{int q;
if(first<last)
{
q=partition(a,first,last);
quick(a,first,q-1);
quick(a,q+1,last);
}
}
int partition(int *a,int f,int l)
{
int b,tmp,i=f-1,j;
for(j=f;j<l;j++)
{
if(a[j]<=a[l])
{++i;
tmp=a[i],a[i]=a[j],a[j]=tmp;
}}
tmp=a[i+1],a[i+1]=a[l],a[l]=tmp;
return i+1;
}

```

```
C:\Users\HP\Desktop\ds\bubble.exe
Enter an array sz
5
Enter data to be sorted
1
7
6
2
3
Sorted array is follow
1 2 3 6 7
Process returned 0 (0x0)   execution time : 5.706 s
Press any key to continue.
_
```

/*Program to sort a array using merge sort*/

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

```
void merge(int * ,int,int ,int *);
```

```
void mrg(int * ,int,int,int,int *);
```

```
void copy(int * ,int,int ,int *);
```

```
int main()
```

```
{
```

```
printf("Enter size\n");
```

```
int s,q;
```

```
scanf("%d",&s);
```

```
int arr[s],ar[s];
```

```
for(q=0;q<s;q++)
```

```
scanf("%d",&arr[q]);
```

```
merge(arr,0,s-1,ar);
```

```
for(q=0;q<s;q++)
```

```
printf("%d ",arr[q]);
```

```
}
```

```
void merge(int *a,int s,int l,int *ar)
```

```
{int mid;
```

```
if(s<l)
```

```
{mid=(s+l)/2;
```

```
merge(a,s,mid,ar);
```

```
merge(a,mid+1,l,ar);
```

```
mrg(a,s,mid,l,ar);
```

```
}
```

```
}
```

```
void mrg(int *a,int s,int mid,int l,int *ar)
```

```
{int i=s,j=mid+1,n=s;
```

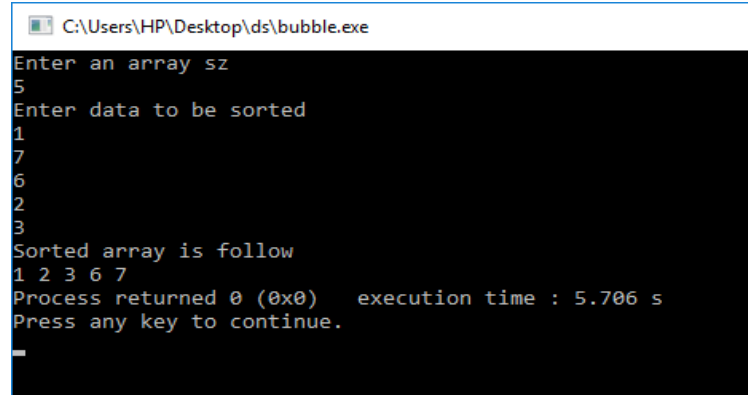
```
while(i<=mid && j<=l)
```

```
{
```

```
if(a[i]<=a[j])
```

```
ar[n++]=a[i++];
```

```
        else if(a[i]>=a[j])
            ar[n++]=a[j++];
    }
    while(i<=mid)
        ar[n++]=a[i++];
    while(j<=l)
        ar[n++]=a[j++];
    copy(a,s,l,ar);
}
void copy(int *a,int s,int l,int *ar)
{int x;
  for(x=s;x<=l;x++)
    a[x]=ar[x];
}
```




```
C:\Users\HP\Desktop\ds\bubble.exe
Enter an array sz
5
Enter data to be sorted
1
7
6
2
3
Sorted array is follow
1 2 3 6 7
Process returned 0 (0x0)   execution time : 5.706 s
Press any key to continue.
_
```

```

/*Creation of Sparse Matrix*/
#include<stdio.h>
#include<stdlib.h>
struct matrix* create(int (*a)[],int r,int c);
struct matrix{
int row;
int col;
int value;
};
int c;
int main()
{printf("Enter matrix Row and Column Size\n");
int r;
scanf("%d%d",&r,&c);
int arr[r][c],i,j,tmp;
printf("Enter Values in Matrix\n");
for(i=0;i<r;i++)
    for(j=0;j<c;j++)
{scanf("%d",&tmp);
    arr[i][j]=tmp;
}
struct matrix *t=create(arr,r,c);
for(i=0;i<=t[0].value;i++)
    printf("%d %d %d\n",t[i].row,t[i].col,t[i].value);
    free(t);
return 0;
}
struct matrix *create(int (*a)[c],int r,int c)
{int sz=1;
    struct matrix *t=(struct matrix *)malloc(sz*sizeof(struct matrix));
    int i,j,n=0;
for(i=0;i<r;i++)
    for(j=0;j<c;j++)
        if(a[i][j])
            {t=realloc(t,++sz*sizeof(struct matrix));
            t[++n].row=i,t[n].col=j,t[n].value=a[i][j];
}t[0].value=n,t[0].row=r,t[0].col=c;
return t;
}

```

 "C:\Users\HP\Desktop\New folder (2)\sparse1_cr.exe"

```
Enter matrix Row and Column Size
2
3
Enter Values in Matrix
1
2
3
4
5
0
2 3 5
0 0 1
0 1 2
0 2 3
1 0 4
1 1 5

Process returned 0 (0x0)   execution time : 8.713 s
Press any key to continue.
```

```
/*Addition of Sparses Matrix*/
#include<stdio.h>
#include<stdlib.h>
struct matrix{
int row;
int col;
int value;
};
int c;
struct matrix* create(int (*)[],int ,int );
struct matrix*add(struct matrix *,struct matrix *);
int main()
{printf("Enter matrix Row and Column Size\n");
int r,r1,c1;
scanf("%d%d%d%d",&r,&c,&r1,&c1);
if(r1!=r || c1!=c)
{
printf("Matrix addition is not valid\n"); return 0;

}
int arr[r][c],arr1[r1][c1],i,j,tmp;
printf("Enter Values in First Matrix\n");
for(i=0;i<r;i++)
for(j=0;j<c;j++)
scanf("%d",&arr[i][j]);
```

```

struct matrix *t=create(arr,r,c);
printf("Enter Values in First Matrix\n");
for(i=0;i<r1;i++)
    for(j=0;j<c1;j++)
        scanf("%d",&arr1[i][j]);
struct matrix *t2=create(arr1,r1,c1);
struct matrix *t3=add(t,t2);
for(i=0;i<=t[0].value;i++)
    printf("%d %d %d\n",t3[i].row,t3[i].col,t3[i].value);
    free(t);
return 0;
}
struct matrix *create(int (*a)[c],int r,int c)
{int sz=1;
    struct matrix *t=(struct matrix *)malloc(sz*sizeof(struct matrix));
    int i,j,n=0;
for(i=0;i<r;i++)
    for(j=0;j<c;j++)
        if(a[i][j])
            {t=realloc(t,++sz*sizeof(struct matrix));
                t[++n].row=i,t[n].col=j,t[n].value=a[i][j];
            }t[0].value=n,t[0].row=r,t[0].col=c;
return t;
}
struct matrix *add(struct matrix *t1,struct matrix *t2)
{int s=1,r=1,n=0;
struct matrix *t=(struct matrix *)malloc((t1[0].value+t2[0].value)*sizeof(struct matrix));
    while(s<=t1[0].value && r<=t2[0].value)
        if(t1[s].row>t2[r].row)
            t[++n].row=t2[r].row,t[n].col=t2[r].col,t[n].value=t2[r++].value;
        else if(t1[s].row<t2[r].row)
            t[++n].row=t1[s].row,t[n].col=t1[s].col,t[n].value=t1[s++].value;
    else{
        if(t1[s].col<t2[r].col)
            t[++n].row=t1[s].row,t[n].col=t1[s].col,t[n].value=t1[s++].value;
        else if(t1[s].col>t2[r].col)
            t[++n].row=t2[r].row,t[n].col=t2[r].col,t[n].value=t2[r++].value;
        else
            t[++n].row=t1[s].row,t[n].col=t1[r].col,t[n].value=t1[s++].value+t2[r++].value;
    }
while(s<=t1[0].value)
    t[++n].row=t1[s].row,t[n].col=t1[s].col,t[n].value=t1[s++].value;
while(r<=t2[0].value)
    t[++n].row=t2[r].row,t[n].col=t2[r].col,t[n].value=t2[r++].value;
t[0].value=n;t[0].row=t1[0].row;t[0].col=t1[0].col;
return t;}

```

```

C:\Users\HP\Desktop\ds\spar.add.exe
Enter matrix Row and Column Size
2
3
2
3
Enter Values in First Matrix
1
2
3
4
5
6
Enter Values in First Matrix
2
3
4
5
6
7
2 3 6
0 0 3
0 1 5
0 2 7
1 0 9
1 1 11
1 2 13

Process returned 0 (0x0)   execution time : 14.418 s
Press any key to continue.

```

```

/*Program to add sparse polynomial*/
#include<stdio.h>
struct poly{
int cof;
int expo;
};
void add(struct poly*,struct poly *,struct poly *,int,int, int *);
int main()
{
printf("Enter sz of polynomials");
int s1,s2,n=0;
scanf("%d%d",&s1,&s2);
printf("Enter first Polynomial: cof and expo\n");
struct poly p1[s1];
struct poly pp1[s1];
int a;
for(a=0;a<s1;a++)

```



```


{scanf("%d%d",&p1[a].cof,&p1[a].expo);
if(p1[a].cof)
{pp1[n].expo=p1[a].expo;
pp1[n++].cof=p1[a].cof;
}}
s1=n;n=0;
printf("Enter 2nd Polynomial:cof and expo\n");
struct poly p2[s2];
struct poly pp2[s2];
for(a=0;a<s2;a++)
{scanf("%d%d",&p2[a].cof,&p2[a].expo);
if(p2[a].cof)
{pp2[n].expo=p2[a].expo;
pp2[n++].cof=p2[a].cof;
}}s2=n;
struct poly pp3[s1+s2];int s3;
add(pp1,pp2,pp3,s1,s2,&s3);
printf("First poly as Follow:Expo and cof\n");
for(a=0;a<s1;a++)
    printf("%d %d\n",pp1[a].expo,pp1[a].cof);
printf("2nd poly as Follow:Expo and cof\n");
for(a=0;a<s2;a++)
    printf("%d %d\n",pp2[a].expo,pp2[a].cof);
printf("Added poly as Follow:Expo and cof\n");
for(a=0;a<s3;a++)
    printf("%d %d\n",pp3[a].expo,pp3[a].cof);

return 0;
}

void add(struct poly *p1,struct poly *p2,struct poly *p3,int s1,int s2,int *s3)
{int s=0,r=0,n=0;
while(s<s1 && r<s2)
{
if(p1[s].expo>p2[r].expo)
p3[n].expo=p1[s].expo,p3[n++].cof=p1[s++].cof;
else if(p1[s].expo<p2[r].expo)
p3[n].expo=p2[r].expo,p3[n++].cof=p2[r++].cof;
else
if(p1[s].cof+p2[r].cof)
{
    p3[n].expo=p1[s].expo;p3[n++].cof=p1[s++].cof+p2[r++].cof;
}
else
{++s;++r;
}
}
while(s<s1)
p3[n].expo=p1[s].expo,p3[n++].cof=p1[s++].cof;

```

```
while(r<s2)
p3[n].expo=p2[s].expo,p3[n++].cof=p2[r++].cof;
*s3=n;
}
```

 "C:\Users\HP\Desktop\ds\poly add.exe"

```
1
2
0
0
Enter 2nd Polynomial:cof and expo
2
5
4
6
6
7
First poly as Follow:Expo and cof
4 1
3 1
2 1
2nd poly as Follow:Expo and cof
5 2
6 4
7 6
Added poly as Follow:Expo and cof
5 2
5 4
5 6
4 1
3 1
2 1

Process returned 0 (0x0)   execution time : 22.695 s
Press any key to continue.
```

```

/*postfix Evaluation*/
#include<stdio.h>
#include<string.h>
void check(char);
int stack[30];
void push(int );
int pop();
int top=-1;
int main()
{printf("Enter Expression to evaluate in postfix\n");
char postfix[26];
gets(postfix);
int a;
for(a=0;a<strlen(postfix);a++)
check(postfix[a]);
printf("value is:%d",pop());
return 0;
}
void check(char a)
{int n,z;
switch(a)
{
case '+':
n=pop();z=pop();
push(z+n);
break;
case '-':
n=pop();z=pop();
push(z-n);
break;
case '*':
n=pop();z=pop();
push(z*n);
break;
case '/':
n=pop();z=pop();
push(z/n);
break;
case '%':
n=pop();z=pop();
push(z%n);
break;
default:push(a-'0');
}
}
void push(int key)
{


```

```

if(top==30)
{
    printf("Stack full");
    exit(1);
}
stack[++top]=key;
}
int pop()
{

    if(top== -1)
    {
        printf("Stack is empty");
        return -1;
    }
    return stack[top--];
}

```

 C:\Users\HP\Desktop\ds\postdix.exe

```

Enter Expression to evaluate in postfix
23+5+6*
value is:60
Process returned 0 (0x0)   execution time : 9.239 s
Press any key to continue.

```

```

/*Program of Circular Queue*/
#include<stdio.h>
#include<stdlib.h>
void isempty();
void isfull();
static int x;
int pop();
void push(int);
int front,rear;
int *queue,capacity=1;
int main()
{queue=(int *)malloc(sizeof(int));
    int a,n;
    printf("Enter number of values Required\n");
    int sz;
    scanf("%d",&sz);
    for(a=0;a<sz;a++)
    {scanf("%d",&n);
        push(n);
    }
}

```

```

    }
    printf("pop values from queue\n");
    for(a=0;a<=sz;a++)
        printf("%d ",pop());

return 0;
}
void push(int value)
{
if(front==rear && x==capacity)
isfull();
rear=(rear+1)%capacity; //*****
++x;
queue[rear]=value;
}
void isfull()
{
capacity*=2;
int *tmp,*que=(int *)malloc(sizeof(int)*capacity);
int a,n=1;
for(a=front+1;a<capacity/2;a++)
que[n++]=queue[a];
for(a=0;a<=rear;a++)
que[n++]=queue[a];
front=0;
rear=n-1;
tmp=queue;
queue=que;
free(tmp);
tmp=NULL;
que=NULL;
}
int pop()
{if(rear==front && x==0)
    isempty();
front=(front+1)%capacity; //*****
--x;
return queue[front];
}
void isempty()
{
    printf("\nQ is Empty");
    exit(1);
}

```

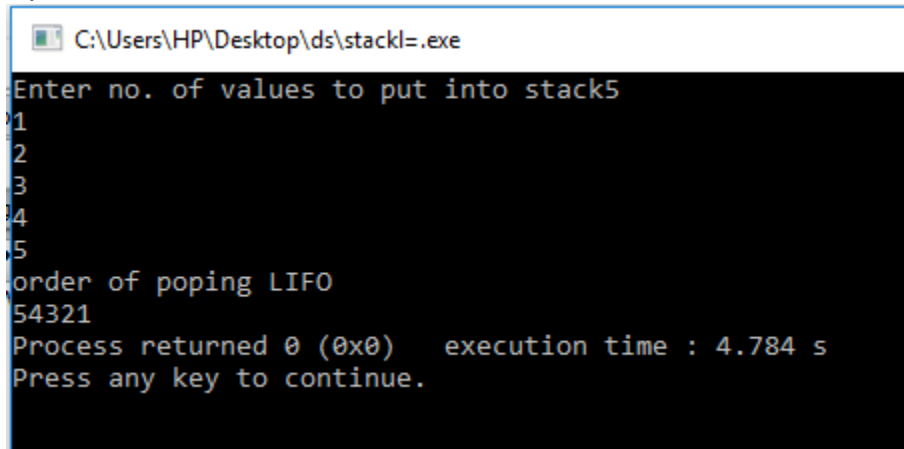
```
"C:\Users\HP\Desktop\New folder (2)\circular queue.exe"
Enter number of values Required
5
1
6
7
5
4
pop values from queue
1 6 7 5 4
Q is Empty
Process returned 1 (0x1)   execution time : 4.661 s
Press any key to continue.
```

```
/*program of stack*/
#include<stdio.h>
#include<string.h>
int stack[30];void push(int );
int pop();
int top=-1;
int main()
{int v;
printf("Enter no. of values to put into stack");
int a,value;
scanf("%d",&a);
for(v=0;v<a;v++)
{scanf("%d",&value);
push(value);
}
printf("order of popping LIFO\n");
for(v=0;v<a;v++)
printf("%d",pop());
return 0;
}
void push(int key)
{
if(top==30)
{
printf("Stack full");
exit(1);
}
stack[++top]=key;
}
int pop()
{
```

```

    if(top== -1)
    {
        printf("Stack is empty");
        return -1;
    }
    return stack[top--];
}

```



The screenshot shows a Windows command prompt window titled "C:\Users\HP\Desktop\ds\stackl=.exe". The program prompts the user to "Enter no. of values to put into stack" and the user enters "5". The program then prompts for the "order of popping LIFO" and the user enters "54321". The output shows "Process returned 0 (0x0) execution time : 4.784 s" and "Press any key to continue.".

```

C:\Users\HP\Desktop\ds\stackl=.exe
Enter no. of values to put into stack
5
order of popping LIFO
54321
Process returned 0 (0x0) execution time : 4.784 s
Press any key to continue.

```

```


/*Creation and Transversal of Linked list*/
#include<stdio.h>
#include<stdlib.h>
struct ll{
int info;
struct ll *next;
};
struct ll*create(int );
void transverse(struct ll*);
int main()
{printf("Enter sz of ll to create it");
int sz;
scanf("%d",&sz);
struct ll*l=create(sz);
transverse(l);
return 0;
}
struct ll *create(int sz)
{if(!sz)
return NULL;
struct ll *tmp,*tmp2,*q=(struct ll *)malloc(sizeof(struct ll));
int value;
printf("Enter key value\n");

```

```

scanf("%d",&value);
q->info=value;
if(!q)
{printf("No mm");return NULL;}
tmp2=q;
while(sz-1)
{tmp=(struct ll *)malloc(sizeof(struct ll));
if(!tmp){printf("No mm");return NULL;}
tmp2->next=tmp;
printf("Enter key value\n");
scanf("%d",&value);
tmp->info=value;
tmp2=tmp;
--sz;
}
tmp2->next=NULL;
return q;
}
void transverse(struct ll*p)
{if(!p)
return ;
struct ll*tmp=p;
while(tmp)
{
printf("\n%d",tmp->info);
tmp=tmp->next;
}
}

```

 C:\Users\HP\Desktop\ds\Untitled7.exe

```

Enter sz of ll to create it5
Enter key value
1
Enter key value
3
Enter key value
2
Enter key value
5
Enter key value
4

1
3
2
5
4
Process returned 0 (0x0)   execution time : 3.553 s
Press any key to continue.

```



```

/*Creation , Transversal ,Inserton and Deletion of LLS*/
#include<stdio.h>
#include<stdlib.h>
struct ll{
int info;
struct ll *next;
};
struct ll*create(int );
void transverse(struct ll*);
struct ll*delete(struct ll*);
struct ll*insert_b(struct ll*,struct ll*,int );
int main()
{printf("Enter sz of ll to create it\n");
int sz;
scanf("%d",&sz);
struct ll*l=create(sz);
printf("Before Insertion\n");
transverse(l);
printf("\nEnter a key value to insert and node no,\n");
int value,n;
scanf("%d%d",&value,&n);
struct ll*tmp=l;
int a;
for(a=0;a<n-1;a++)
    tmp=tmp->next;
l=insert_b(tmp,l,value);
printf("After Insertion\n");
transverse(l);
printf("\nif Wanted to deltte LL press 1\n");
scanf("%d",&a);
if(a==1)
{
    l=delete(l);
    transverse(l);
}
return 0;
}
struct ll *create(int sz)
{if(!sz)
return NULL;
struct ll *tmp,*tmp2,*q=(struct ll *)malloc(sizeof(struct ll));
int value;
printf("Enter key value\n");
scanf("%d",&value);
q->info=value;
if(!q)
{printf("No mm");return NULL;}
tmp2=q;

```

```

while(sz-1)
{tmp=(struct ll *)malloc(sizeof(struct ll));
if(!tmp){printf("No mm");return NULL;}
tmp2->next=tmp;
printf("Enter key value\n");
scanf("%d",&value);
tmp->info=value;
tmp2=tmp;
--sz;
}
tmp2->next=NULL;
return q;
}
void transverse(struct ll*p)
{if(!p)
{
printf("LL is Empty\n");
return ;
}struct ll*tmp=p;
while(tmp)
{
printf("\n%d",tmp->info);
tmp=tmp->next;
}
}
struct ll* insert_b(struct ll*x,struct ll*p,int key)
{if(!x) //Necessary*
return p ;
struct ll*tmp2=(struct ll*)malloc(sizeof(struct ll));
if(!tmp2)
return NULL;
if(x==p)
{tmp2->next=x;
tmp2->info=key;
p=tmp2;
return p;
}
struct ll*tmp=p;
while(tmp &&tmp->next!=x)
tmp=tmp->next;
if(!tmp)
{
printf("\nNode is not present after which we have to insert");
return p;
}
tmp2->next=x;
tmp->next=tmp2;
tmp2->info=key;

```

```

return p;
}
struct ll*delete(struct ll*q)
{struct ll*tmp;
while(q)
{tmp=q;
q=q->next;
free(tmp);
}
return q;
}

```

```

Enter sz of ll to create it
5
Enter key value
1
Enter key value
2
Enter key value
5
Enter key value
4
Enter key value
3
Before Insertion

1
2
5
4
3
Enter a key value to insert and node no,
10
1
After Insertion

10
1
2
5
4
3
if Wanted to delttte LL press 1
1
LL is Empty

```

```
/*program for insertion in tree ,checking equality of two Trees,Transversal and Creation of Tree and Counting No. of Leaf Nodes*/
```

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

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

```
struct t{
```

```
struct t*l;
```

```
int value;
```

```
struct t*r;
```

```
};
```

```
struct t*insert_tree(int,struct t* );
```

```
void inorder(struct t*);
```

```
struct t*create(int );
```

```
int count_l(struct t*);
```

```
int similar(struct t*,struct t*);
```

```
int main()
```

```
{struct t *p=NULL,*q=NULL;
```

```
printf("Enter values in array to make a tree\n");
```

```
int n,idx1,idx2;
```

```
scanf("%d%d",&idx1,&idx2);
```

```
int arr[idx1],arr2[idx2];
```

```
printf("Enter first Tree\n");
```

```
for(n=0;n<idx1;n++)
```

```
scanf("%d",&arr[n]);
```

```
printf("Enter second tree\n");
```

```
for(n=0;n<idx2;n++)
```

```
scanf("%d",&arr2[n]);
```

```
for(n=0;n<idx1;n++)
```

```
p=insert_tree(arr[n],p);
```

```
for(n=0;n<idx1;n++)
```

```
q=insert_tree(arr2[n],q);
```

```
printf("inorder Transversal in First Tree\n");
```

```
inorder(p);
```

```
printf("\ninorder Transversal in 2nd Tree\n");
```

```
inorder(q);
```

```
printf("\nChecking Whether Equal or not\n");
```

```
if(similar(q,p))
```

```
    printf("Equal\n");
```

```
else
```

```
    printf("Not Equal\n");
```

```
printf("Counting No. of leaf Nodes\n");
```

```
n =count_l(p);
```

```
int m=count_l(q);
```

```
printf("First Tree:%d and Second Tree:%d",n,m);
```

```
return 0;
```

```
}
```

```
int count_l(struct t*root)
```

```
{
```

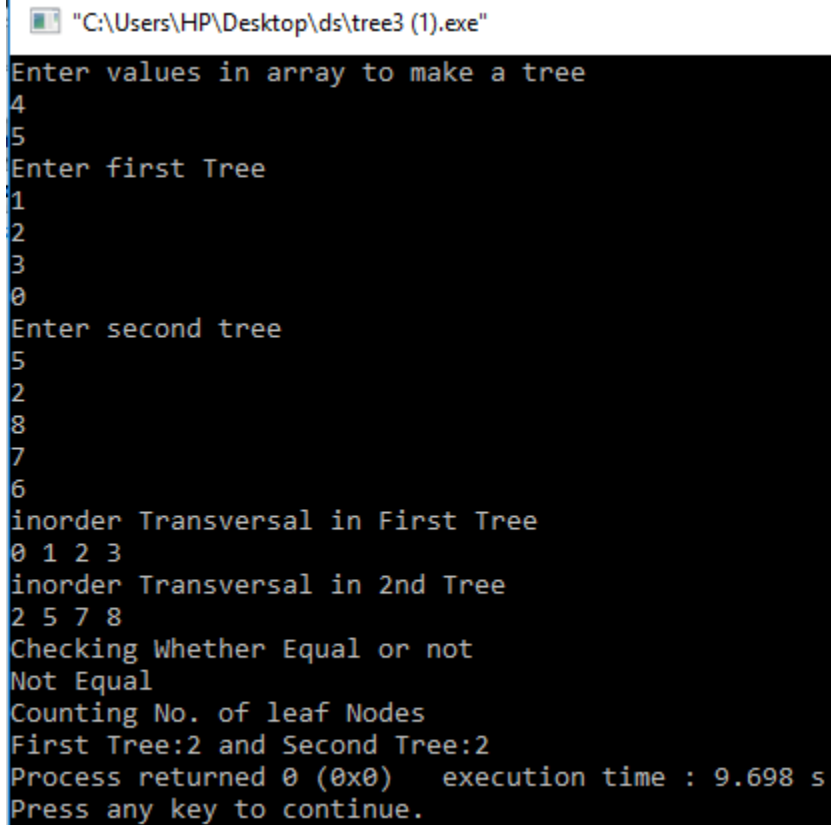
```
int c=0;
```

```

if(root)
{if(!root->l && !root->r)
return 1;
if(root->l)
c=c+count_l(root->l);
if(root->r)
c=c+count_l(root->r);
return c;
}
return 0;
}
struct t*insert_tree(int n,struct t*p)
{struct t*tmp=p;
if(!p)
{return create(n);
}
for(;;)
{if(tmp->value==n)
return p;
else if(tmp->value<n)
{if(tmp->r)
tmp=tmp->r;
else
{tmp->r=create(n);
return p;
}}
else
{
if(tmp->l)
tmp=tmp->l;
else
{tmp->l=create(n);
return p;
}}}
}
struct t*create(int a)
{struct t*tmp;
tmp=(struct t*)malloc(sizeof(struct t));
tmp->value=a;
tmp->l=tmp->r=NULL;
return tmp;}
int similar(struct t*p,struct t*q)
{
return ((!p && !q) || (p && q)&&(p->value==q->value)&&similar(p->l,q->l)&&similar(p->r,q->r));
}
void inorder(struct t*p)
{

```

```
if(p)
{
    inorder(p->l);
    printf("%d ",p->value);
    inorder(p->r);
}
}
```



```
"C:\Users\HP\Desktop\ds\tree3 (1).exe"
Enter values in array to make a tree
4
5
Enter first Tree
1
2
3
0
Enter second tree
5
2
8
7
6
inorder Transversal in First Tree
0 1 2 3
inorder Transversal in 2nd Tree
2 5 7 8
Checking Whether Equal or not
Not Equal
Counting No. of leaf Nodes
First Tree:2 and Second Tree:2
Process returned 0 (0x0)   execution time : 9.698 s
Press any key to continue.
```