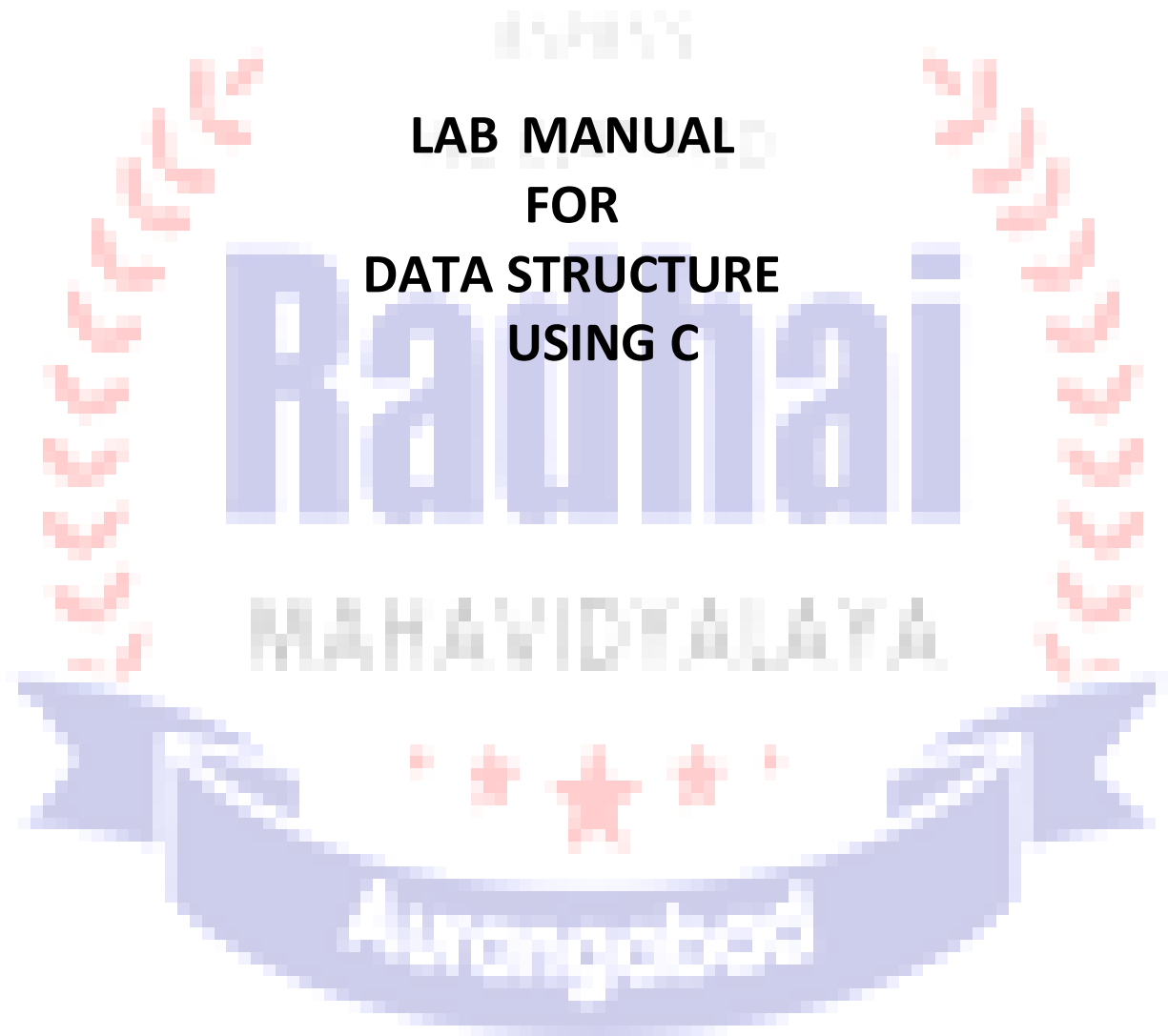


Radhai Mahavidyalaya Aurangabad College of Computer science  
& management science

Approved by Govt. of Maharashtra & Affilate to Dr. Babasaheb Ambedkar university Aurangabad.

Recognized Under Section 2(f) & 12 (B) of the U.G.C Act.

An ISO 9001-2015 Certified institution



## Lab manual for Data Structure using C


1.

Aim: To display fibounacci series up to a range.

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

void main()
{
int a,b,c,n;
clrscr();
printf("\nEnter range:");
scanf("%d",&n);
a=0,b=1,c=0;
printf("%d \t %d",a,b);
c=a+b;
while(c<=n)
{
printf("\t%d",c);
a=b;
b=c;
c=a+b;
}
getch();
}
```

Output:



```
Enter range:13
0      1      1      2      3      5      8      13
```

2.

Aim: To read n numbers and display it.

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

void main()
{
int i,n, a[10];
clrscr();
printf("\nEnter the number of element : \n");
scanf("%d",&n);
```

## Lab manual for Data Structure using C

```
printf("Enter element: \n");
for(i=0;i<n;i++)
{
    printf("a[%d]=",i);
    scanf("%d",&a[i]);
}
printf("\n Display array element: \n");
for(i=0;i<n;i++)
{
    printf("a[%d]=%d\n",i,a[i]);
}
getch();
}
```

### Output:

A screenshot of a terminal window with a black background and white text. The text shows the execution of a C program. It starts with a prompt 'Enter the number of element :', followed by the input '6'. Then it prompts 'Enter element:' and shows six lines of input: 'a[0]=54', 'a[1]=45', 'a[2]=67', 'a[3]=76', 'a[4]=78', and 'a[5]=98'. After a blank line, it prompts 'Display array element:' and shows the same six elements being printed: 'a[0]=54', 'a[1]=45', 'a[2]=67', 'a[3]=76', 'a[4]=78', and 'a[5]=98'. The prompt character '-' is visible at the bottom.

```
Enter the number of element :
6
Enter element:
a[0]=54
a[1]=45
a[2]=67
a[3]=76
a[4]=78
a[5]=98

Display array element:
a[0]=54
a[1]=45
a[2]=67
a[3]=76
a[4]=78
a[5]=98
-
```

3.

Aim: To demonstrate the concept of one dimensional array finding the sum of array elements.


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

```
void main()
```

## Lab manual for Data Structure using C

```
{
int i,n, a[10],s;
clrscr();
printf("Enter the number of element :\n");
scanf("%d",&n);
s=0;
printf("Enter element:\n");
for(i=0;i<n;i++)
{
printf("a[%d]=",i);
scanf("%d",&a[i]);
s=s+a[i];
}
printf("Sum of array element:%d",s);
getch();
}
```

### Output:



```
Enter the number of element :
5
Enter element:
a[0]=1
a[1]=2
a[2]=3
a[3]=4
a[4]=5

Sum of array element:15
```

4.

Aim: To insert an element in an array.

```
#include<stdio.h>
#include<
{
int i,n,pos,num, a[10];
clrscr();
printf("Enter the number of element :\n");
scanf("%d",&n);
printf("Enter element:\n");
for(i=0;i<n;i++)
{
printf("a[%d]=",i);
scanf("%d",&a[i]);
}
```

## Lab manual for Data Structure using C

```
}
printf("\nEnter the pos where the no. is to be inserted :");
scanf("%d",&pos);
printf("\nEnter the the no. is to be inserted :");
scanf("%d",&num);
for(i=n-1;i>=pos;i--)
a[i+1]=a[i];
n=n+1;
a[pos]=num;
printf("\n Display array after insertion:\n");
for(i=0;i<n;i++)
{
printf("a[%d]=%d\n",i,a[i]);
}
getch();
}
```

### Output:

```
Enter the number of element :4

Enter element:
a[0]=10
a[1]=22
a[2]=33
a[3]=44

Enter the pos where the no. is to be inserted :2

Enter the the no. is to be inserted :90

Display array after insertion:
a[0]=10
a[1]=22
a[2]=90
a[3]=33
a[4]=44
```

5.

To delete an element from an array.

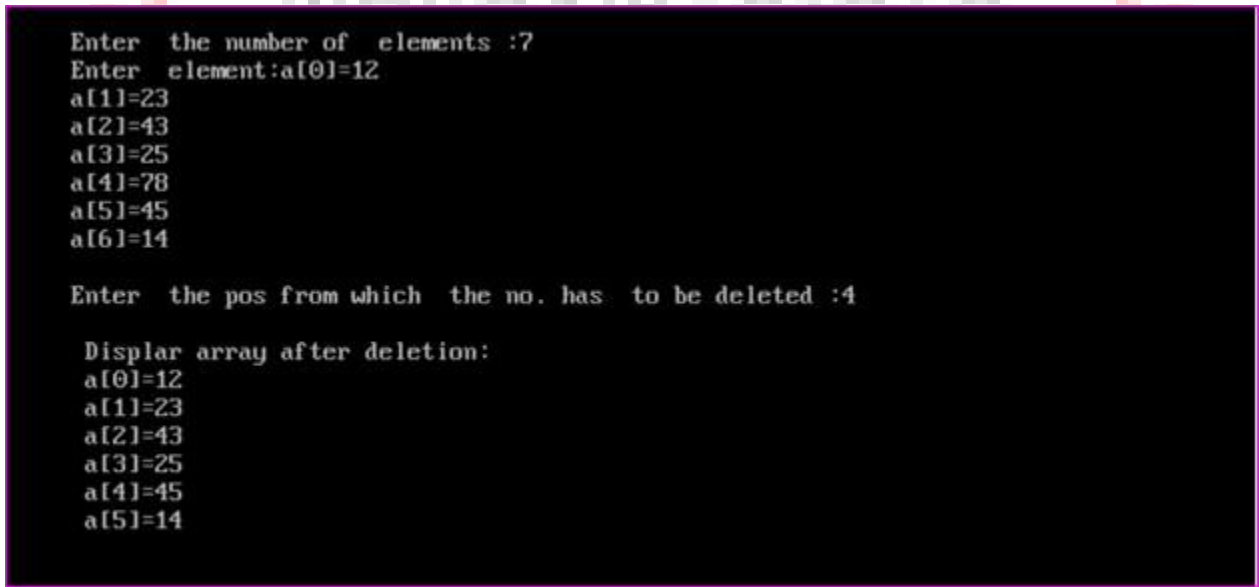
```
#include<stdio.h>
#include<conio.h>
```

```
void main()
{
```

## Lab manual for Data Structure using C

```
int i,n,pos, a[10];
clrscr();
printf("Enter the number of elements :\n");
scanf("%d",&n);
printf("Enter element: \n ");
for(i=0;i<n;i++)
{
printf("a[%d]=",i);
scanf("%d",&a[i]);
}
printf("\nEnter the pos from which the no. has to be deleted :");
scanf("%d",&pos);
for(i=pos;i<n;i++)
a[i]=a[i+1];
n=n-1;
printf("\n Displar array after deletion: \n ");
for(i=0;i<n;i++)
{
printf("\n a[%d]=%d",i,a[i]);
}
getch();
}
```

### Output:

A screenshot of a terminal window with a black background and white text. It shows the execution of a C program that takes an array of 7 elements, deletes the element at index 4, and displays the resulting array of 6 elements. The output is as follows:

```
Enter the number of elements :7
Enter element:a[0]=12
a[1]=23
a[2]=43
a[3]=25
a[4]=78
a[5]=45
a[6]=14

Enter the pos from which the no. has to be deleted :4

Displar array after deletion:
a[0]=12
a[1]=23
a[2]=43
a[3]=25
a[4]=45
a[5]=14
```

## Lab manual for Data Structure using C

Aim: To add two matrix A and B.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i,j,m,n,p,q;
    int a[10][10], b[10][10], c[10][10];
    clrscr();
    printf("\nEnter no of rows and column of matrixA:");
    scanf("%d%d",&m,&n);
    printf("\nEnter no of rows and column of matrixB:");
    scanf("%d%d",&p,&q);
    if(m!=p && n!=q)
    {
        printf("\n Matrix cannot be added.");
        exit(0);
    }
    printf("\n Matrix can be added");
    printf("\n Enter elements of matrix A:");
    for(i=0;i<m;i++)
    for(j=0;j<n;j++)
    scanf("%d",&a[i][j]);
    printf("\n Enter elements of matrix B:");
    for(i=0;i<p;i++)
    for(j=0;j<q;j++)
    scanf("%d",&b[i][j]);
    for(i=0;i<m;i++)
    for(j=0;j<n;j++)
    c[i][j]=a[i][j]+b[i][j];
    printf("\n Display matrix A:\n");
    for(i=0;i<m;i++)
    {
        for(j=0;j<n;j++)
        printf("%d\t",a[i][j]);
        printf("\n");
    }
    printf("\n Display matrix B:\n");
    for(i=0;i<p;i++)
    {
        for(j=0;j<q;j++)
        printf("%d\t",b[i][j]);
        printf("\n");
    }
    printf("\n Display matrix C:\n");
    for(i=0;i<p;i++)
```

## Lab manual for Data Structure using C

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

### Output:

```
Enter no of rows and column of matrixA:3  
3  
Enter no of rows and column of matrixB:3  
3  
  
Matrix can be added  
Enter elements of matrix A:1  
2  
3  
4  
5  
6  
7  
8  
9  
  
Enter elements of matrix B:1  
2  
3  
4  
5  
6  
7  
8  
9
```

Strongnotes



## Lab manual for Data Structure using C

```
Enter elements of matrix B:1
2
3
4
5
6
7
8
9

Display matrix A:
1 2 3
4 5 6
7 8 9

Display matrix B:
1 2 3
4 5 6
7 8 9

Display matrix C:
2 4 6
8 10 12
14 16 18
```

7.(Assignment-1) Write a program to subtract two matrix A and B.

8.

Aim: To multiply two matrix A and B.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,m,n,p,q,k;
int a[10][10], b[10][10], c[10][10];
clrscr();
printf("\nEnter no of rows and column of matrixA:");
scanf("%d%d",&m,&n);
printf("\nEnter no of rows and column of matrixB:");
scanf("%d%d",&p,&q);
printf("\n Enter elements of matrix A:\n");
for(i=0;i<m;i++)
for(j=0;j<n;j++)
scanf("%d",&a[i][j]);
printf("\n Enter elements of matrix B:\n");
for(i=0;i<p;i++)
```

## Lab manual for Data Structure using C

```
for(j=0;j<q;j++)
scanf("%d",&b[i][j]);
if(n==p)
{
for(i=0;i<m;i++)
for(j=0;j<q;j++)
{
c[i][j]=0;
for(k=0;k<n;k++)
c[i][j]=c[i][j]+(a[i][k]*b[k][j]);
}
}
else
{
printf("\n Matrix  cannot be multiplied");
exit(1);
}
printf("\n Display matrix A:\n");
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
printf("%d\t",a[i][j]);
printf("\n");
}
printf("\n Display matrix B:\n");
for(i=0;i<p;i++)
{
for(j=0;j<q;j++)
printf("%d\t",b[i][j]);
printf("\n");
}
printf("\n Display Product:\n");
for(i=0;i<m;i++)
{
for(j=0;j<q;j++)
printf("%d\t",c[i][j]);
printf("\n");
}
getch();
}
```

Output:

## Lab manual for Data Structure using C

```
Enter no of rows and column of matrix A:2
2

Enter no of rows and column of matrix B:2
2

Enter elements of matrix A:
1
2
3
4

Enter elements of matrix B:
1
2
3
4_
```

```
Display matrix A:
1      2
3      4

Display matrix B:
1      2
3      4

Display Product:
7      10
15     22
```

9.

Aim: To Concatenate two string.

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

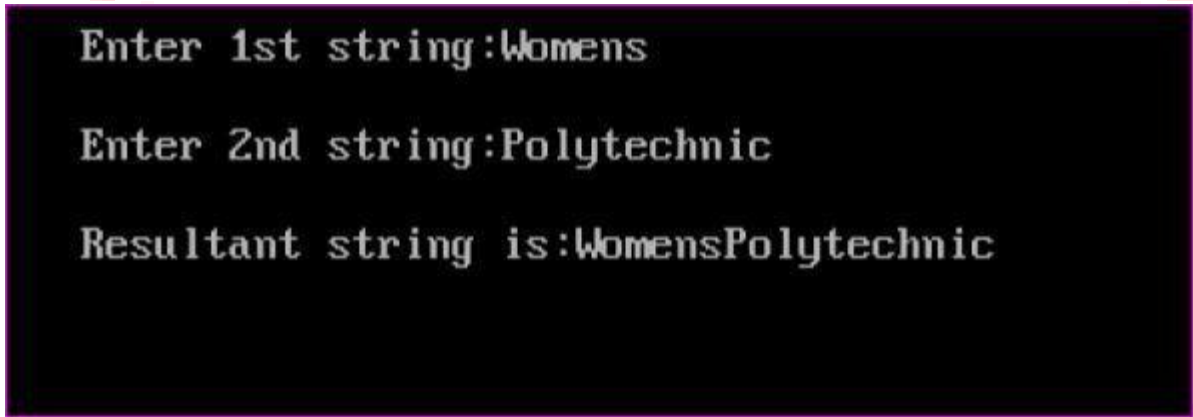
```
void main()
{
char str[20],str1[20],str2[20];
int i,j;
clrscr();
i=j=0;
printf("\n Enter 1st string:");
```

## Lab manual for Data Structure using C

```
scanf("%s",&str);
printf("\n Enter 2nd string:");
scanf("%s",&str1);
while(str[i]!='\0')
{
    str2[i]=str[i];
    i++;
}

while(str1[j]!='\0')
{
    str2[i]=str1[j];
    i++;
    j++;
}
str2[i]='\0';
printf("\n Resultant string is:%s",str2);
getch();
}
```

Output:



```
Enter 1st string:Womens
Enter 2nd string:Polytechnic
Resultant string is:WomensPolytechnic
```

10.

Aim: To copy a string into another string.

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

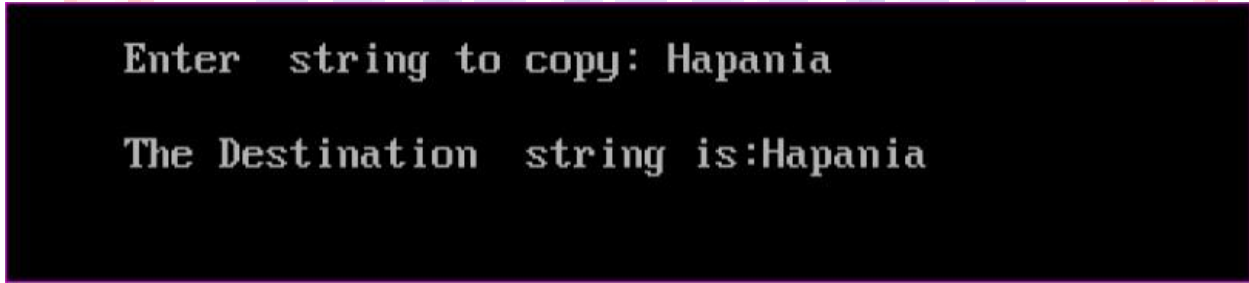
```
void main()
{
```

## Lab manual for Data Structure using C

```
char str[20],str1[20];
int i ;
clrscr();
i=0;
printf("\n Enter string to copy:");
scanf("%s",&str);
while(str[i]!='\0')
{
    str1[i]=str[i];
    i++;
}

str1[i]='\0';
printf("\n The Destination string is:%s",str1);
getch();
}
```

### Output:



```
Enter string to copy: Hapania
The Destination string is:Hapania
```

11.

Aim: Implementation of linked list using array.

```
#include<stdio.h>
#include<conio.h>
#define TRUE 1
#define SIZE 10
struct link
{
    int info;
    int next;
};
struct link node[SIZE];
```

## Lab manual for Data Structure using C

```
int Getnode();
void Createlist();
void Freenode(int);
void Display();
void Insert(int,int);
void Delete(int);
int p, avail=0;
void main()
{
    int ch=1,i,n,x;
    clrscr();
    /*Creation of available list*/
    for(i=0;i<SIZE-1;i++)
        node[i].next=i+1;
    node[SIZE-1].next=-1;
    printf("\n Create a List:");
    Createlist();
    while(ch!=4)
    {
        printf("\n1-DISPLAY");
        printf("\n2-INSERT");
        printf("\n3-DELETE");
        printf("\n4-QUIT");
        printf("\n Enter your choice:");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1 :
                Display();
                break;
            case 2:
                printf("\n Node insertion:after which node:");
                scanf("%d",&n);
                p=n;
                printf("\n Enter the item for insertion:");
                scanf("%d",&x);
                Insert(p,x);
                break;
            case 3:
                printf("\n Enter the node after which the node will be deleted:");
                scanf("%d",&n);
                p=n;
                Delete(p);
                break;
            case 4:
                break;
            default:
```

## Lab manual for Data Structure using C

```
printf("\n Wrong choice!Try again:");
}
}
}
int Getnode()
{
if (avail== -1)
{
printf("\n Overflow:");
exit(0);
}
p=avail;
avail=node[avail].next;
return p;
}
void Freenode(int q)
{
node[q].next=avail;
avail=q;
return;
}
void Createlist()
{
int x;
char c;
p=Getnode();
printf("\n Enter an item to be inserted:");
scanf("%d", &x);
node[p].info=x ;
node[p].next= -1;
while(TRUE)
{
printf("\n Enter the choice(y/n):");
fflush(stdin);
c=getchar();
if(c=='y' || c=='Y')
{
printf("\n Enter an item to be inserted:");
scanf("%d",&x);
Insert(p,x);
node[p].next= -1;
}
else
return;
}
}
void Display()
```

## Lab manual for Data Structure using C

```
{
p=0;
while(node[p].next!=-1)
{
printf("\n%d\t%d\t%d:",p,node[p].info,node[p].next);
p=node[p].next;
}
printf("\n%d\t%d\t%d:",p,node[p].info,node[p].next);
}
void Insert(int r,int x)
{
int q;
if(r==-1)
{
printf("\n void insertion:");
return;
}
q=Getnode();
node[q].info=x;
node[q].next=node[r].next;
node[r].next=q;
return;
}
void Delete(int r)
{
int q;
if(r==-1||node[r].next==-1)
{
printf("\n void deletion:");
return;
}
q=node[r].next;
node[r].next=node[q].next;
Freenode(q);
return;
}
```

Output:



## Lab manual for Data Structure using C

```
Enter an item to be inserted:4
Enter the choice(y/n):y
Enter an item to be inserted:23
Enter the choice(y/n):y
Enter an item to be inserted:87
Enter the choice(y/n):y
Enter an item to be inserted:22
Enter the choice(y/n):y
Enter an item to be inserted:12
Enter the choice(y/n):n
```

```
1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
```

```
Enter your choice:
```

```
Enter your choice:2
Node insertion:after which node:3
Enter the item for insertion:99
```

```
1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
```

```
Enter your choice:1
```

```
0      4      1:
1      23     2:
2      87     3:
3      22     5:
5      99     4:
4      12     -1:
```

```
1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
```

```
Enter your choice:3
```

```
Enter the node after which the node will be deleted:_
```

## Lab manual for Data Structure using C

```
4      12      -1:
1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
Enter your choice:3

Enter the node after which the node will be deleted:1

1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
Enter your choice:1

0      4      1:
1      23     3:
3      22     5:
5      99     4:
4      12     -1:
1-DISPLAY
2-INSERT
3-DELETE
4-QUIT
Enter your choice:4
```

12.

Aim: Implementation of stack using array.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define MAXSTK 100
int top=-1;
int items[MAXSTK];
int Iseempty();
int Isfull();
void Push(int);
int Pop();
void Display();
void main()
{
int x;
char ch='1';
clrscr();
while(ch!='4')
{
printf("\n 1-PUSH");
```

## Lab manual for Data Structure using C

```
printf("\n 2-POP");
printf("\n 3-DISPLAY");
printf("\n 4-QUIT");
printf("\n Enter your choice:");
fflush(stdin);
ch=getchar();
switch(ch)
{
case '1':
printf("\n Enter the element to be pushed:");
scanf("%d",&x);
Push(x);
break;
case '2':
x=Pop();
printf("\n Pop element is %d\n:",x);
break;
case '3':
Display();
break;
case '4':
break;
default:
printf("\n Wrong choice!Try again:");
}
}
}
int Isempy()
{
if(top==-1)
return 1;
else
return 0;
}
int Isfull()
{
if(top==MAXSTK-1)
return 1;
else
return 0;
}
void Push(int x)
{
if(Isfull())
{
printf("\n Stack full");
return;
}
```

## Lab manual for Data Structure using C

```
}
top++;
items[top]=x;
}
int Pop()
{
int x;
if(Isempty())
{
printf("\n Stack empty");
exit(0);
}
x=items[top];
top--;
return x;
}
void Display()
{
int i;
if(Isempty())
{
printf("\n Stack empty");
return;
}
printf("\n Elements in the Stack are :\n");
for(i=top;i>=0;i--)
printf("%d\n",items[i]);
}
```

Output:

## Lab manual for Data Structure using C

```
Enter the element to be pushed:78
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:1
```

```
Enter the element to be pushed:87
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:3
```

```
Elements in the Stack are :
```

```
87
```

```
78
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:_
```

```
87
```

```
78
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:2
```

```
Pop element is 87
```

```
:
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:3
```

```
Elements in the Stack are :
```

```
78
```

```
1-PUSH
```

```
2-POP
```

```
3-DISPLAY
```

```
4-QUIT
```

```
Enter your choice:4_
```

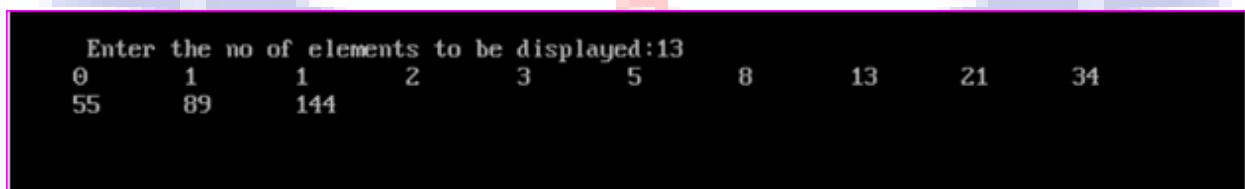
## Lab manual for Data Structure using C

13.

Aim: To Create fibonacci series using recursive function.

```
#include<stdio.h>
#include<conio.h>
int Fibonacci(int);
void main()
{
    int i,n;
    clrscr();
    printf("\n Enter the no of elements to be displayed:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
        printf("%d\t",Fibonacci(i));
    getch();
}
int Fibonacci(int n)
{
    if(n<=0)
        return 0;
    else if (n==1)
        return 1;
    else
        return Fibonacci(n-1)+ Fibonacci(n-2);
}
```

Output:



```
Enter the no of elements to be displayed:13
0      1      1      2      3      5      8      13      21      34
55     89     144
```

14.


Aim: Calculate factorial of a number using recursive function.

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

## Lab manual for Data Structure using C

```
int Factorial(int);
void main()
{
    int i,n;
    clrscr();
    printf("\n Enter the no of elements:");
    scanf("%d",&n);
    printf("Factorial of %d is %d",n,Factorial(n));
    getch();
}
int Factorial(int n)
{
    if(n==0)
        return 1;
    else
        return n*Factorial(n-1);
}
```

### Output:



```
Enter the no of elements:5
Factorial of 5 is 120
```

15.

### Aim: Implementation of queue using array.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define MAXQ 100
int front=0,rear=-1;
int items[MAXQ];
int Isempty();
int Isfull();
void Insert(int);
int Delete();
void Display();
void main()
{
    int x;
```

## Lab manual for Data Structure using C

```
char ch='1';
clrscr();
while(ch!='4')
{
printf("\n 1-INSERT");
printf("\n 2-DELETE");
printf("\n 3-DISPLAY");
printf("\n 4-QUIT");
printf("\n Enter your choice:");
fflush(stdin);
ch=getchar();
switch(ch)
{
case '1':
printf("\n Enter the element to be inserted:");
scanf("%d",&x);
Insert(x);
break;
case '2':
x=Delete();
printf("\n Delete element is %d\n",x);
break;
case '3':
Display();
break;
case '4':
break;
default:
printf("\n Wrong choice!Try again:");
}
getch();
}
int Isempy()
{
if(rear<front)
return 1;
else
return 0;
}
int Isfull()
{
if(rear==MAXQ-1)
return 1;
else
return 0;
}
```



## Lab manual for Data Structure using C

```
void Insert(int x)
{
    if(Isfull())
    {
        printf("\n Queue full");
        return;
    }
    rear++;
    items[rear]=x;
}
int Delete()
{
    int x;
    if(Isempy())
    {
        printf("\n Queue is empty");
        exit(0);
    }
    x=items[front];
    front++;
    return x;
}
void Display()
{
    int i;
    if(Isempy())
    {
        printf("\n Queue is empty");
        return;
    }
    printf("\n Elements in the Queue are :\n");
    for(i=front;i<=rear;i++)
        printf("%d\n",items[i]);
}
```

Output:

## Lab manual for Data Structure using C

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:1

Enter the element to be inserted:30

1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:1

Enter the element to be inserted:40

1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:1

Enter the element to be inserted:50_
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:3

Elements in the Queue are :
30
40
50

1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:2

Delete element is 30
:
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:
```

## Lab manual for Data Structure using C

```
50
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:2

Delete element is 30
:
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:3

Elements in the Queue are :
40
50

1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:4
```

16.

Aim: Implementation of circular queue using array.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define MAXQ 100
int front=-1,rear=-1;
int items[MAXQ];
int Iseempty();
int Isfull();
void Insert(int);
int Delete();
void Display();
void main()
{
int x;
char ch='1';
clrscr();
while(ch!='4')
```

## Lab manual for Data Structure using C

```
{
printf("\n 1-INSERT");
printf("\n 2-DELETE");
printf("\n 3-DISPLAY");
printf("\n 4-QUIT");
printf("\n Enter your choice:");
fflush(stdin);
ch=getchar();
switch(ch)
{
case '1':
printf("\n Enter the nos of element to be inserted:");
scanf("%d",&x);
Insert(x);
break;
case '2':
x=Delete();
printf("\n Deleted element is %d\n",x);
break;
case '3':
Display();
break;
case '4':
break;
default:
printf("\n Wrong choice!Try again:");
}
}
getch();
}
int Isempy()
{
if(front== -1)
return 1;
else
return 0;
}
int Isfull()
{
if(front==(rear+1)%MAXQ)
return 1;
else
return 0;
}
void Insert(int x)
{
if(Isfull())
```

## Lab manual for Data Structure using C

```
{
printf("\n Queue full");
return;
}
if (front== -1)
{
front=0;
rear=0;
}
else
rear=(rear+1)%MAXQ;
items[rear]=x;
}
int Delete()
{
int x;
if(Isempty())
{
printf("\n Queue is empty");
exit(0);
}
x=items[front];
if (front==rear)
{
front=-1;
rear=-1;
}
else
front=(front+1)%MAXQ;
return x;
}
void Display()
{
int i,n;
if(Isempty())
{
printf("\n Queue is empty");
return;
}
printf("\n Elements in the Queue are :\n");
if(front<=rear)
{
for(i=front;i<=rear;i++)
printf("%d\n",items[i]);
}
else
{

```

## Lab manual for Data Structure using C

```
for(i=front;i<=MAXQ-1;i++)  
printf("%d\n",items[i]);  
for(i=0;i<=rear;i++)  
printf("%d\n",items[i]);  
}  
}
```

### Output:

```
1-INSERT  
2-DELETE  
3-DISPLAY  
4-QUIT  
Enter your choice:1  
  
Enter the nos of element to be inserted:20  
  
1-INSERT  
2-DELETE  
3-DISPLAY  
4-QUIT  
Enter your choice:1  
  
Enter the nos of element to be inserted:30  
  
1-INSERT  
2-DELETE  
3-DISPLAY  
4-QUIT  
Enter your choice:1  
  
Enter the nos of element to be inserted:40
```

## Lab manual for Data Structure using C

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:1
```

```
Enter the nos of element to be inserted:50
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:3
```

```
Elements in the Queue are :
20
30
40
50
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:_
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:2
```

```
Deleted element is 20
:
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:3
```

```
Elements in the Queue are :
30
40
50
```

```
1-INSERT
2-DELETE
3-DISPLAY
4-QUIT
Enter your choice:
```

## Lab manual for Data Structure using C

17.

Aim: Implementation of binary search tree using array.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define TRUE 1
#define TREENODES 100
#define FALSE 0
struct tree
{
    int info;
    int used;
};
struct tree node[TREENODES];
void Createtree();
void Insert(int);
void Display();
void Setleft(int,int);
void Setright(int,int);
void main()
{
    int x;
    char ch='1';
    clrscr();
    printf("\n Enter root node value:");
    scanf("%d", &x);
    Createtree(x);
    while(ch!='3')
    {
        printf("\n1-INSERT");
        printf("\n2-DISPLAY");
        printf("\n3-QUIT");
        printf("\n Enter your choice:");
        fflush(stdin);
        ch=getchar();
        switch(ch)
        {
            case '1' :
                printf("\n Enter the element to be inserted:");
                scanf("%d",&x);
                Insert(x);
                break;
            case '2':
                Display();
                break;
```



## Lab manual for Data Structure using C

```
case '3':
break;
default:
printf("\n Wrong choice!Try again:");
}
}
}
void Createtree(int x)
{
int i;
node[0].info=x;
node[0].used=TRUE;
for(i=1;i<TREENODES;i++)
node[i].used=FALSE;
}
void Insert(int x)
{
int p,q;
p=q=0;
while(q<TREENODES && node[q].used && x!=node[p].info)
{
p=q;
if(x<node[p].info)
q=2*p+1;
else
q=2*p+2;
}
if(x==node[p].info)
printf("\n %d is a duplicate number\n",x);
else
if(x<node[p].info)
Setleft(p,x);
else
Setright(p,x);
}
void Setleft(int pos,int x)
{
int q;
q=2*pos+1;
if(q>TREENODES)
printf("\n Array overflow.");
else
if(node[q].used==TRUE)
printf("\n Invalid insertion.");
else
{
node[q].info=x;
```

## Lab manual for Data Structure using C

```
node[q].used=TRUE;
}
}
void Setright(int pos,int x)
{
int q;
q=2*pos+2;
if(q>TREENODES)
printf("\n Array overflow.");
else
if(node[q].used==TRUE)
printf("\n Invalid insertion.\n");
else
{
node[q].info=x;
node[q].used=TRUE;
}
}
void Display()
{
int i;
for(i=0;i<TREENODES;i++)
if(node[i].used==TRUE)
printf("%d ",node[i].info);
printf("\n");
}
```

Output:

## Lab manual for Data Structure using C

```
Enter root node value:60

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:1

Enter the element to be inserted:40

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:1

Enter the element to be inserted:30

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:1

Enter the element to be inserted:70

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:1

Enter the element to be inserted:70

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:1

Enter the element to be inserted:90

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:2
60 40 70 30 90

1-INSERT
2-DISPLAY
3-QUIT
Enter your choice:3_
```

18.

Aim: To Search an element using sequential search.

## Lab manual for Data Structure using C

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

int Sequentialsearch(int[],int,int);
void main()
{
    int x[20],i,n,p,key;
    clrscr();
    printf("\n Enter the no of element:");
    scanf("%d",&n);
    printf("\n Enter %d elements:",n);
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    printf("\n Enter the element to be search:");
    scanf("%d",&key);
    p=Sequentialsearch(x,n,key);
    if(p==-1)
        printf("\n The search is unsuccessful:\n");
    else
        printf("\n %d is found at location %d",key,p);
    getch();
}

int Sequentialsearch(int a[],int n ,int k)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(k==a[i])
            return(i);
    }
    return(-1);
}
```

Output:

## Lab manual for Data Structure using C

```
Enter the no of element:4
Enter 4 elements:34
89
90
24

Enter the element to be search:90

90 is found at location 2
```

19.

Aim: To Search an element using binary search.

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

int Binarysearch(int[],int,int);
void main()
{
    int x[20],i,n,p,key;
    clrscr();
    printf("\n Enter the no of element:");
    scanf("%d",&n);
    printf("\n Enter %d elements in assending order:",n);
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    printf("\n Enter the element to be search:");
    scanf("%d",&key);
    p=Binarysearch(x,n,key);
    if(p== -1)
        printf("\n The searchis unsuccessful:\n");
    else
        printf("\n %d is found at location %d",key,p);
}
```

## Lab manual for Data Structure using C

```
getch();
}

int Binarysearch(int a[],int n ,int k)
{
    int lo,hi,mid;
    lo=0;
    hi=n-1;
    while(lo<=hi)
    {
        mid=(lo+hi)/2;
        if(k==a[mid])
            return(mid);
        if(k<a[mid])
            hi=mid-1;
        else
            lo=mid+1;
    }
    return(-1);
}
```

Output:

```
Enter the no of element:6

Enter 6 elements in assending order:34
56
67
84
89
90

Enter the element to be search:89

89 is found at location 4
```

## Lab manual for Data Structure using C

20.

Aim: Arrange the list of numbers in ascending order using Bubble Sort.

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

void Bubblesort(int[],int);
void main()
{
    int x[20],i,n;
    clrscr();
    printf("\n Enter the no of element to be sorted:");
    scanf("%d",&n);
    printf("\n Enter %d elements:",n);
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    Bubblesort(x,n);
    printf("\n The sorted array is:\n");
    for(i=0;i<n;i++)
        printf("%4d",x[i]);
    getch();
}

void Bubblesort(int a[],int n)
{
    int temp,pass,i;
    for(pass=0;pass<n-1;pass++)
    {
        for(i=0;i<n-pass-1;i++)
        {
            if(a[i]>a[i+1])
            {
                temp=a[i];
                a[i]=a[i+1];
                a[i+1]=temp;
            }
        }
    }
}
```

Output:

## Lab manual for Data Structure using C

```
Enter the no of element to be sorted:6
```

```
Enter 6 elements:12
```

```
90
```

```
76
```

```
45
```

```
13
```

```
7
```

```
The sorted array is:
```

```
7 12 13 45 76 90
```

21.

Aim: Arrange the list of numbers in ascending order using Insertion Sort.

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

```
#include<conio.h>
```

```
void Insertionsort(int[],int);
```

```
void main()
```

```
{
```

```
int x[20],i,n;
```

```
clrscr();
```

```
printf("\n Enter the no of element to be sorted:");
```

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

```
printf("\n Enter %d elements:",n);
```

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

```
scanf("%d",&x[i]);
```

```
Insertionsort(x,n);
```

```
printf("\n The sorted array is:\n");
```

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

```
printf("%4d",x[i]);
```

```
getch();
```

```
}
```

```
void Insertionsort(int a[],int n)
```



## Lab manual for Data Structure using C

```
{  
int i,j,key;  
for(j=1;j<n;j++)  
{  
key=a[j];  
i=j-1;  
while((i>-1)&&(a[i]>key))  
{  
a[i+1]=a[i];  
i=i-1;  
}  
a[i+1]=key;  
}  
}
```

Enter the no of element to be sorted:6

Enter 6 elements:54

12

90

35

81

16

The sorted array is:

12 16 35 54 81 90

22.

Aim: Arrange the list of numbers in ascending order using Selection Sort.

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

```
#include<conio.h>
```

```
void Selectionsort(int[],int);
```

```
void main()
```

## Lab manual for Data Structure using C

```
{
int x[20],i,n;
clrscr();
printf("\n Enter the no of element to be sorted:");
scanf("%d",&n);
printf("\n Enter %d elements:",n);
for(i=0;i<n;i++)
scanf("%d",&x[i]);
Selectionsort(x,n);
printf("\n The sorted array is:\n");
for(i=0;i<n;i++)
printf("%4d",x[i]);
getch();
}
void Selectionsort(int a[],int n)
{
int i,j,pos,large;
for(i=n-1;i>0;i--)
{
large=a[0];
pos=0;
for(j=1;j<=i;j++)
{
if (a[j]>large)
{
large=a[j];
pos=j;
}
}
a[pos]=a[i];
a[i]=large;
}
}
```

Output:

## Lab manual for Data Structure using C

```
Enter the no of element to be sorted:7

Enter 7 elements:45
12
32
10
34
67
41

The sorted array is:
10 12 32 34 41 67 45
```

23.

Aim: Arrange the list of numbers in ascending order using Merge Sort.

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

void Mergesort(int[],int,int);
void Merge(int[],int,int,int);
void main()
{
    int x[20],i,n;
    clrscr();
    printf("\n Enter the no of element to be sorted:");
    scanf("%d",&n);
    printf("\n Enter %d elements:",n);
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    Mergesort(x,0,n-1);
    printf("\n The sorted array is:\n");
    for(i=0;i<n;i++)
        printf("%4d",x[i]);
    getch();
}
```

## Lab manual for Data Structure using C

```
void Mergesort(int a[],int p,int r)
{
    int q;
    if(p<r)
    {
        q=(p+r)/2;
        Mergesort(a,p,q);
        Mergesort(a,q+1,r);
        Merge(a,p,q,r);
    }
}

void Merge(int a[], int p, int q,int r)
{
    int b[20],l1,r1,i;
    l1=p;
    r1=q+1;
    i=p;
    while((l1<=q)&&(r1<=r))
    {
        if(a[l1]<a[r1])
        {
            b[i]=a[l1];
            l1=l1+1;
            i=i+1;
        }
        else
        {
            b[i]=a[r1];
            r1=r1+1;
            i=i+1;
        }
    }
    while(l1<=q)
    {
        b[i]=a[l1];
        l1=l1+1;
        i=i+1;
    }
    while(r1<=r)
    {
        b[i]=a[r1];
        r1=r1+1;
        i=i+1;
    }
    for(i=p;i<=r;i++)
    a[i]=b[i];
}
```

## Lab manual for Data Structure using C

### Output:

```
Enter the no of element to be sorted:8

Enter 8 elements:12
10
34
26
78
51
36
79

The sorted array is:
10 12 26 34 36 51 78 79_
```

24.

Aim: Arrange the list of numbers in ascending order using Quick Sort.

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

void Quicksort(int[],int,int);
int partition(int[],int,int);
void main()
{
    int x[20],i,n;
    clrscr();
    printf("\n Enter the no of element to be sorted:");
    scanf("%d",&n);
    printf("\n Enter %d elements:",n);
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    Quicksort(x,0,n-1);
}
```

## Lab manual for Data Structure using C

```
printf("\n The sorted array is:\n");
for(i=0;i<n;i++)
printf("%4d",x[i]);
getch();
}
void Quicksort(int a[],int p,int r)
{
int q;
if(p<r)
{
q=Partition(a,p,r);
Quicksort(a,p,q);
Quicksort(a,q+1,r);
}
}
int Partition(int a[], int p,int r)
{
int k,i,j,temp;
k=a[p];
i=p-1;
j=r+1;
while(1)
{
do
{
j=j-1;
}while(a[j]>k);
do
{
i=i+1;
}while(a[i]<k);
if(i<j)
{
temp=a[i];
a[i]=a[j];
a[j]=temp;
}
else
return(j);
}
}
```

Output:

## Lab manual for Data Structure using C

```
Enter the no of element to be sorted:9
```

```
Enter 9 elements:23
```

```
12
```

```
41
```

```
30
```

```
40
```

```
90
```

```
60
```

```
49
```

```
89
```

```
The sorted array is:
```

```
12 23 30 40 41 49 60 89 90
```

25.

Aim: Arrange the list of numbers in ascending order using Radix Sort.

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

```
#include<conio.h>
```

```
void Radixsort(int[],int);
```

```
void main()
```

```
{
```

```
int x[20],i,n;
```

```
clrscr();
```

```
printf("\n Enter the no of element to be sorted:");
```

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

```
printf("\n Enter %d elements:",n);
```

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

```
scanf("%d",&x[i]);
```

```
Radixsort(x,n);
```

```
printf("\n The sorted array is:\n");
```

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

```
printf("%4d",x[i]);
```

```
getch();
```

```
}
```

```
void Radixsort(int a[],int n)
```

## Lab manual for Data Structure using C

```
{
int bucket[10][10],buck[10];
int i,j,k,l,num,div,large,pass;
div=1;
num=0;
large=a[0];
for(i=0;i<n;i++)
{
if(a[i]>large)
large=a[i];
}
while(large>0)
{
num=num+1;
large=large/10;
}
for(pass=0;pass<num;pass++)
{
for(k=0;k<10;k++)
buck[k]=0;
for(i=0;i<n;i++)
{
l=(a[i]/div)%10;
bucket[l][buck[l]++]=a[i];
}
i=0;
for(k=0;k<10;k++)
{
for(j=0;j<buck[k];j++)
a[i++]=bucket[k][j];
}
div=div*10;
}
}
```

Output:



## Lab manual for Data Structure using C

```
Enter the no of element to be sorted:6
```

```
Enter 6 elements:102
```

```
401
```

```
34
```

```
95
```

```
305
```

```
289
```

```
The sorted array is:
```

```
34 95 102 289 305 401
```

26.

Aim: Arrange the list of numbers in ascending order using Heap Sort.

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

```
void Heapsort(int[],int);
int Parent(int);
int Left(int);
int Right(int);
void Heapify(int[],int,int);
void Buildheap(int[],int);
void main()
{
int x[20],i,n;
clrscr();
printf("\n Enter the no of element to be sorted:");
scanf("%d",&n);
printf("\n Enter %d elements:",n);
```

## Lab manual for Data Structure using C

```
for(i=0;i<n;i++)
scanf("%d",&x[i]);
Heapsort(x,n);
printf("\n The sorted array is:\n");
for(i=0;i<n;i++)
printf("%4d",x[i]);
getch();
}
int Parent(int i)
{
return(i/2);
}
int Left(int i)
{
return(2*i+1);
}
int Right(int i)
{
return(2*i+2);
}
void Heapify(int a[],int i,int n)
{
int l,r,large,temp ;
l=Left(i);
r=Right(i);
if((l<=n-1)&&(a[l]>a[i]))
large=l;
else
large=i;
if((r<=n-1)&&(a[r]>a[large]))
large=r;
if(large!=i)
{
temp=a[i];
a[i]=a[large];
a[large]=temp;
Heapify(a,large,n);
}
}
void Buildheap(int a[],int n)
{
int i;
for (i=(n-1)/2;i>=0;i--)
Heapify(a,i,n);
}
void Heapsort(int a[],int n)
{
}
```

## Lab manual for Data Structure using C

```
int i,m,temp;
Buildheap(a,n);
m=n;
for(i=n-1;i>=1;i--)
{
temp=a[0];
a[0]=a[i];
a[i]=temp;
m=m-1;
Heapify(a,0,m);
}
}
```

### Output:

```
Enter the no of element to be sorted:9
Enter 9 elements:23
67
45
89
70
90
34
12
36

The sorted array is:
12 23 34 36 45 67 70 89 90_
```