Practical No.1

A) Write a program to store the elements in one dimensional array and perform the operations like searching, sorting and reversing the elements.

1. Program for reversing the elements of array:

Code:-

```
include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int arr[50],size,i,j,temp;
cout<<"Enter array size:";
cin>>size;
cout<<"Enter array element:";
for(i=0;i<size;i++)
{
cin>>arr[i];
}
j=i-1;
i=0;
while(i<j)
{
temp=arr[i];
arr[i]=arr[j];
arr[j]=temp;
i++;
j--;
}
cout<<"Now the reverse of the array is:";
for(i=0;i<size;i++)
{
cout<<arr[i];
}
getch();
}
```

Output:-

```
Enter array size:5
Enter array element:1 2 3 4 5
Now the reverse of the array is:543
```

2.Program for linear search in array:

Code:-

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int arr[10],num,i,n,c=0,pos;
cout<<"Enter array size:";
cin>>n;
cout<<"Enter array element:";
for(i=0;i<n;i++)
{
cin>>arr[i];
}
cout<<"Enter the number to be search:";
cin>>num;
for(i=0;i<n;i++)
{
if(arr[i]==num)
{
c=1;
pos=i+1;
break;
}
}
if(c==0)
{
cout<<"Number not found";
}
else
{
cout<<num<<" found at position;"<<pos;
}
getch();
}
```

Output:-

```
Enter array size:5
Enter array element:4 5 6 7 8
Enter the number to be search:6
6 found at position:3
```

3.Program for sorting array in ascending order:

Code:-

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int a[10],temp,j,i;
cout<<"Enter any 10 number in
array:";
for(i=0;i<10;i++)
{
cin>>a[i];
}
cout<<"Data before sorting:\n";
for(j=0;j<10;j++)
{
cout<<a[j];
}
for(i=0;i<=10;i++)
{
for(j=0;j<=10;j++)
{
if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
cout<<"\nData after sorting:\n";
for(j=0;j<10;j++)
{
cout<<a[j];
}
getch();
}
```

Output:-

```
Enter any 10 number in array:7 4 1 8
Data before sorting:
7418529630
Data after sorting:
0123456789
```

4.Program for sorting array in descending order:

Code:-

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int a[10],temp,j,i;
cout<<"Enter any 10 number in
array:";
for(i=0;i<10;i++)
{
cin>>a[i];
}
cout<<"Data before sorting:\n";
for(j=0;j<10;j++)
{
cout<<a[j];
}
for(i=0;i<=10;i++)
{
for(j=0;j<=10;j++)
{
if(a[j]<a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
cout<<"\nData after sorting:\n";
for(j=0;j<10;j++)
{
cout<<a[j];
}
getch();
}
```

Output:-

```
Enter any 10 number in array:7 4 1 0 8 5 2 9 6 3
Data before sorting:
7410852963
Data after sorting:
9876543210
```

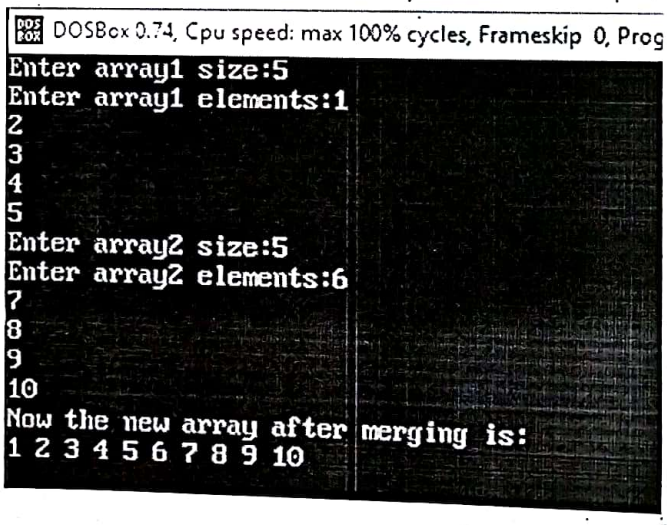
Practical no.01(b)

Aim: Read the two arrays from the user and merge them and display the elements in sorted order.

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int
arr1[50],arr2[50],size1,size2,size,i,j,k,
merge[100];
cout<<"Enter array1 size:";
cin>>size1;
cout<<"Enter array1 elements:";
for(i=0;i<size1;i++)
{
cin>>arr1[i];
}
cout<<"Enter array2 size:";
cin>>size2;
cout<<"Enter array2 elements:";
for(i=0;i<size2;i++)
{
cin>>arr2[i];
}
for(i=0;i<size1;i++)
{
merge[i]=arr1[i];
}
size=size1+size2;
for(i=0,k=size1;k<size &&
i<size2;i++,k++)
{
merge[k]=arr2[i];
}
cout<<"Now the new array after
merging is:\n";
for(i=0;i<size;i++)
{
cout<<merge[i]<<" ";
}
getch();
}
```

Output:



PRACTICAL NO.01(C)

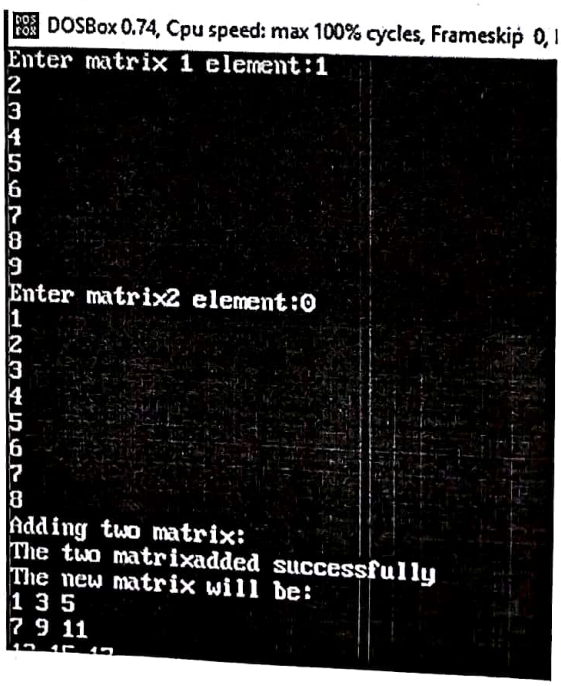
Aim: Write a program to perform addition , multiplication and transpose operation.

i)Addition of two matrices:

Code|:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int
i,j,mat1[3][3],mat2[3][3],mat3[3][3];
cout<<"Enter matrix 1 element:";
for(i=0:i<3:i++) {
for(j=0;j<3;j++)
{
cin>>mat1[i][j];
}
}
cout<<"Enter matrix2 element:";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cin>>mat2[i][j];
}}
cout<<"Adding two matrix:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
mat3[i][j]=mat1[i][j]+mat2[i][j];
}
}
cout<<"The two matrixadded
successfully\n";
cout<<"The new matrix will be:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cout<<mat3[i][j]<<" ";
}
cout<<"\n";
}
getch():
}
```

Output:

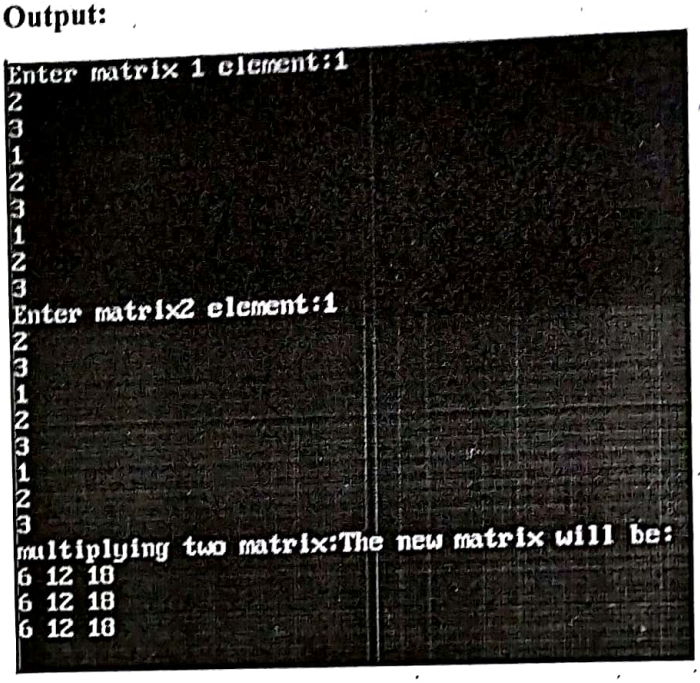


ii) Multiplication of two matrices

code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int
i,j,mat1[3][3],mat2[3][3],mat3[3][3],k,
sum=0;
cout<<"Enter matrix 1
element:";
for(i=0;i<3;i++) {
for(j=0;j<3;j++)
{
cin>>mat1[i][j];
}
}
cout<<"Enter matrix2
element:";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cin>>mat2[i][j];
}
}
cout<<"multiplying two
matrix:";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
sum=0;
for(k=0;k<3;k++)
{
sum=+mat1[i][k]*mat2[k][j]
;
}
mat3[i][j]=sum;
}
}
cout<<"The new matrix will
be:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cout<<mat3[i][j]<<" ";
}
cout<<"\n";
}
getch();
}
```

Output:



iii)program to perform transpose of matrix:

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int arr[3][3],i,j,arrt[3][3];
cout<<"Enter 3*3 array elemnts:";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cin>>arr[i][j];
}
}
cout<<"Transposing an array:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
arrt[i][j]=arr[j][i];
}
}
cout<<"Transpose of matrix is:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cout<<arrt[i][j]<<"\t";
}
cout<<"\n";
}
getch();
}
```

Output:

```
Enter 3*3 array elemnts:1 2 3 4 5 6 7 8 9
Transposing an array:
Transpose of matrix is:
1       4       7
2       5       8
3       6       9
```

Practical no.02 (A)

Aim: Write a program to create a single linked list and display the node elements in the reverse order.

Code:

```
#include<iostream.h>
#include<process.h>
#include<conio.h>
struct node
{
int info;
node *next;
}
*start,*newptr,*save,*ptr;
node *create_new_node(int);
void insert_at_beg(node*);
void display(node*);
void main()
{
clrscr();
start=NULL;
int inf;
char ch='y';
while(ch=='y'||ch=='Y')
{
clrscr();
cout<<"Enter information for the new node:";
cin>>inf;
cout<<"\nCreating new node....!!Press any key to comtinue";
getch();
newptr=create_new_node(inf);
if(newptr!=NULL)
{
cout<<"\nNew node created successfully!!!\n";
cout<<"Press any ket to continue...";
getch();
}
else
{
cout<<"\nsorry..!!Cannot created new node...!!";
cout<<"\nPress any key to exit..";
getch();
exit(1);
}
cout<<"\nNow inserting new node at the beginning..";
cout<<"Press any key to continue..";
getch();
insert_at_beg(newptr);
cout<<"\nNew node successfully inserted at the beginning..";
cout<<"Now the list is:\n";
display(start);
cout<<"\nWant to enter more node?(y/n)..";
cin>>ch;
}
```

```
getch();
}
node *create_new_node(int n)
{
ptr=new node;
ptr->info=n;
ptr->next=NULL;
return ptr;
}
void insert_at_beg(node *np)
{
if(start==NULL)
{
start=np;
}
else
{
save=start;
start=np;
np->next=save;
}
}
void display(node *np)
{
while(np!=NULL)
{
cout<<np->info<<"->";
np=np->next;
}
cout<<"!!\n";
}
```

Output:

```
Enter information for the new node:7

Creating new node....!!Press any key to continue
New node created successfully!!!
Press any ket to continue...
Now inserting new node at the beginning..Press any key to continue..
New node successfully inserted at the beginning..Now the list is:
7->6->3->!!

Want to enter more node?(y/n)..n
```

Practical no.02(b)

Aim: Write a program to create double linked list and sort the elements in the linked list.

Code:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
struct node
{
node *next,*prev;
int data;
}
*head=NULL,*tail=NULL,*p=NULL,
*r=NULL,*np=NULL;
void create(int x)
{
int c;
np=new node;
np->data=x;
np->next=NULL;
np->prev=NULL;
if(c==0)
{
tail=np;
head=np;
p=head;
p->next=NULL;
c++;
}
else
{
p=head;
r=p;
if(np->data<p->data)
{
np->next=p;
p->prev=np;
np->prev=NULL;
head=np;
p=head;
do
{
p=p->next;
}
while(p->next!=NULL);
tail=p;
}
else if(np->data>p->data)
{
while(p!=NULL&&np->data>p->data)
{
r=p;
p=p->next;
if(p==NULL)
{
r->next=np;
```

```
np->prev=r;
np->next=NULL;
tail=np;
break;
}
else if(np->data<p->data)
{
r->next=np;
np->prev=r;
np->next=p;
p->prev=np;
if(p->next!=NULL)
{
do
{
p=p->next;
}
while(p->next!=NULL);
tail=p;
}
break;
}}}}}
void traverse_tail()
{
node *t=tail;
while(t!=NULL)
{
cout<<t->data<<"\t";
t=t->prev;
}
cout<<endl;
}
void traverse_head()
{
node *t=head;
while(t!=NULL)
{
cout<<t->data<<"\t";
t=t->next;
}
cout<<endl;
}
int main()
{
clrscr();
int i=0,n,x,ch;
cout<<"Enter the no. of nodes:\n";
cin>>n;
while(i<n)
{
cout<<"\nEnter value of node:\n";
Cin>>x;
Create(x);
I++;
}
Court<<"\traversing doubly linked
head first:"
Traverse head ();
```

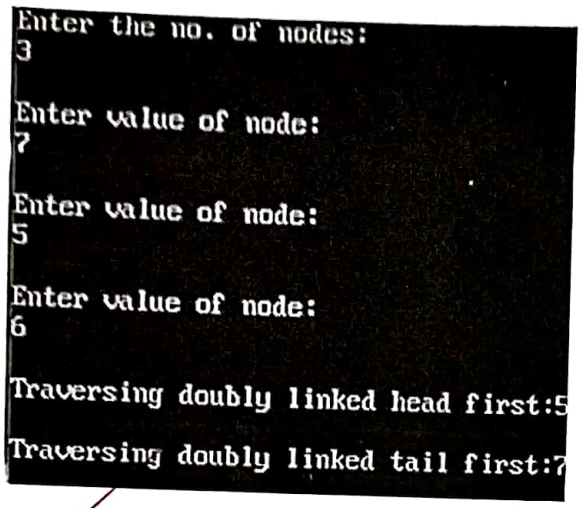
```
Court<<"\traversing doubly linked tail
first:"

Traverse tail ();

Ltch (),

}
```

Output:



Aatif
11/10/18

Practical no. 03(a)

Aim: Write a program to implement the concept of stack with push, pop, display and exit operations.

Code:

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
class stack
{
Int stk [5];
Int top;
Public:
Stack ()
{
Top=-1;
}
Void push (int x)
{
If (top>4)
{
Court<<"Stack over flow";
Return;
}
Stk [++top] =x;
Court<<"Inserted"<<x;
}
Void pop ()
{
If (top<0)
{
Court<<"Stack under flow";
Return;
}
Court<<"deleted"<<stk [top--];
}
Void display ()
{
If (top<0)
{
Court<<"Stack empty";
Return;
}
For (int I=top; I>=0; i--)
Court<<stk [I] <<"";
}
};
Int main ()
{
Closer ();
Int chi;
Stack stk;
While (1)
{
Court<<"\n1.push 2.pop 3.display
4.exit\enter your choice:"
Cin>>chi;
Switch (chi)
{
Case 1:
Court<<"Enter an element:"

Cin>>chi;

St. Push (chi);

Break;

Case 2:

St. Pop ();

Break;

Case 3:

St. Display ();

Break;

Case 4:

Exit (0);

}

Return (0);

}
```

Output:

```
1.push 2.pop 3.display 4.exit
Enter your choice:1
Enter an element:3
Inserted3
1.push 2.pop 3.display 4.exit
Enter your choice:2
deleted3
1.push 2.pop 3.display 4.exit
Enter your choice:3
Stack empty
1.push 2.pop 3.display 4.exit
Enter your choice:4
```

Practical no.03 (b)

Aim: Write a program to implement tower of Hanoi problem

Code:

```
#include<iostream.h>

#include<conio.h>

void towers(int,char,char,char);

int main()

{

int num;

clrscr();

cout<<"Enter the number of disks:";

cin>>num;

cout<<"The sequence of moves
involved in  the tower of Hanoi are:\n";

towers(num,'A','C','B');

return 0;

}

Void towers (int

 {

If (numb==1)

{

court<<"Move disk 1 from peg:
"<<from peg<<" to peg:
"<<tope<<"\n";

Return;

}

Towers (num-1, frompeg, auxpeg,
topeg);

court<<"\move disk: "<<numb<<"
from peg: "<<from peg<<" to peg:
"<<tope<<"\n":

Return;

}

Towers (num-1, frompeg, auxpeg,
topeg);

court<<"\move disk: "<<numb<<"
from peg: "<<from peg<<" to peg:
"<<tope<<"\n";

Towers (num-1, auxpeg, topeg,
frompeg);

Getch ();

}
```

Output:

```
Enter the number of disks:3
The sequence of moves involved in  the towe
Move disk 1 from peg: A to peg: C

Move disk: 2 frompeg: A to peg: B
Move disk 1 from peg: C to peg: B

Move disk: 3 frompeg: A to peg: C
Move disk 1 from peg: B to peg: A

Move disk: 2 frompeg: B to peg: C
Move disk 1 from peg: A to peg: C
```

Aatul
11/10/18

Practical no.04 (a)

Aim: Write a program to implement the concept of queue with insert, delete, and display and exit operations.

Code:

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
#include<stdlib.h>
Class queue
{
Int queue1 [5];
int rear,front,i;
public:
queue()
{
rear=-1;
front -1;
}
void insert(int x)
{
if(rear>4)
{
cout<<"Queue overflow:";
front=rear=-1;
return:
}
queue1[++rear]=x;
cout<<"Inserted:"<<x;
}
void delet()
{
if(front==rear)
{
cout<<"Queue underflow:";
return;
}
cout<<"Deleted:"<<queue1[++front];
}
void display()
{
if(rear==front)
{
cout<<"Queue empty:";
return;
}
for(i=front+1;i<=rear;i++)
cout<<queue1[i]<<" ";
}
};
int main()
{
clrscr();
int ch;
queue qu;
while(1)
{
cout<<"\n1.insert 2.delete 3.display
4.exit\n Enter your choice:";
cin>>ch;
switch(ch)
{
case 1:
cout<<"Enter a element:";
cin>>ch;
qu.insert(ch);
break;
case 2:
qu.delet();
break;
case 3:
qu.display();
break;
case 4:
exit(0);
}
}
return 0;
}
```

Output:

```
1.insert 2.delete 3.display 4.exit
 Enter your choice:1
Enter a element:5
Inserted:5
1.insert 2.delete 3.display 4.exit
 Enter your choice:2
Deleted:5
1.insert 2.delete 3.display 4.exit
 Enter your choice:3
Queue empty:
1.insert 2.delete 3.display 4.exit
 Enter your choice:4
```

Practical no.04(b)

Aim: Write a program to implement the concept of circular queue.

Code:

```
#include<iostream.h>
#include<conio.h>
class cqueue
{
private:
int *arr;
int front,rear;
int max;
public:
cqueue(int maxsize=10);
void addq(int item);
int delq();
void display();
};
cqueue::cqueue(int maxsize)
{
max=maxsize;
arr=new int[max];
front=rear=-1;
for(int i=0;i<max;i++)

arr[i]=0;
}
void cqueue::addq(int item)
{
if((rear+1)%max==front)
{
cout<<"\nQueue is full";
return;
}
rear=(rear+1)%max;
arr[rear]=item;
if(front==-1)
front=0;
}
int cqueue::delq()

{
int data;
if(front==-1)
{
cout<<"\nQueue is empty";
return NULL;
}
data=arr[front];
arr[front]=0;
if(front==rear)
{
front=-1;
rear=-1;
}
else
front=(front+1)%max;
return data;
}
void cqueue::display()
{
cout<<endl;
for(int i=0;i<max;i++)
cout<<arr[i]<<"\t";
cout<<endl;
}
void main()
{
clrscr();

cqueue a(10);

a.addq(14);

a.addq(22);

a.addq(13);

a.addq(-6);

a.addq(25);

cout<<"\nElement in the circular
queue:";

a.display();

int i=a.delq();

cout<<"Item deleted"<<i<<"\n";

i=a.delq();

cout<<"Item deleted"<<i<<"\n";

cout<<"\nElements in the circular
queue after deletion:";

a.display();

a.addq(21);
```

```
a.addq(17);
a.addq(18);
a.addq(9);
a.addq(20);
cout<<"\nElements in the circular queue after addition:";
a.display();
a.addq(32);
cout<<"\nElements in the circular queue after addition:";
a.display();
getch();
}
```

Output:

```
Element in the circular queue:
14      22      13      -6      25

Item deleted14
Item deleted22

Elements in the circular queue after de
0       0       13      -6      25


Elemnts in the circular queue after add
0       0       13      -6      25


Elements in the circular queue after ad
32      0       13      -6      25
```

Practical no.04(c)

Aim: write a program to implement the concept of Dqueue.

Code:

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
class node
{
public:
int data;
class node*next;
class node*prev;
};
class dqueue:public node
{
node *head,*tail;
int top1,top2;
public:
dqueue()
{
top1=0;
top2=0;
head=NULL;
tail=NULL;
}
void push(int x)
{
node *temp;
int ch;
if(top1+top2>=5)
{
cout<<"dqueue overflow";
return;
}
if(top1+top2==0)
{
head=new node;
head->data=x;
head->next=NULL;
head->prev=NULL;
tail=head;
top1++;
}
else
{
```

```
cout<<"ADD element 1.First
2.Last\nEnter your choice:";
cin>>ch;
if(ch==1)
{
top1++;
temp=new node;
temp->data=x;
temp->next=head;
temp->prev=NULL;
head=temp;
}
else
{
top2++;
temp=new node;
temp->data=x;
temp->next=NULL;
temp->prev=tail;
tail->next=temp;
tail=temp;
}
}
}
void pop()
{
int ch;
cout<<"Delete 1.First node 2.Last
node\nEnter your choice:";
cin>>ch;
if(top1+top2<=0)
{
cout<<"\nDqueue under flow";
return;
}
if(ch==1)
{
head=head->next;
head->prev=NULL;
top1--;
}
else
{
top2--;
tail=tail->prev;
tail->next=NULL;
}
}
void display()
{
int ch;
node *temp;
cout<<"\nDisplay from 1.Starting
2.Ending \nEnter your choice:";
cin>>ch;
if(top1+top2<=0)
{
cout<<"under flow";
return;
}
if(ch==1)
{
temp=head;
while(temp!=NULL)
{
cout<<temp->data<<"";
temp=temp->next;
}
}
else
{
temp=tail;
while(temp!=NULL)
{
cout<<temp->data<<"";
temp=temp->prev;
}
}
}
};
main()
{
clrscr();
dqueue d1;
int ch;
while(1)
{
cout<<"\n1.Insert 2.Delete 3.Display
4.Exit\nEnter your choice:";
cin>>ch;
switch(ch)
{
case 1:
cout<<"Enter element:";
cin>>ch;
d1.push(ch);
break;
case 2:
d1.pop();
break;
```

```
case 3:
d1.display();
break:
case 4:
exit(1);
}
}
}
```

Output:

```
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:1
Enter element:5
ADD element 1.First 2.Last
Enter your choice:1

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3

Display from 1.Starting 2.Ending
Enter your choice:1
54
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:2
Delete 1.First node 2.Last node
Enter your choice:1

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3

Display from 1.Starting 2.Ending
Enter your choice:1
4
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:4
```

Aadi
11/10/18

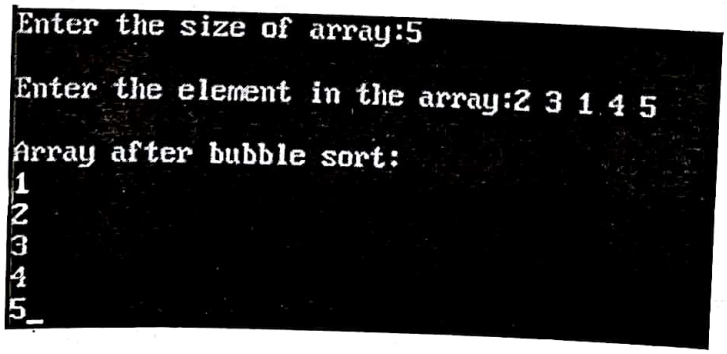
Practical no.05(a)

Aim: Write a program to implement bubble sort.

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int a[50],n,i,j,temp;
cout<<"Enter the size of array:";
cin>>n;
cout<<"\nEnter the element in the array:";
for(i=0;i<n;++i)
cin>>a[i];
for(i=1;i<n;++i)
{
for(j=0;j<(n-1);++j)
if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
cout<<"\nArray after bubble sort:";
for(i=0;i<n;++i)
cout<<"\n"<<a[i];
getch();
}
```

Output:



Practical no.05 (b)

Aim: Write a program to implement selection sort.

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int size,arr[50],i,j,temp;
cout<<"Enter size of an array:";
cin>>size;
cout<<"\nEnter the elements of an
array:";
for(i=0;i<size;i++)
{
cin>>arr[i];
}
cout<<"\nSorting an array using
selection sort:\n";
for(i=0;i<size;i++)
{
for(j=i+1;j<size;j++)
{
if(arr[i]>arr[j])
{
temp=arr[i];
arr[i]=arr[j];
arr[j]=temp;
}}}
cout<<"\nArray after selection sort:\n";
for(i=0;i<size;i++)
{
cout<<arr[i]<<"\t";
}
getch();
}
```

Output:

```
Enter size of an array:5

Enter the elements of an array:2 8

Sorting an array using selection so

Array after selection sort:
1       2       5       8       9
```

11/10/18

Practical no.05(c)

Aim: Write a program to implement insertion sort.

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int size,arr[50],i,j,temp;
cout<<"Enter the size of an array:";
cin>>size;
cout<<"\nEnter array elements:";
for(i=0;i<size;i++)
{
cin>>arr[i];
}
cout<<"\nSorting array using insertion
sort:\n";
for(i=1;i<size;i++)
{
temp=arr[i];
j=i-1;
while((temp<arr[j])&&(j<=0))
{
arr[j+1]=arr[j];
j=j-1;
}
arr[j+1]=temp;
}
cout<<"\nArray after sorting:\n";
for(i=0;i<size;i++)
{
cout<<arr[i]<<"\t";
}
getch();
}
```

Output:

```
Enter the size of an array:5

Enter array elements:2 6 1 7 4

Sorting array using insertion sort:

Array after sorting:
2       6       1       7       4
```

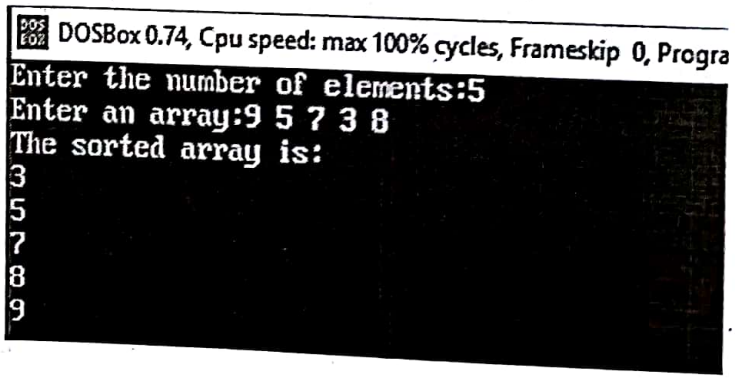
Practical no.06(a)

Aim: Write a program to implement merge sort.

Code:

```
#include<iostream.h>
#include<conio.h>
void mergesort(int[],int,int);
void merge(int[],int,int,int);
void main()
{
int a[10],p,q,r,i,n;
clrscr();
cout<<"Enter the number of
elements:";
cin>>n;
p=0;
r=n-1;
cout<<"Enter an array:";
for(i=0;i<n;i++)
{
cin>>a[i];
}
mergesort(a,p,r);
cout<<"The sorted array is:";
for(i=0;i<n;i++)
{
cout<<"\n"<<a[i];
}
getch();
}
void mergesort(int a[],int p,int r)
{
if(p<r)
{
int 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 c[10];
int i=p;
int j=q+1;
int k=p;
while((i<=q)&&(j<=r))
{
if(a[i]<a[j])
{
c[k]=a[i];
i=i+1;
k=k+1;
}
else
{
c[k]=a[j];
j=j+1;
k=k+1;
}
}
while(i<=q)
{
c[k]=a[i];
i=i+1;
k=k+1;
}
while(j<=r)
{
c[k]=a[j];
j=j+1;
k=k+1;
}
int l=p;
while(l<=r)
{
a[l]=c[l];
l=l+1;
}}
```

Output:



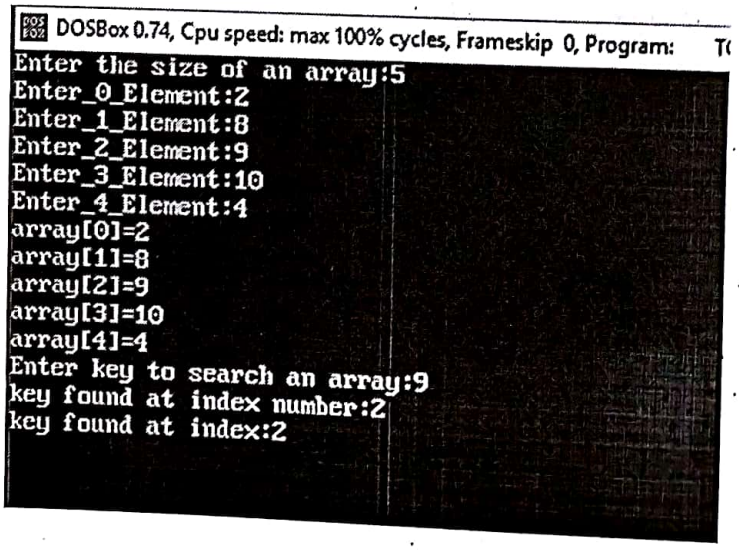
Practical no.06(b)

Aim: write a program to search the element using sequential search

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
cout<<"Enter the size of an array:";
int size;
cin>>size;
int array[10],key,i;
for(i=0;i<size;i++)
{
cout<<"Enter_"<<i<<"_Element:";
cin>>array[i];
}
for(i=0;i<size;i++)
{
cout<<"array["<<i<<"]=";
cout<<array[i]<<endl;
}
cout<<"Enter key to search an array:";
cin>>key;
for(i=0;i<size;i++)
{
if(key==array[i])
{
cout<<"key found at index number:"<<i<<endl;
break;
}
}
if(i!=size)
{
cout<<"key found at index:"<<i;
}
else
{
cout<<"key not found in array";
}
getch();
}
```

Output:



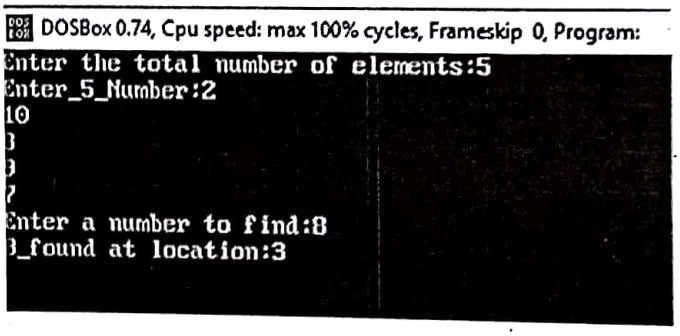
Practical no.06(c)

Aim: Write a program to search element using binary search.

Code:

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int n,arr[50],search,first,last,middle,i;
cout<<"Enter the total number of elements:";
cin>>n;
cout<<"Enter_"<<n<<"_Number:";
for(i=0;i<n;i++)
{
cin>>arr[i];
}
cout<<"Enter a number to find:";
cin>>search;
first=0;
last=n-1;
middle=(first+last)/2;
while(first<=last)
{
if(arr[middle]<search)
{
first=middle+1;
}
else if(arr[middle]==search)
{
cout<<search<<"_found at location:"<<middle+1<<"\n";
break;
}
else
{
last=middle-1;
}
middle=(first+last)/2;
}
if(first>last)
{
cout<<"Not found!_"<<search<<"_is not present in the list";
}
getch();
}
```

Output:



Aatif 11/10/18

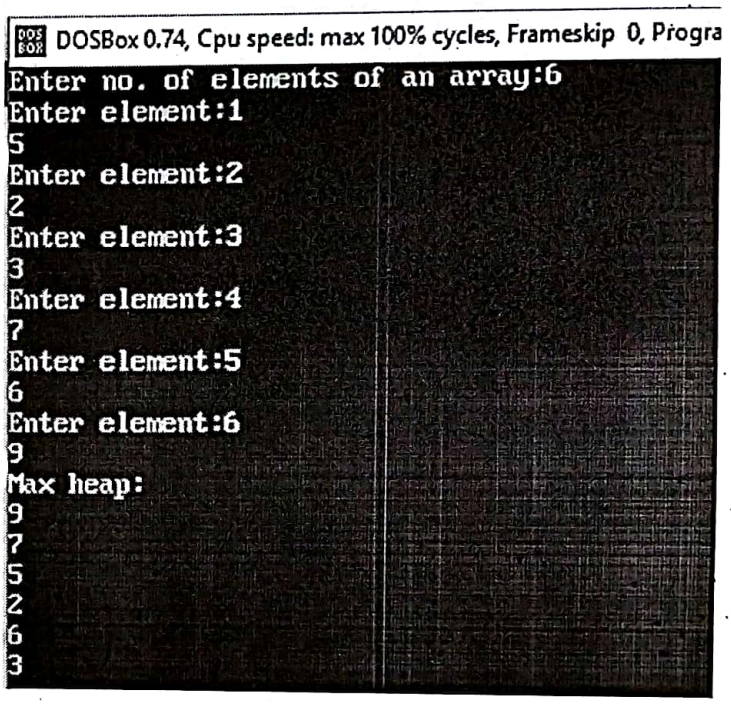
Practical no.07(a)

Aim: Write a program to insert the element into maximum heap.

Code:

```
#include<iostream.h>
#include<conio.h>
void max_heapify(int *a,int i,int n)
{
int j,temp;
temp=a[i];
j=2*i;
while(j<=n)
{
if(j<n&&a[j+1]>a[j])
j=j+1;
if(temp>a[j])
break;
else if(temp<=a[j])
{
a[j/2]=a[j];
j=2*j;
}
}
a[j/2]=temp;
return;
}
void build_maxheap(int *a,int n)
{
int i;
for(i=n/2;i>=1;i--)
{
max_heapify(a,i,n);
}
}
void main()
{
clrscr();
int n,i,x;
cout<<"Enter no. of elements of an
array:";
cin>>n;
int a[20];
for(i=1;i<=n;i++)
{
cout<<"Enter element:"<<i<<endl;
cin>>a[i];
}
build_maxheap(a,n);
cout<<"Max heap:\n";
for(i=1;i<=n;i++)
{
cout<<a[i]<<endl;
}
getch();
}
```

Output:



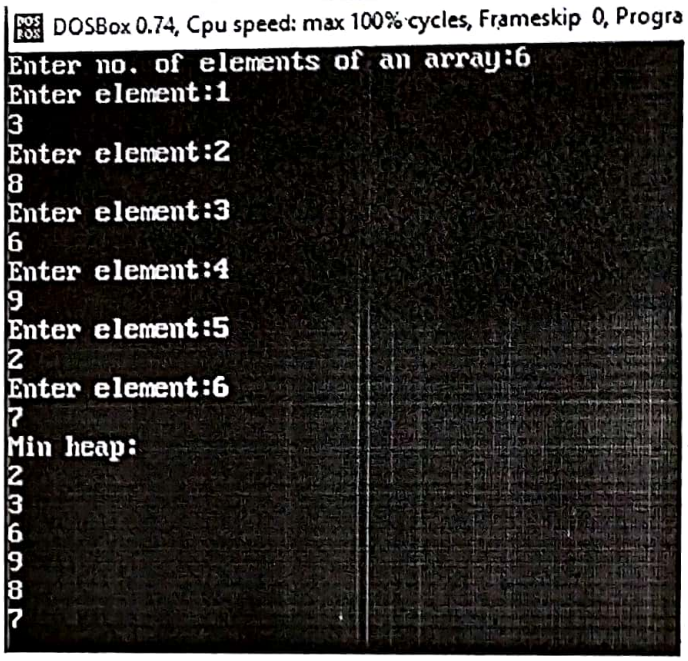
Practical no.07(b)

Aim: Write a program to insert the elements into minimum heap.

Code:

```
#include<iostream.h>
#include<conio.h>
void min_heapify(int *a,int i,int n)
{
int j,temp;
temp=a[i];
j=2*i;
while(j<=n)
{
if(j<n&&a[j+1]<a[j])
j=j+1;
if(temp<a[j])
break;
else if(temp>=a[j])
{
a[j/2]=a[j];
j=2*j;
}
}
a[j/2]=temp;
return;
}
void build_minheap(int *a,int n)
{
int i:
for(i=n/2;i>=1;i--)
{
min_heapify(a,i,n);
}
}
void main()
{
clrscr();
int n,i,x;
cout<<"Enter no. of elements of an
array:";
cin>>n;
int a[20];
for(i=1;i<=n;i++)
{
cout<<"Enter element:"<<i<<endl;
cin>>a[i]:
}
build_minheap(a,n);
cout<<"Min heap:\n";
for(i=1;i<=n;i++)
{
cout<<a[i]<<endl;
}
getch();
}
```

Output:



Practical no.08

Aim: Write a program to shortest path diagram.

Code:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
int shortest(int,int);
int
cost[10][10],dist[20],i,j,n,k,m,s[20],v,t
otcost,path[20],p;
main()
{
int c;
clrscr();
cout<<"Enter no. of vertices:";
cin>>n;
cout<<"\nEnter no. of edges:";
cin>>m;
cout<<"\nEnter\nEDGE cost:\n";
for(k=1;k<=m;k++)
{
cin>>i>>j>>c;
cost[i][j]=c;
}
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
if(cost[i][j]==0)
cost[i][j]=31999;
cout<<"Enter initial vertex:";
cin>>v;
cout<<v<<"\n";
shortest(v,n);
}
int shortest(int v,int n)
{
int min;
for(i=1;i<=n;i++)
{
s[i]=0;
dist[i]=cost[v][i];
}
path[++p]=v;
s[v]=1;
dist[v]=0;
for(i=2;i<=n;i++)
{
k=-1;
min=31999;
for(j=1;j<=n;j++)
{
if(dist[j]<min&&s[j]!=1)
{
min=dist[j];
k=j;
}
}
if(cost[v][k]<=dist[k])
p=1;
path[++p]=k;
for(j=1;j<=p;j++)
cout<<path[j];
cout<<"\n";
s[k]=1;
for(j=1;j<=n;j++)
if(cost[k][j]!=31999&&dist[j]>=dist[k]
+cost[k][j]&&s[j]!=1)
dist[j]=dist[k]+cost[k][j];
}
getch();
}
```

Dated 1/10/18

Output:

```
Enter no. of vertices:6

Enter no. of edges:11

Enter
EDGE cost:
1 2 50
1 3 45
1 4 10
2 3 10
2 4 15
3 5 30
4 1 10
4 5 15
5 2 20
5 3 35
6 5 3
Enter initial vertex:1
1
14
145
1452
13
1-1
```