Practical No:- 1 Program for Creating Maximum And Minimum Heap Tree

A. Insert

```
#include<iostream>
#include<conio.h>
using namespace std;
class insert
{
public:
int i,j,n,A[50],item;
void getdata();
void putdata();
void Maxinsert(int);
void Mininsert(int);
};
void insert::getdata()
{
cout<<"\n Enter number of nodes: "; cin>>n;
cout<<"\n Enter Element: "; for(i=1;i<=n;i++)
cin>>A[i];
}
void insert::putdata()
{
cout<<"\n Element after insertion= "; for(i=1;i<=n;i++)
cout<<" "<<A[i];
}
void insert::Maxinsert(int  n)
{
j=n; i=n/2;
item=A[n];
while(i>0 && A[i]<item)
{
A[j]=A[i];
j=i; i=j/2;
}
A[j]=item;
}
void insert::Mininsert(int  n)
{
j=n;
```

```
i=n/2;
item=A[n];
while(i>0 && A[i]>item)
{
A[j]=A[i];
j=i;
i=j/2;
}
A[j]=item;
}
void main()
{
int ch;
insert m;
//clrscr();
do
{
cout<<"\n\n1.Getdata";
cout<<"\n\n2.Max  Heap";
cout<<"\n\n3.Min  Heap";
cout<<"\n\n4.Exit";
cout<<"\n\n Enter Your Choice :-";
cin>>ch;
switch(ch)
{
case 1:
m.getdata();
break;
case 2:
cout<<"\n\n ******Max Heap******";
for(int k=2;k<=m.n;k++)
m.Maxinsert(k);
m.putdata();
break;
case 3:
cout<<"\n\n ******Min Heap******";
for(int h=2;h<=m.n;h++)
m.Mininsert(h);
m.putdata();
break;
```

```
case 4:break;
default:
cout<<"\n\n Invalid Choice";
}
}
while(ch!=4);
getch();
}
```

/*OUTPUT*/

```
1.Getdata
2.Max Heap
3.Min Heap
4.Exit

 Enter Your Choice :-1

 Enter number of nodes: 5

 Enter Element: 10 20 70 30 60

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

 Enter Your Choice :-2

 ******Max Heap******
 Element after insertion=  70 60 20 10 30

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

 Enter Your Choice :-3

 ******Min Heap******
```

Element after insertion= 10 20 60 70 30

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

Enter Your Choice :-4

B. Adjust/Heapify

```
#include<iostream>
#include<conio.h>
#include<stdio.h>
using namespace std;
int a[20],i,j,n,ch;
class heap_adjust
{
public:
void getdata();
void MinAdjust(int[],int,int);
void MaxAdjust(int[],int,int);
void Heapify(int[],int);
void show();
}h;
void heap_adjust::getdata()
{
cout<<"\n Enter the number of nodes: ";
cin>>n;
cout<<"\n Enter Element: ";
for(i=1;i<=n;i++)
cin>>a[i];
}
void heap_adjust::Heapify(int  a[],int n)
{
if(ch==2)
{
for(i=n/2;i>=1;i--)
        MaxAdjust(a,i,n);
}
if(ch==3)
{
for(i=n/2;i>=1;i--)
        MinAdjust(a,i,n);
}
}
void heap_adjust::MinAdjust(int  a[],int i,int n)
{
int item=a[i];
```

```
j=2*i;
while(j<=n)
{
if(j<n && a[j]>a[j+1])
        j++;
if(item<=a[j])
        break;
else
{
a[j/2]=a[j];
j=j*2;
}
a[j/2]=item;
}
}
void heap_adjust::MaxAdjust(int  a[],int i,int n)
{
int item=a[i]; j=2*i; while(j<=n)
{
if(j<n&&a[j]<a[j+1])
        j++;
if(item>=a[j])
        break;
else
{
a[j/2]=a[j];
j=j*2;
}
a[j/2]=item;
}
}
void heap_adjust::show()
{
for(i=1;i<=n;i++)
        cout<<" "<<a[i];
}
void main()
{
//clrscr();
do
```

```
{
cout<<"\n\n1.Getdata";
cout<<"\n\n2.Max Heap";
cout<<"\n\n3.Min Heap";
cout<<"\n\n4.Exit";
cout<<"\n\n Enter Your Choice :-";
cin>>ch;
switch(ch)
{
case 1:
h.getdata();
break;
case 2:
cout<<"\n Maximum heap Elements are:- \n\n";
h.Heapify(a,n);
h.show();
break;
case 3:
cout<<"\n Minimum heap elements are:- \n\n";
h.Heapify(a,n);
h.show();
break;
case 4:
break;
default:
cout<<"\n\n Invalid Choice";
}
}
while(ch!=4);
getch();
}
```

/*OUTPUT*/

```
1.Getdata
2.Max Heap
3.Min Heap
4.Exit

Enter Your Choice :-1
```

Enter the number of nodes: 5

Enter Element: 10 40 20 30 50

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

Enter Your Choice :-2

Maximum heap Elements are:-

50 40 20 30 10

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

Enter Your Choice :-3

Minimum heap elements are:-

10 30 20 50 40

1.Getdata
2.Max Heap
3.Min Heap
4.Exit

Enter Your Choice :-4