**AIM:** Sort a given set of N integer elements using **Heap Sort** technique and compute its time taken. Run the program for different values of N and record the time taken to sort. Plot a graph of the time taken versus N using MS Excel. The program should allow both manual entry of the array elements and also reading of array elements using random number generator.

**ALGORITHM** : build\_heap(a[0….n-1])

// constructs a max heap from the elements in the given array

// Input : An array a[0….n-1] of orderable elements

//Output : a[0….n-1] contains a max heap

**for** i🡨(n-1)/2 downto 0 **do**

heapify(a,n,p)

**end for**

**ALGORITHM :** heapify(a[0….n-1],p)

//create a heap for a subtree whose root node is identified as parent node ***p***

//Input : An array a[0….n-1] of orderable elements

//Output : The subtree whose root node was identified as parent node ***p*** will be in a heap

item🡨a[p]

c🡨2\*p+1

**while** c<=n-1 **do**

**if** c+1<=n-1

**if** a[c]<a[c+1]

c🡨c+1

**end if**

**end if**

**if** item<a[c]

a[p]🡨a[c]

p🡨c

c🡨2\*p+1

**else**

**break**

**end if**

**end while**

a[p]🡨item

**ALGORITHM** : heap\_sort(a[0….n-1])

// To sort the items by using heap

//Input : The items of array a[0….n-1] to be sorted

//Output : a[0…n-1] contains sorted items

**for** i🡨n-1 downto 0 **do**

swap a[0] and a[i]

build\_heap(a,i)

**end for**

#include<stdio.h>

#include<conio.h>

voidbottom\_up\_heapify(int n, int a[], int p)

{

intitem,c;

item=a[p];

c=2\*p+1;

while(c<=n-1)

{

if(c+1<=n-1)

{

if(a[c]<a[c+1])

c++;

}

if(item<a[c])

{

a[p]=a[c];

p=c;

c=2\*p+1;

}

else

break;

}

a[p]=item;

}

voidtop\_down\_heapify(int n, int a[])

{

intk,c,key,p;

for(k=1;k<n;k++)

{

key=a[k];

c=k;

p=(c-1)/2;

while(c>0 && key>a[p])

{

a[c]=a[p];

c=p;

p=(c-1)/2;

}

a[c]=key;

}

}

voidheap\_sort(int n, int a[])

{

inti,temp;

for(i=n-1;i>0;i++)

{

temp=a[0];

a[0]=a[i];

a[i]=temp;

bottom\_up\_heapify(i,a,0);

}

}

void main()

{

inti,n,a[20];

clrscr();

printf("Enter the value of n\n");

scanf("%d", &n);

printf("Enter the elements to sort\n");

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

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

top\_down\_heapify(n,a);

heap\_sort(n,a);

printf("The sorted vector is\n");

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

printf("%d", &a[i]);

getch();

}

1. **Output:**

Enter the value of n

4

Enter the elements to sort

5

4

3

2

The sorted vector is

2

3

4

5

**Warshall Algorithm**

#include<stdio.h>

#include<conio.h>

int n,a[10][10],p[10][10];

void warshall(int n,int a[10][10],int p[10][10])

{

int i,j,k;

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

for(j=0;j<n;j++)

p[i][j]=a[i][j];

for(k=0;k<n;k++)

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

for(j=0;j<n;j++)

if((p[i][j]==0) && (p[i][k]==1 && p[k][j]==1))

p[i][j]=1;

}

void main()

{

int i,j;

clrscr();

printf("enter the number of vertices\n");

scanf("%d",&n);

printf("enter the adjacency matrix\n");

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

{

for(j=0;j<n;j++)

{

scanf("%d",&a[i][j]);

}

}

warshall(n,a,p);

printf("trasitive closure\n");

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

{

for(j=0;j<n;j++)

{

printf("%d\t",p[i][j]);

}

printf("\n");

}

getch(); }