PROGRAM 18:

HEAP SORT:

PROGRAM:

#include<stdio.h>

heapify(int a[], int n, int i)

{

int root,l,r,t;

root = i;

l = 2\*i + 1;

r = 2\*i + 2;

/\* If left child is larger than root \*/

if (l<n && a[l] > a[root])

root = l;

/\*If right child is larger than largest so far \*/

if (r<n && a[r] > a[root])

root = r;

/\* If largest is not root \*/

if (root != i)

{

t=a[i];

a[i] = a[root];

a[root] = t;

/\* Recursively heapify the affected sub-tree \*/

heapify(a, n, root);

}

}

/\* main function to do heap sort \*/

heapsort(int a[], int n)

{

int i,t;

/\* Build heap (rearrange array) \*/

for(i=n/2-1; i>=0; i--)

heapify(a, n, i);

/\* One by one extract an element from heap \*/

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

{

/\* Move current root to end \*/

t= a[0];

a[0]= a[i];

a[i] = t;

/\* call max heapify on the reduced heap \*/

heapify(a, i, 0);

}

}

int main()

{

int a[50],i,n;

printf("Enter total number of elements:");

scanf("%d", &n);

printf("Enter the elements:\n");

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

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

heapsort(a,n);

printf("\n\nAfter Heap sort:\n");

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

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

return 0;

}