**PROGRAM 10: Heap Sort**

Sort a given set of N integer elements using Heap Sort technique and compute its time taken.

**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

// Program for n values

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int arr[1000000];

int temp;

void maxheap(int arr[], int size, int i)

{

int j,k;

for (k = 0; k < 180; k++)

{

for (j = 0; j < 40; j++)

{

}

}

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if (left < size && arr[left] > arr[largest])

largest = left;

if (right < size && arr[right] > arr[largest])

largest = right;

if (largest != i)

{

temp = arr[i];

arr[i] = arr[largest];

arr[largest] = temp;

maxheap(arr, size, largest);

}

}

void heapSort(int arr[], int size)

{

int i;

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

maxheap(arr, size, i);

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

{

temp = arr[0];

arr[0] = arr[i];

arr[i] = temp;

maxheap(arr, i, 0);

}

}

void printArray(int arr[], int n)

{

int i;

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

printf("%d ", arr[i]);

printf("\n");

}

int main()

{

time\_t start, end;

int n;

srand(time(0));

printf("Enter the no of elements \n");

scanf("%d", &n);

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

{

arr[i] = rand();

}

start = time(NULL);

heapSort(arr, n);

end = time(NULL);

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

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

// printArray(arr, n);

printf("The time taken is %.10f\n", difftime(end, start) / CLOCKS\_PER\_SEC);

return 0;

}





