**Program 10- Sort a given set of N integer elements using Heap Sort technique and compute its time taken.**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void heapify(int arr[], int n, int i) {

// Find largest among root, left child and right child

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

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

largest = left;

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

largest = right;

// Swap and continue heapifying if root is not largest

if (largest != i) {

swap(&arr[i], &arr[largest]);

heapify(arr, n, largest);

}

}

// Main function to do heap sort

void heapSort(int arr[], int n) {

// Build max heap

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

heapify(arr, n, i);

// Heap sort

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

swap(&arr[0], &arr[i]);

// Heapify root element to get highest element at root again

heapify(arr, i, 0);

}

}

void printArray(int arr[], int n) {

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

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

printf("\n");

}

int main() {

int n, i, j, temp,c;

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

scanf("%d",&n);

int arr[n];

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

{

arr[i]=rand()%100;

}

printf("The numbers generated are\n");

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

{

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

}

clock\_t begin = clock();

heapSort(arr, n);

printf("Sorted array is \t");

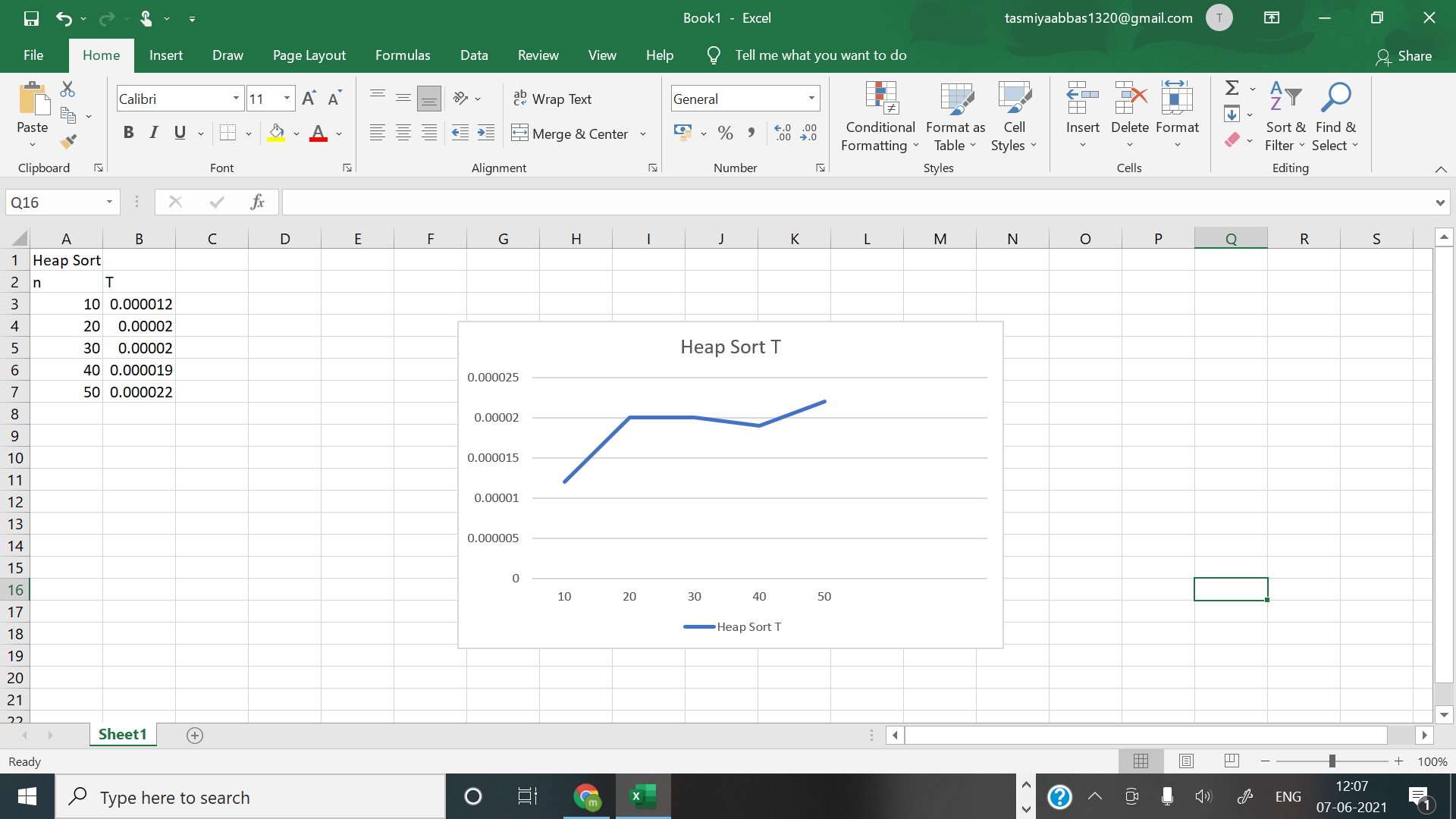
printArray(arr, n);

clock\_t end = clock();

double time\_spent = (double)(end - begin) / CLOCKS\_PER\_SEC;

printf("\n\nEXECUTION TIME : %.10fseconds\n", time\_spent);

}

****