heap sort :-

2016/24

```
#include<stdio.h>
#include<time.h>
#include<stdlib.h>

void heap_sort(int a[], int n);
void heapify(int a[], int n, int i);
void print_array(int a[], int n);

void main(){
    int a[15000], n, i, j, ch;
    clock_t start, end;
    while(1){
        printf("\n1: for manual entry of N value & array elements");
        printf("\n2: To display time taken for sorting number of elements N in the range 500 to 14500");
        printf("\n3: To exit");
        printf("\n Enter your choice: ");
        scanf("%d", &ch);
        switch(ch){
            case 1:
                printf("\n Enter the number of elements:");
                scanf("%d", &n);
                printf(" Enter array elements: ");
                for(i=0; i<n; i++){
                    scanf("%d", &a[i]);
                }
                start = clock();
                heap_sort(a, n);
                end = clock();
                printf("\n Sorted array is :");
                print_array(a, n);
                printf("\n time taken to sort %d numbers is %f secs", n, ((double)(end-start))/CLOCKS_PER_SEC);
                break;
            case 2:
                n=500;
                while(n<=14500){
                    for(i=0; i<n; i++){
                        a[i]=n-i;
                    }
```

```
        start = clock();
        head_sort(a,n);
        for(j=0; j<500000; j++){
            int temp = 38/600;
        }
        End = clock();
        printf("\n Time taken to sort %d numbers
        is %.f sec", n, ((double)(end - start))/
                                CLOCKS_PER_SEC));

        n += 1000;
      }
      break;
    Case 3:
        exit(0);
    }
    getchar();
  }
}
void heap_sort(int a[], int n){
    for(int i = n/2 - 1; i >= 0; i--){
        heapify(a, n, i);
    }
    for(int i=n-1; i>=0; i--){
        int temp = a[0];
        a[0] = a[i];
        a[i] = temp;
        heapify(a, i, 0);
    }
}
void heapify(int a[], int n, int i){
    int largest = i;
    int left = 2*i + 1;
    int right = 2*i + 2;
    if(left < n && a[left] > a[largest]){
        largest = left;
}
```

```
if (right < n && a[right] > a[largest]) {
        largest = right;
    }
    if( largest != i ){
        int temp = a[i];
        a[i] = a[largest];
        a[largest] = temp;
        heapify(a, n, largest);
    }
}
void print_array (int a[], int n){
    for(int i=0; i<n; i++){
        printf("%d\t", a[i]);
    }
}
```

Op:-

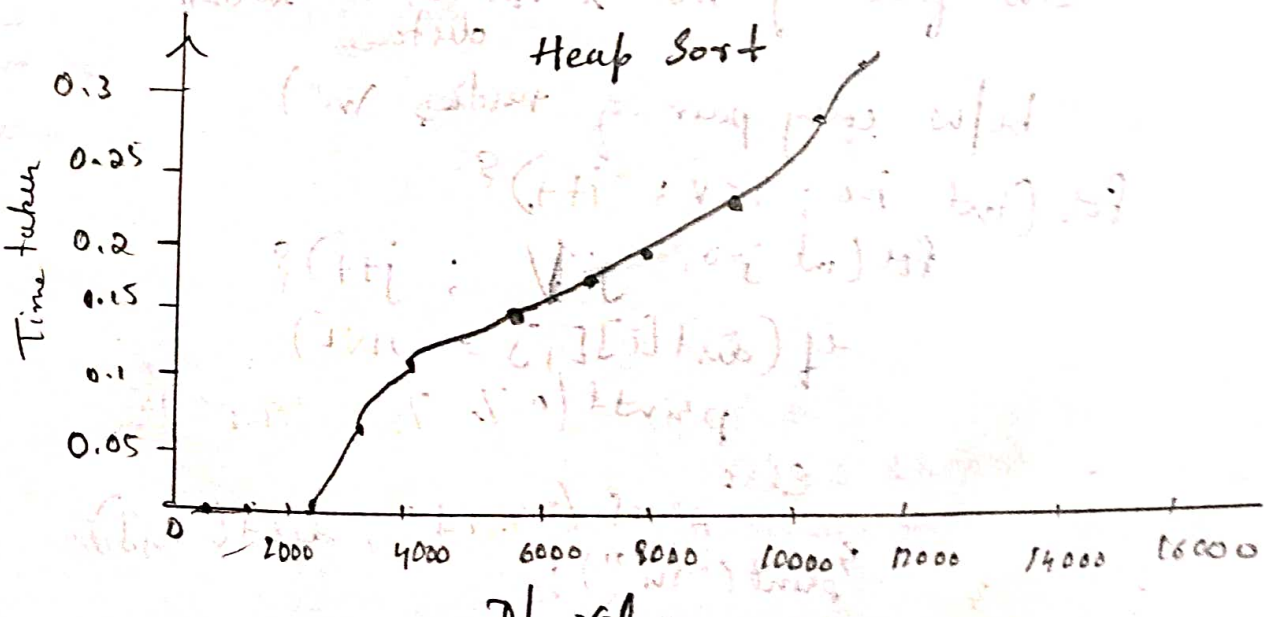
1. For manual entry of N value & array elements
2. To display time taken for sorting number of elements N in the range 500 to 14500.
3. To exit

Enter choice: 1

Enter the number of elements: 8

Enter array elements: 10 20 15 20 25 35 50 60

Sorted array is: 10 15 20 20 25 35 50 60



2) All pair shortest paths problem using Floyd's algorithm

```
#include<stdio.h>
#define V 5
#define INF 99999
void printSolution(int dist[][V]);
void floydWarshall(int dist[][V])
{
    int i, j, k;
    for(k=0; k<V; k++){
        for(i=0; i<V; i++){
            for(j=0; j<V; j++){
                if(dist[i][k] + dist[k][j] <
                        dist[i][j])
                dist[i][j] = dist[i][k] +
                        dist[k][j];
            }
        }
    }
    printSolution(dist);
}
void printSolution(int dist[][V])
{
    printf("The following matrix shows the shortest
                distances"
        "between every pair of vertices\n");
    for(int i=0; i<V; i++){
        for(int j=0; j<V; j++){
            if(dist[i][j] == INF)
                printf("%7s", "INF");
            else
                printf("%7d", dist[i][j]);
        }
        printf("\n");
    }
}
```

```
int main()
{
    int graph[V][V] = { { 0,  4, INF,  5, INF },
                        { INF, 0,  1, INF,  6 },
                        {  2, INF, 3, INF, INF },
                        { INF, INF, 4,  0,  3 },
                        { INF, INF, INF, 1,  0 } };
    floydWarshall(graph);
    return 0;
}
```

o/p :-

The following matrix shows the shortest distance b/w every pair of vertices

```
0    4    5    5    7
3    0    1    4    6
2    6    0    3    5
3    7    1    0    2
1    5    5    4    0
```

9/6/24