LAB-10

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

#include<stdio.h>

int a[10], n;

void heapify(int[], int);

int main() {

printf("Enter the number of array elements:");

scanf("%d", &n);

int i;

printf("Enter array elements:");

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

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

}

heapify(a, n);

printf("Array elements:");

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

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

}

return 0;

}

void heapify(int a[], int n) {

int k;

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

int key = a[k];

int c = k;

int p = (c - 1) / 2;

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

a[c] = a[p];

c = p;

p = (c - 1) / 2;

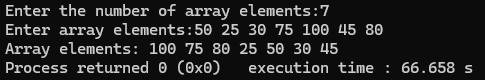
}

a[c] = key;

}

}

OUTPUT:



2. Implement “N-Queens Problem” using Backtracking.

#include <stdio.h>

#include <stdbool.h>

bool place(int[], int);

void printSolution(int[], int);

void nQueens(int);

int main() {

int n;

printf("Enter the number of queens: ");

scanf("%d", &n);

nQueens(n);

return 0;

}

void nQueens(int n) {

int x[10];

int count = 0;

int k = 1;

while (k != 0) {

x[k] = x[k] + 1;

while (x[k] <= n && !place(x, k)) {

x[k] = x[k] + 1;

}

if (x[k] <= n) {

if (k == n) {

printSolution(x, n);

printf("Solution found\n");

count++;

} else {

k++;

x[k] = 0;

}

} else {

k--;

}

}

printf("Total solutions: %d\n", count);

}

bool place(int x[10], int k) {

int i;

for (i = 1; i < k; i++) {

if ((x[i] == x[k]) || (i - x[i] == k - x[k]) || (i + x[i] == k + x[k])) {

return false;

}

}

return true;

}

void printSolution(int x[10], int n) {

int i;

for (i = 1; i <= n; i++) {

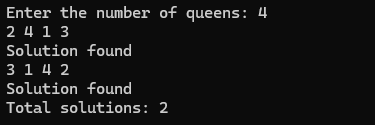
printf("%d ", x[i]);

}

printf("\n");

}

**OUTPUT:**

****