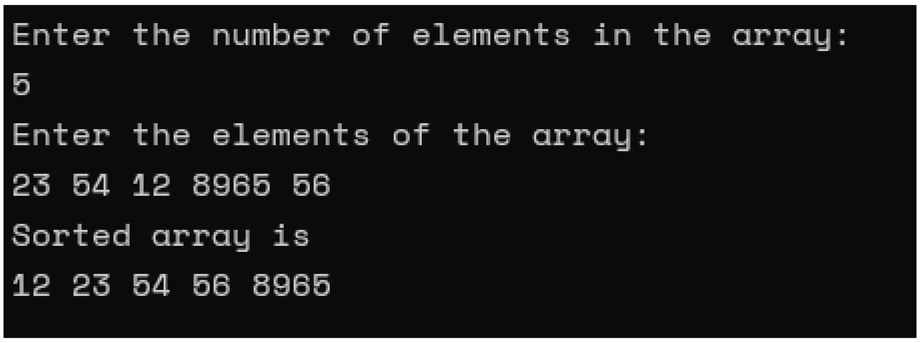
LAB-8

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

```
#include <stdio.h>

void swap(int* a, int* b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}

void heapify(int arr[], int N, int i)
{
    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;

    if (largest != i) {
        swap(&arr[i], &arr[largest]);
        heapify(arr, N, largest);
    }
}

void heapSort(int arr[], int N)
```

```
{

    for (int i = N / 2 - 1; i >= 0; i--)
        heapify(arr, N, i);

    for (int i = N - 1; i >= 0; i--) {
        swap(&arr[0], &arr[i]);
        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 i, N;
    printf("Enter the number of elements in the array:\n");
    scanf("%d", &N);
    int arr[N];
    printf("Enter the elements of the array:\n");
    for(i = 0; i<N; i++)
        scanf("%d", &arr[i]);

    heapSort(arr, N);
    printf("Sorted array is\n");
    printArray(arr, N);
}
```

OUTPUT

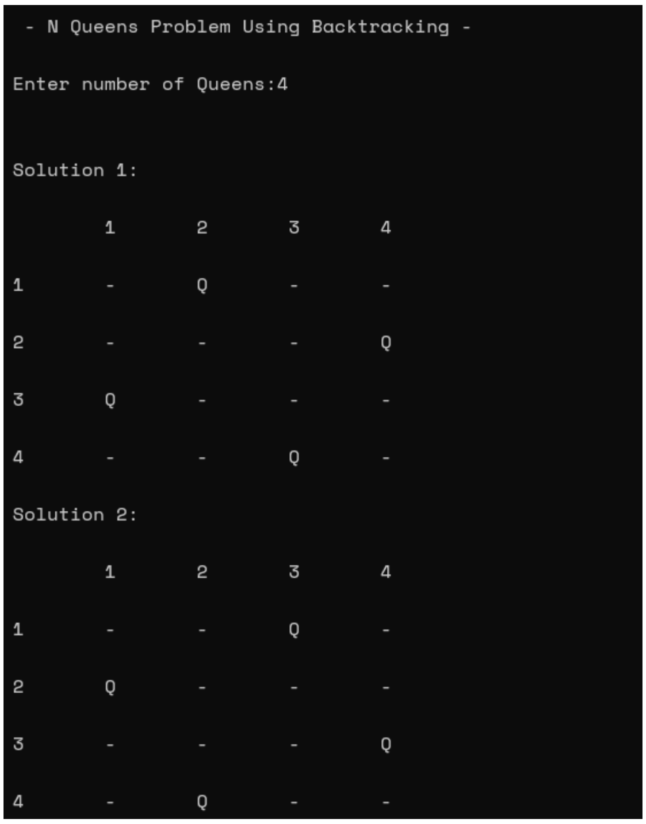


```
Enter the number of elements in the array:
5
Enter the elements of the array:
23 54 12 8965 56
Sorted array is
12 23 54 56 8965
```

Implement “N-Queens Problem” using Backtracking.

```
#include <stdio.h>
#include <math.h>

int board[20], count;

int main()
{
  int n, i, j;
  void queen(int row, int n);

  printf(" - N Queens Problem Using Backtracking -");
  printf("\n\nEnter number of Queens:");
  scanf("%d", &n);
  queen(1, n);
  return 0;
}
void print(int n)
{
  int i, j;
  printf("\n\nSolution %d:\n\n", ++count);
  for (i = 1; i <= n; ++i)
    printf("\t%d", i);

  for (i = 1; i <= n; ++i)
  {
    printf("\n\n%d", i);
    for (j = 1; j <= n; ++j)
      if (board[i] == j)
        printf("\tQ");
      else
        printf("\t-");
  }
}
int place(int row, int column)
{
  int i;
  for (i = 1; i <= row - 1; ++i)
  {
    if (board[i] == column)
      return 0;
    else if (abs(board[i] - column) == abs(i - row))
      return 0;
  }

  return 1;
}
void queen(int row, int n)
{
  int column;
  for (column = 1; column <= n; ++column)
  {
    if (place(row, column))
    {
      board[row] = column;
      if (row == n)
        print(n);
      else
        queen(row + 1, n);
    }
  }
}
```

OUTPUT



```
 - N Queens Problem Using Backtracking -

Enter number of Queens:4


Solution 1:

        1       2       3       4

1       -       Q       -       -

2       -       -       -       Q

3       Q       -       -       -

4       -       -       Q       -

Solution 2:

        1       2       3       4

1       -       -       Q       -

2       Q       -       -       -

3       -       -       -       Q

4       -       Q       -       -
```