// C++ program to find an element x in a

// sorted array using Exponential search.

#include <bits/stdc++.h>

using namespace std;

int binarySearch(int arr[], int, int, int);

// Returns position of first occurrence of

// x in array

int exponentialSearch(int arr[], int n, int x)

{

    // If x is present at first location itself

    if (arr[0] == x)

        return 0;

    // Find range for binary search by

    // repeated doubling

    int i = 1;

    while (i < n && arr[i] <= x)

        i = i\*2;

    //  Call binary search for the found range.

    return binarySearch(arr, i/2,

                            min(i, n-1), x);

}

// A recursive binary search function. It returns

// location of x in  given array arr[l..r] is

// present, otherwise -1

int binarySearch(int arr[], int l, int r, int x)

{

    if (r >= l)

    {

        int mid = l + (r - l)/2;

        // If the element is present at the middle

        // itself

        if (arr[mid] == x)

            return mid;

        // If element is smaller than mid, then it

        // can only be present n left subarray

        if (arr[mid] > x)

            return binarySearch(arr, l, mid-1, x);

        // Else the element can only be present

        // in right subarray

        return binarySearch(arr, mid+1, r, x);

    }

    // We reach here when element is not present

    // in array

    return -1;

}

// Driver code

int main(void)

{

   int arr[] = {2, 3, 4, 10, 40};

   int n = sizeof(arr)/ sizeof(arr[0]);

   int x = 10;

   int result = exponentialSearch(arr, n, x);

   (result == -1)? cout <<"Element is not present in array"

                 : cout <<"Element is present at index " << result;

   return 0;

}