#include <bits/stdc++.h>

using namespace std;

const int maxn = 100 + 10;

int cmp(const void \*a, const void \*b) {

return \*(int \*) a - \*(int \*) b;

}

//普通的二分查找

int bs(int \*arr,int L,int R,int target){

while( L <= R){

int mid = (L) + (R-L)/2;

if(arr[mid] == target)

return mid;

if(arr[mid] > target)

R = mid - 1;

else

L = mid + 1;

}

return -1; // not find

}

//求最小的i，使得a[i] = target，若不存在，则返回-1

//返回 如果有重复的 下界(比如1,2,2,2,3,4)查找2,返回1

int firstEqual(int arr[], int L, int R, int target) {

while (L < R) {

int mid = L + (R - L) / 2;

if (arr[mid] < target)

L = mid + 1;

else

R = mid;

}

if (arr[L] == target)

return L;

return -1;

}

//求最大的i的下一个元素的下标(c++中的upperbound函数)，使得a[i] = target，若不存在，则返回-1

int lastEqualNext(int arr[], int L, int R, int target) {

while (L < R) {

int m = L + (R - L) / 2;

if (arr[m] <= target)

L = m + 1;

else

R = m;

}

if (arr[L - 1] == target)

return L;

return -1;

}

//求最大的i，使得a[i] = target，若不存在，则返回-1

int lastEqual(int arr[], int L, int R, int target) {

while (L < R) {

int mid = L + ((R + 1 - L) >> 1);//向上取整

if (arr[mid] <= target)

L = mid;

else

R = mid - 1;

}

if (arr[L] == target)

return L;

return -1;

}

//求最小的i，使得a[i] > target，若不存在，则返回-1

int firstLarge(int arr[], int L, int R, int target) {

while (L < R) {

int m = L + ((R - L) >> 1);//向下取整

if (arr[m] <= target)

L = m + 1;

else

R = m;

}

if (arr[R] > target)

return R;

return -1;

}

//求最大的i，使得a[i] < target，若不存在，则返回-1

int lastSmall(int arr[], int L, int R, int target) {

while (L < R) {

int m = L + ((R + 1 - L) >> 1);//向上取整

if (arr[m] < target)

L = m;

else

R = m - 1;

}

if (arr[L] < target)

return L;

return -1;

}

int main() {

//freopen("in.txt", "r", stdin);

int n, a[maxn], v;

scanf("%d", &n);

for (int i = 0; i < n; i++)scanf("%d", &a[i]); //1 3 2 9 4 1 3 7 2 2

scanf("%d", &v); //input the number you need find

qsort(a, n, sizeof(a[0]), cmp); // 1 1 2 2 2 3 3 4 7 9

printf("after sorted : \n");

for (int i = 0; i < n; i++)printf("%d ", a[i]);

printf("\n-------------test----------------");

printf("\n%d\n", firstEqual(a, 0, n, v)); //output 2 第一个

printf("%d\n", lastEqualNext(a, 0, n, v)); //output 4 + 1,最后一个的下一个

printf("%d\n", lastEqual(a, 0, n, v)); //output 4 最后一个

printf("%d\n", firstLarge(a, 0, n, v)); //output 5(第一个3大于2)

printf("%d\n", lastSmall(a, 0, n, v)); //output 1(不是0)

return 0;

}