//PROBLEM 1(WEEK 4)

#include<stdio.h>

#define MAX 100

int comp = 0;

int merge\_sort(int A[], int lb, int ub);

int merge(int A[], int lb, int mid, int ub);

int merge\_sort(int A[], int lb, int ub)

{

int inversions = 0;

if(lb < ub)

{

int mid = lb+(ub-lb)/2;

inversions+= merge\_sort(A, lb, mid);

inversions+= merge\_sort(A, mid+1, ub);

inversions+= merge(A, lb, mid, ub);

}

return inversions;

}

int merge(int A[], int lb, int mid, int ub)

{

int inversioncount = 0;

int n1 = mid+1-lb;

int n2 = ub-mid;

int L[MAX], R[MAX];

for(int i = 0; i < n1; i++)

L[i] = A[lb+i];

for(int j = 0; j < n2; j++)

R[j] = A[mid+j+1];

int i = 0, j = 0, k = lb;

while(i<n1 && j<n2)

{

if(L[i] <= R[j])

{

comp++;

A[k] = L[i];

i++;

}

else

{

comp++;

A[k] = R[j];

j++;

inversioncount = inversioncount+(mid-i);

}

k++;

}

while(i<n1)

{

A[k] = L[i];

i++;

k++;

}

while(j<n2)

{

A[k] = R[j];

j++;

k++;

}

return inversioncount;

}

int main()

{

int count = 0;

int n, t, A[MAX];

printf("Enter the number of test cases: ");

scanf("%d",&t);

while(t--)

{

printf("Enter the size of the array: ");

scanf("%d",&n);

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

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

{

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

}

count = merge\_sort(A, 0, n-1);

printf("Sorted Array is: ");

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

{

printf("%d\t",A[i]);

}

}

printf("Total inversions: %d and comparisons: %d\n ",count,comp);

return 0;

}

***OUTPUT:***

Enter the number of test cases: 1

Enter the size of the array: 5

Enter the elements of array: 74 56 068 306 369

Sorted Array is: 56

68

74

306

369

Total inversions: 0 and comparisons: 7

//PROBLEM 2(WEEK 4)

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#define MAX 100

int swaps = 0, compare = 0;

int partition(int A[], int low, int high)

{

int pivot = A[low]; int i = low-1;

int j = high+1;

if(i >= j)

return j;

while(1)

{

compare++;

do

{

i++;

} while(A[i] < pivot);

do

{

j--;

} while(A[j] > pivot);

if(i >= j)

return j;

else

{

int temp = A[i];

A[i] = A[j];

A[j] = temp;

swaps++;

}

}

}

int partition\_random(int A[], int low, int high)

{

srand(time(NULL));

int random = low+rand()%(high-low);

{

int temp = A[random];

A[random] = A[low];

A[low] = temp;

swaps++;

}

return partition(A, low, high);

}

void Quick\_sort(int A[], int low, int high)

{

if(low < high)

{

int part = partition\_random(A, low, high);

printf("%d",part);

Quick\_sort(A, low, part);

Quick\_sort(A, part+1, high);

}

}

int main()

{

int A[MAX], n;

printf("Enter the size of array: ");

scanf("%d",&n);

printf("Enter the elements in the array: ");

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

{

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

}

Quick\_sort(A, 0, n-1);

printf("Sorted array is: ");

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

{

printf("%d\t",A[i]);

}

printf("\ntotal comparisons :%d\n",compare);

printf("total swaps :%d\n",swaps);

return 0;

}

***OUTPUT:***

Enter the size of array: 4

Enter the elements in the array: 285 064 754 23

201Sorted array is: 23 64 285 754

total comparisons :4

total swaps :4

//PROBLEM 3(WEEK 4)

#include<stdio.h>

#include<limits.h>

#define MAX 100

int kth(int A[], int lb, int ub, int k)

{

if(k > 0 && k <= ub-lb+1)

{

int pos = partition(A, lb, ub);

if((pos-lb)==k-1)

return A[pos];

if(pos-lb > k-1)

return kth(A, lb, pos-1, k);

return kth(A, pos+1, ub, k-pos+lb-1);

}

return INT\_MAX;

}

void swap(int\* a, int\* b)

{

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int partition(int A[], int lb, int ub)

{

int x = A[ub], i = lb;

for (int j = lb; j <= ub - 1; j++)

{

if (A[j] <= x)

{

swap(&A[i], &A[j]);

i++;

}

}

swap(&A[lb], &A[ub]);

return i;

}

int main()

{

int t, n, A[MAX];

printf("Enter the number of test cases: ");

scanf("%d",&t);

while(t--)

{

printf("Enter the size of the array: ");

scanf("%d",&n);

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

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

{

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

}

int k;

printf("Enter the kth element: ");

scanf("%d",&k);

printf("%d",kth(A, 0, n-1, k+1));

}

return 0;

}

***OUTPUT:***

Enter the number of test cases: 1

Enter the size of the array: 5

Enter the elements of array: 2 56 89 34 56

Enter the kth element: 56

2147483647