//PROBLEM 1(WEEK 3) **(INSERTION SORT)**

#include<stdio.h>

#define MAX 100

void insertion\_sort(int A[], int n)

{

int comp = 0, shifts = 0;

int i, j, temp;

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

{

temp = A[i];

j = i-1;

shifts++;

while(j >= 0 && temp <= A[j])

{

comp++;

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

j--;

shifts++;

}

A[j+1] = temp;

}

printf("Sorted Array: ");

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

{

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

}

printf("\nTotal shifts: %d and comparisons: %d\n",shifts ,comp);

}

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]);

}

insertion\_sort(A, n);

}

return 0;

}

***OUTPUT:***

Enter the number of test cases: 2

Enter the size of the array: 5

Enter the elements of array: 23 56 78 90 98

Sorted Array: 23 56 78 90 98

Total shifts: 4 and comparisons: 0

Enter the size of the array: 6

Enter the elements of array: 90 9 75 0 67 8

Sorted Array: 0 8 9 67 75 90

Total shifts: 16 and comparisons: 11

//PROBLEM 2(WEEK 3) **(SELECTION SORT)**

#include<stdio.h>

#define MAX 100

void selection\_sort(int A[], int n)

{

int i, j, pos = 0, min = 0;

int swaps = 0, comp = 0;

for(i = 0; i < n-1; i++)

{

pos = i;

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

{

if(A[pos] > A[j])

{

//min = A[j];

pos = j;

}

comp++;

}

if(pos!=i)

{

int temp = A[pos];

A[pos] = A[i];

A[i] = temp;

}

swaps++;

}

printf("Sorted array: ");

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

{

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

}

printf("\nTotal swaps: %d and total comparisons: %d\n",swaps ,comp);

}

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]);

}

selection\_sort(A, n);

}

return 0;

}

***OUTPUT:***

Enter the number of test cases: 2

Enter the size of the array: 5

Enter the elements of array: 67 85 09 0455 37

Sorted array: 9 37 67 85 455

Total swaps: 4 and total comparisons: 10

Enter the size of the array: 6

Enter the elements of array: 28 45 49 0 4578 78

Sorted array: 0 28 45 49 78 4578

Total swaps: 5 and total comparisons: 15

//PROBLEM 3(WEEK 3) **(QUICK SORT)**

#include<stdio.h>

#include<stdlib.h>

#include<stdbool.h>

#define MAX 100

void find\_duplicate(int A[], int lb, int ub);

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

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

void find\_duplicate(int A[], int lb, int ub)

{

int flag = 0;

while(lb <= ub)

{

int mid = (lb+ub)/2;

if(A[mid] == A[mid-1] || A[mid] == A[mid+1])

{

printf("Duplicate element found\n");

flag = 1;

break;

}

else if((A[ub]+A[lb])/2 == A[mid])

lb = mid+1;

else ub = mid-1;

}

if(flag == 0)

{

printf("No duplicate element found\n");

}

}

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

{

int n1 = mid+1-lb;

int n2 = ub-mid;

int L[MAX], R[MAX];

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

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

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

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

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

while(i < n1 && j < n2)

{

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

{

A[k] = L[i];

i++;

}

else

{

A[k] = R[j];

j++;

}

k++;

}

while(i < n1)

{

A[k] = L[i];

k++;

i++;

}

while(j < n2)

{

A[k] = R[j];

j++;

k++;

}

}

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

{

if(lb >= ub)

{

int mid = (ub+lb)/2;

merge\_sort(A, lb, mid);

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

merge(A, lb, mid, ub);

}

}

int main()

{

int result, result2, key;

int A[MAX], n, t;

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]);

}

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

find\_duplicate(A, 0, n-1);

}

return 0;

}

***OUTPUT:***

Enter the number of test cases: 1

Enter the size of the array: 5

Enter the elements of array: 78 79 80 86 90

No duplicate element found