LAB 07

TASK 01

#include <iostream>

#include <climits>

using namespace std;

void swap(int &a, int &b){

    int tmp=a;

    a=b;

    b=tmp;

}

int partition(int\* arr, int low, int high){

    int mid=(low+high)/2;

    swap(arr[mid],arr[high]);

    int pivot=arr[high];

    int index=low-1;

    for (int i=low; i<high; i++) if (arr[i]<=pivot) swap(arr[++index], arr[i]);

    swap(arr[++index], arr[high]);

    return index;

}

void quickSort(int\* arr, int low, int high){

    if(low>=high) return;

    int pi=partition(arr, low, high);

    quickSort(arr, low, pi-1);

    quickSort(arr, pi+1, high);

}

void display(int\* arr,int size){

    cout<<endl;

    for (int i=0; i<size; i++) cout<<arr[i]<<" ";

    cout<<endl;

}

int\* inputArray(int& size){

    cout<<"Input Array Size:";

    cin>>size;

    int\* arr=new int[size];

    for(int i=0 ; i<size ; i++){

        cout<<"Input Element "<<i+1<<" :";

        cin>>arr[size-i-1];

    }

    return arr;

}

int main(){

    int\* arr;

    int size;

    arr=inputArray(size);

    cout<<endl<<"Unsorted: ";

    display(arr,size);

    quickSort(arr,0,size-1);

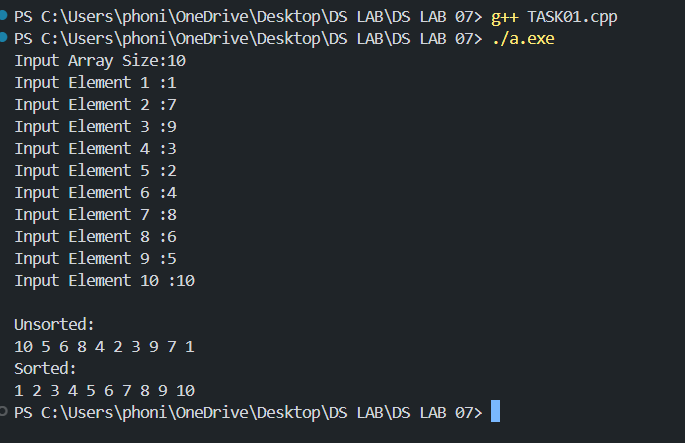
    cout<<"Sorted: ";

    display(arr,size);

    delete[] arr;

    return 0;

}



TASK 02

#include <iostream>

#include <climits>

using namespace std;

void countingSort(int arr[],int size,int pos,bool ascending){

    int freq[10]={0};

    for(int i=0 ; i<size ; i++) freq[(arr[i]/pos)%10]++;

    if(ascending) for(int i=1; i<10; i++) freq[i]+=freq[i-1];

    else for(int i=8; i>=0; i--) freq[i]+=freq[i+1];

    int ans[size];

    for(int i=size-1 ; i>-1 ;i--) ans[--freq[(arr[i]/pos)%10]]=arr[i];

    for(int i=0 ; i<size ; i++) arr[i]=ans[i];

}

int max(int a, int b){

    return (a>b) ? a:b;

}

int maxElement(int arr[],int size){

    int m=INT\_MIN;

    for(int i=0 ; i<size ; i++) m=max(m,arr[i]);

    return m;

}

void radixSort(int arr[], int size,bool ascending){

    int m=maxElement(arr, size);

    for(int pos=1 ; m/pos>0 ; pos\*=10) countingSort(arr,size,pos,ascending);

}

void display(int\* arr,int size){

    cout<<endl;

    for (int i=0; i<size; i++) cout<<arr[i]<<" ";

    cout<<endl;

}

int\* inputArray(int& size){

    cout<<"Input Array Size:";

    cin>>size;

    int\* arr=new int[size];

    for(int i=0 ; i<size ; i++){

        cout<<"Input Element "<<i+1<<" :";

        cin>>arr[size-i-1];

    }

    return arr;

}

int main(){

    int\* arr;

    int size;

    arr=inputArray(size);

    cout<<endl<<"Unsorted: ";

    display(arr,size);

    radixSort(arr,size,true);

    cout<<"Sorted In Ascending Order: ";

    display(arr,size);

    radixSort(arr,size,false);

    cout<<"Sorted In Dscending Order: ";

    display(arr,size);

    delete[] arr;

    return 0;

}

A screenshot of a computer

Description automatically generated

TASK 03

#include <iostream>

#include <climits>

using namespace std;

void merge(int\* arr, int l, int mid, int r,bool ascending){

    int n1=mid-l+1;

    int n2=r-mid;

    int left[n1];

    int right[n2];

    for (int i=0; i<n1; i++) left[i]=arr[l+i];

    for (int j=0; j<n2; j++) right[j]=arr[mid+1+j];

    int i=0,j=0,k=l;

    while(i<n1 && j<n2){

        if(ascending){

            if(left[i]<=right[j]) arr[k++]=left[i++];

            else arr[k++]=right[j++];

        } else {

            if(left[i]>=right[j]) arr[k++]=left[i++];

            else arr[k++]=right[j++];

        }

    }

    while(i<n1) arr[k++]=left[i++];

    while(j<n2) arr[k++]=right[j++];

}

void mergeSort(int\* arr, int left, int right,bool ascending){

    if(left>=right) return;

    int mid=(left+right)/2;

    mergeSort(arr,left,mid,ascending);

    mergeSort(arr,mid+1,right,ascending);

    merge(arr,left,mid,right,ascending);

}

void display(int\* arr,int size){

    cout<<endl;

    for (int i=0; i<size; i++) cout<<arr[i]<<" ";

    cout<<endl;

}

int\* inputArray(int& size){

    cout<<"Input Array Size:";

    cin>>size;

    int\* arr=new int[size];

    for(int i=0 ; i<size ; i++){

        cout<<"Input Element "<<i+1<<" :";

        cin>>arr[size-i-1];

    }

    return arr;

}

int main(){

    int\* arr;

    int size;

    arr=inputArray(size);

    cout<<endl<<"Unsorted: ";

    display(arr,size);

    mergeSort(arr,0,size-1,true);

    cout<<"Sorted In Ascending Order: ";

    display(arr,size);

    mergeSort(arr,0,size-1,false);

    cout<<"Sorted In Dscending Order: ";

    display(arr,size);

    delete[] arr;

    return 0;

}

A screenshot of a computer

Description automatically generated