NAME : Ridhima Goel

SUBGROUP : 2C32

ROLL NO : 1024030466

DSA ASSIGNMENT

QUES 1) Implement the Binary search algorithm regarded as a fast search algorithm with run-time complexity of Ο(log n) in comparison to the Linear Search.

SOLUTION :

#include<iostream>

using namespace std;

main(){

    int arr[100];

    int n ;

    cout<<"enter the size of array : ";

    cin>>n;

    for(int i=0;i<n;i++){

        cout<<"Enter element "<<i+1;

        cin>>arr[i];

    }

    for(int i=0;i<n-1;i++){

        for(int j = 0;j<n-i-1;j++){

            if(arr[j]>arr[j+1]){

                int temp = arr[j];

                arr[j]=arr[j+1];

                arr[j+1]=temp;

            }

        }

    }

    int item;

    cout<<"Enter the no you want to search";

    cin>>item;

    int low = 0;

    int high = n-1;

    int mid;

    int pos = -1 ;

    while(low<=high){

        mid = (low+high)/2;

        if(arr[mid]==item){

            pos = mid;

            break;

        }

        else if ( arr[mid]<item){

            low = mid+1;

        }

        else{

            high = mid + 1 ;

        }

    }

    if(pos!=-1){

        cout<<"Element found at position "<<pos+1<<endl;

    }

    else{

        cout<<"Element not found ";

    }

}

Output:



Ques2) Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. Code the Bubble sort with the following elements:

Solution :

#include<iostream>

using namespace std;

main(){

    int arr[7]={64,34,25,12,22,11,90};

    for(int i = 0;i<=6;i++){

       for(int j =0;j<=6-i;j++){

        if(arr[j]>arr[j+1]){

        int temp = arr[j];

        arr[j]=arr[j+1];

        arr[j+1]=temp;

       }

    }

}

    for(int i = 0; i<7;i++){

        cout<<arr[i];

        cout<<endl;

    }

}

Output:

A screen shot of a computer

AI-generated content may be incorrect.

Ques 3) Given an array of n-1 distinct integers in the range of 1 to n, find the missing number in it in a Sorted Array (a) Linear time (b) Using binary search.

Solution:

#include<iostream>

using namespace std;

main(){

    int n ;

    int arr[100];

    cout<<"enter the size of array : ";

    cin>>n;

    for(int i = 0;i<n-1;i++){

        cout<<"Enter element from (1-n) range : " ;

        cin>>arr[i];

    }

    //sorting the above array

    for(int i = 0;i<n-2;i++){

        for(int j = 0;j<n-i-2;j++){

         if(arr[j]>arr[j+1]){

            int temp = arr[j];

            arr[j]=arr[j+1];

            arr[j+1]=temp;

         }

        }

    }

    for(int i = 0;i<n-1;i++){

        cout<<arr[i];

    }

    //by linear search

    int missing;

    for(int i =0;i<n;i++){

        if(arr[i]!=i+1){

         missing = i+1;

            break;

        }

    }

    cout<<"\nmissing no was : "<<missing;

    //by binary search

    int left = 0;

    int right = n-2;

    int mid;

    while(left<=right){

       mid = (left+right)/2;

       if(arr[mid]==mid+1){

        left = mid+1;

       }

       else{

        right = mid - 1;

       }

    }

cout<<"\nmissing no is :"<<left+1;

}

Ouput:

A screen shot of a computer

AI-generated content may be incorrect.

Ques 4) String Related Programs (a) Write a program to concatenate one string to another string. (b) Write a program to reverse a string. (c) Write a program to delete all the vowels from the string. (d) Write a program to sort the strings in alphabetical order. (e) Write a program to convert a character from uppercase to lowercase.

Solution:

#include<iostream>

using namespace std;

main(){

    int option;

    cout<<"Option 1 : Concatenate one string to another ";

    cout<<"\nOption 2: Reverse a string";

    cout<<"\nOption 3: Delete all vowels from the string";

    cout<<"\nOption 4: Sort the strings inn alphabetic order ";

    cout<<"\nOption 5: Uppercase to lowercase  a string";

    cout<<"\nEnter a option :";

    cin>>option;

    switch(option){

        case 1 :{

            string str1;

            string str2;

            cout<<"Enter one string:";

            getline(cin,str1);

            cout<<"Enter second string:";

            getline(cin,str2);

            str1=str1+ " "+ str2;

            cout<<"Concatenated string : "<<str1;

            break;}

         case 2 :

        { string str3;

         cout<<"Enter a string";

         cin>>str3;

         for(int i = str3.length()-1 ;i>=0;i--){

            cout<<str3[i];

         } break; }

         case 3:{

         string str4;

         cout<<"Enter a string : ";

         cin>>str4;

         for(int i =0;i<str4.length();i++){

            if(str4[i]!= 'a' && str4[i]!='e' && str4[i]!='i'&& str4[i]!='o'&&str4[i]!='u' )

             cout<<str4[i];

        }

      break;}

        case 4 :{

        string str5;

        cout<<"Enter a string: ";

        cin>>str5;

        for(int i= 0;i<str5.length()-1;i++){

            for(int j = 0;j<str5.length()-i-1;j++){

            if(str5[j]>str5[j+1]){

          int  temp = str5[j];

               str5[j]=str5[j+1];

               str5[j+1]=temp;

            }

        }

        }

        for(int i = 0;i<str5.length();i++){

                 cout<<str5[i];

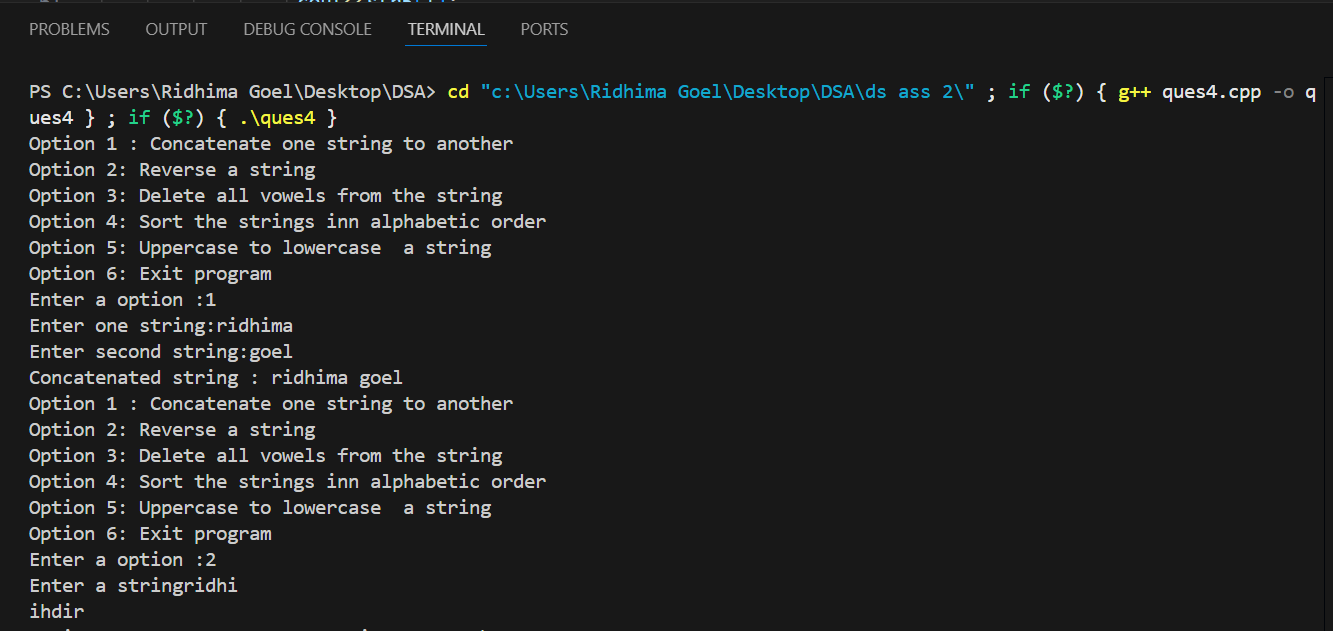
        }

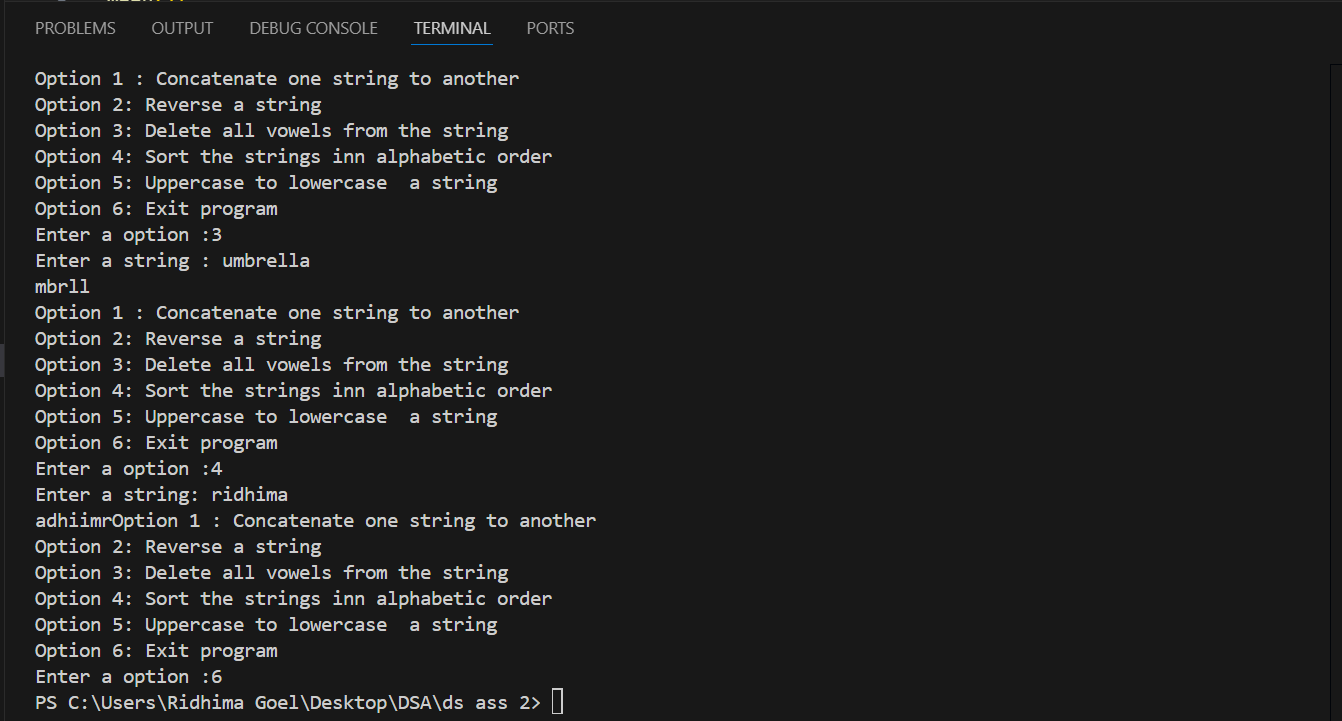
    }

}

}

Output:





Ques 6 Write a program to implement the following operations on a Sparse Matrix, assuming the matrix is represented using a triplet. (a) Transpose of a matrix. (b) Addition of two matrices. (c) Multiplication of two matrices.

Solution

#include<iostream>

using namespace std;

main(){

    int arr[100][100];

    int r;

    int c;

    cout<<"Enter no of rows :";

    cin>>r;

    cout<<"Enter no of columns :";

    cin>>c;

    for(int i = 0;i<r;i++){

        for(int j = 0;j<c;j++){

            cout<<"Enter element "<<"["<<i<<"]"<<"["<<j<<"] :";

            cin>>arr[i][j];

        }

    }

    int sparse[100][3];

    int k = 1;

    for(int i = 0 ;i<r;i++){

        for(int j = 0;j<c;j++){

            if(arr[i][j]!=0){

               sparse[k][0]=i;

               sparse[k][1]=j;

               sparse[k][2]=arr[i][j];

               k++;

            }

        }

    }

    sparse[0][0]=r;

    sparse[0][1]=c;

    sparse[0][2]=k-1;

   for(int i = 0;i<k;i++){

    for(int j =0;j<3;j++){

        cout<<sparse[i][j];

    }

    cout<<endl;

   }

}