DSA Assignment 2

Name: Jashnoor Singh
Roll No. : 1024030106

Answer 1:

```
// Binary search

#include <iostream>
using namespace std;
int main(){
    int num,n,count=0,i;
    cout<<"Enter number of elements: ";
    cin>>num;
    int a[num],s=0,e=num-1,mid;
    cout<<"Enter elements : ";
    for(i=0;i<num;i++){
        cin>>a[i];
    }
    cout<<"Enter number: ";
    cin>>n;
    while(s<=e){
        mid= s + (e-s)/2; // why we are writing this , because sometimes s,e can be just on the verge
of the range of int so adding them could exceed limit thus we write like this
        if(n==a[mid]){
            count=1;
            break;
        }
        else if(n>a[mid]){
            s=mid+1;
        }
        else{
            e=mid-1;
        }
    }
    if(count==1){
        cout<<"Number found\n";
    }
    else{
        cout<<"Number not found";
    }

}
```

Output:

```
Enter number of elements: 5
Enter elements : 1
2
3
4
5
Enter number: 5
Number found
```

Alternate Case:

```
Enter number: 7
Number not found%
```

Answer 2:

// Bubble sort

```
# include <iostream>
using namespace std;
int main(){
    int a[7]={64,34,25,12,22,11,90};
    int i,j,temp;
    for(i=0;i<6;i++){
        for(j=i+1;j<7-i-1;j++){
            if(a[i]>a[j]){
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
            }
        }
    }
    for(i=0;i<7;i++){
        cout<<a[i]<<' ';
    }
}
```

Output:

```
11 22 34 64 25 12 90 %
```

Answer 3:

// Design a logic to find missing number in sorted array

```
#include <iostream>
using namespace std;
int main(){
    int arr[10]={1,2,4,6,7,8,10,11,15,16}; // Let's take an example of sorted array of 10 size where
some numbers are missing
    int diff,i=0;
    cout<<"Missing Numbers are\n";
    while(i<9){ // we are writing here 9 or else array will go out of bounds
        diff=arr[i+1]-arr[i];
        if(diff>1){
            for(int j=arr[i]+1;j<arr[i+1];j++){
                cout<<j<<' ';
            }
        }
        i++;
    }
}
```

Output:

```
Missing Numbers are
3 5 9 12 13 14 %
```

Answer 4:

(A)

```
// Concatenate one string to another

#include <iostream>
#include <string>
using namespace std;
int main(){
    string str1;
    string str2;
    cout<<"Enter first string\n";
    getline(cin,str1);
    cout<<"Enter second string\n";
    getline(cin,str2);
    string str3 = str1 + str2;
    cout<<"\nConcatenated string is \n"<<str3;
}
```

Output:

```
Enter first string
Hello
Enter second string
World

Concatenated string is
HelloWorld%
```

(B)

```
// Reversing of string

#include <iostream>
#include <string>
using namespace std;
int main(){
    string str1;
    int i=0;
    char temp;
    cout<<"Enter your string\n";
    getline(cin,str1);
    int l=str1.length();
    while(i<(l/2)){
        temp=str1[i];
        str1[i]=str1[l-i-1];
```

```
        str1[l-i-1]=temp;
        i++;
    }
    cout<<str1;
}
```

Output:

```
Enter your string
Hello World
dlroW olleH%
```

(C)

```
// Delete all vowels from string

#include <iostream>
#include <string>
using namespace std;
int main(){
    string s;
    int i=0,j;
    cout<<"Enter string\n";
    getline(cin,s);
    int l=s.length();
    while(s[i]!='\0'){
        if(s[i]=='a' || s[i]=='e' || s[i]=='i' || s[i]=='o' || s[i]=='u' || s[i]=='A' || s[i]=='E' || s[i]=='I' || s[i]=='O' ||
s[i]=='U'){
            for(j=i;j<l-1;j++){
                s[j]=s[j+1];
            }
            s[l-1]='\0';
            i--; // This helps us to identify if some vowels are repeated continuously
        }
        i++;
    }
    cout<<s;
}
```

Output:

```
Enter string
Hi there user
H thr sr%
```

(D)

```
// Write a program to sort string in alphabatical order

#include <iostream>
#include <string>
using namespace std;
int main(){
    string s;
    int i=0,j;
    char temp;
    cout<<"Enter string\n";
    getline(cin,s);
    int l=s.length();
    while(s[i]!='\0'){ // converts all letters to lower case
        if(s[i]>=65 & s[i]<=90){
            s[i]+=32;
        }
        i++;
    }
    for(i=0;i<l-1;i++){  // Bubble sort program which sorts according to ascii code
        for(j=i+1;j<l;j++){
            if(s[i]>s[j]){
                temp=s[i];
                s[i]=s[j];
                s[j]=temp;
            }
        }
    }
    cout<<s;

}
```

Output:

```
Enter string
Jashnoor Singh
 aghhijnnoorss%
```

(E)

```
// Conversion of uppercase to lowercase

#include <iostream>
#include <string>
using namespace std;
int main(){
    string s;
    int i=0;
    cout<<"Enter string\n";
    getline(cin,s);
    while(s[i]!='\0'){
        if(s[i]>=65 & s[i]<=90){
            s[i]+=32;
```

```
        }
        i++;
    }
    cout<<s;
}
```

Output:

```
Enter string
LHAFLjjlsJfaljl
lhafljjlsjfaljl%
```

Answer 5:

(A)

```
#include <iostream>
using namespace std;
int main(){
    int a[3][3]={{5,0,0},{0,3,0},{0,0,1}}; // Assume this diagonal matrix
    int b[3]; // In this we will store all the elements of the diagonal matrix
    int i=0,j=0;
    while(i<3 & j<3){
        b[i]=a[i][j];
        i++;
        j++;
    }
    for(i=0;i<3;i++){
        cout<<b[i]<<' ';
    }
}
```

Output:

```
5 3 1 %
```

(B)

// Storing of tri diagonal matrix in array

```
#include<iostream>
using namespace std;
int main(){
    int n;
    cout<<"Enter size of tri-diagonal matrix: ";
    cin>>n;
    int size=3*n-2;
    int arr[size];
    cout<<"Enter elements of the "<<n<<"x"<<n<<" matrix:\n";
```

```
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            int x;
            cin>>x;
            if(i==j){
                arr[i]=x;
            }
            else if(i==j-1){
                arr[n+i]=x;
            }
            else if(i==j+1){
                arr[2*n-1+i]=x;
            }
        }
    }
    cout<<"\nStored 1D array representation:\n";
    for(int i=0;i<size;i++){
        cout<<arr[i]<<" ";
    }
}
```

Output:

```
Enter size of tri-diagonal matrix: 4
Enter elements of the 4x4 matrix:
1
2
0
0
3
4
5
0
0
7
6
8
0
0
9
10

Stored 1D array representation:
1 4 6 10 2 5 8 1 3 7 %
```

(C)

// Store lower triangular matrix in array

```
#include <iostream>
using namespace std;
int main(){
    int n,i,j,k=0;
    cout<<"Enter size: ";
    cin>>n;
    int a[n][n];
    int b[(n*(n+1))/2]; // We will store here
    cout<<"Enter elements: \n";
    for(i=0;i<n;i++){
        for(j=0;j<n;j++){
            cin>>a[i][j];
        }
    }
    i=n-1;
```

```
    while(i>=0){
        j=i;
        while(j>=0){
            b[k]=a[i][j];
            k++;
            j--;
        }
        i--;
    }
    cout<<"Lower triangular matrix in array form is \n";
    for(i=0;i<k;i++){
        cout<<b[i]<<' ';
    }
}
```

Output:

```
Enter size: 3
Enter elements:
1
0
0
2
3
0
4
5
6
Lower triangular matrix in array form is
6 5 4 3 2 1 %
```

(D)

// Store uppper triangular matrix in array

```
#include <iostream>
using namespace std;
int main(){
    int n,i,j,k=0;
    cout<<"Enter size: ";
    cin>>n;
    int a[n][n];
    int b[(n*(n+1))/2]; // We will store here
    cout<<"Enter elements: \n";
    for(i=0;i<n;i++){
        for(j=0;j<n;j++){
            cin>>a[i][j];
        }
    }
    i=0;
    while(i<n){
        j=i;
        while(j<n){
            b[k]=a[i][j];
            k++;
            j++;
        }
```

```
        i++;
    }
    cout<<"Upper triangular matrix in array form is \n";
    for(i=0;i<k;i++){
        cout<<b[i]<<' ';
    }
}
```

Output:

```
Enter size: 3
Enter elements:
1
2
3
0
4
5
0
0
6
Upper triangular matrix in array form is
1 2 3 4 5 6 %
```

(E)

```
#include<iostream>
using namespace std;
int main(){
    int n;
    cout<<"Enter size of symmetric matrix: ";
    cin>>n;
    int size=n*(n+1)/2;
    int arr[size];
    cout<<"Enter elements of symmetric matrix:\n";
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            int x;
            cin>>x;
            if(i>=j){
                int index = i*(i+1)/2 + j;
                arr[index] = x;
            }
        }
    }
    cout<<"\nStored 1D array representation:\n";
    for(int i=0;i<size;i++){
    cout<<arr[i]<<" ";
    cout<<endl;
    }
    cout<<"\nReconstructed Matrix:\n";
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            if(i>=j) cout<<arr[i*(i+1)/2+j]<<" ";
            else cout<<arr[j*(j+1)/2+i]<<" ";
        }
        cout<<endl;
    }
}
```

Output:

```
Enter size of symmetric matrix: 4
Enter elements of symmetric matrix:
1
2
3
4
2
5
6
7
3
6
8
10
4
7
10
9

Stored 1D array representation:
1
2
5
3
6
8
4
7
10
9

Reconstructed Matrix:
1 2 3 4
2 5 6 7
3 6 8 10
4 7 10 9
```

Answer 6:

(A)

```
// transpose of a sparse matrix

#include <iostream>
using namespace std;
int main(){
    int num;
    int a[100][3],b[100][3],i,j; // making a triplet form representation matrix with 100 rows and 3
columns for row,column ,value
    cout<<"Enter number of non zero elements in the matrix: ";
    cin>>num;
    cout<<"Start entering the row number, column number and value :\n";
    for(i=0;i<num;i++){ // now you have formed a triplet form
        cin>>a[i][0]>>a[i][1]>>a[i][2];
    }
    // Now we just have to change row number and column number with each other
    for(i=0;i<num;i++){
        b[i][0]=a[i][1]; // first column of b is written with 2nd column of a
        b[i][1]=a[i][0]; // second column of b is written with 1st column of b
        b[i][2]=a[i][2]; // value remains same
    }
    cout<<"\nOriginal Matrix: \n";
```

```
    // Original matrix:
    for(i=0;i<num;i++){
        for(j=0;j<3;j++){
            cout<<a[i][j]<<' ';
        }
        cout<<endl;
    }
    cout<<"\nTranspose matrix: \n";
    // Transpose matrix:
    for(i=0;i<num;i++){
        for(j=0;j<3;j++){
            cout<<b[i][j]<<' ';
        }
        cout<<endl;
    }
}
```

Output:

```
Enter number of non zero elements in the matrix: 4
Start entering the row number, column number and value :
0
1
5
1
1
6
1
2
8
0
2
3

Original Matrix:
0 1 5
1 1 6
1 2 8
0 2 3

Transpose matrix:
1 0 5
1 1 6
2 1 8
2 0 3
```

(B)

```
#include <iostream>
using namespace std;
int main() {
    int m,n,t1,t2;
    cout<<"Enter rows and cols of matrices: ";
    cin>>m>>n;
    cout<<"Enter number of non-zero elements in first matrix: ";
    cin>>t1;
    int a[100][3];
    cout<<"Enter triplet form (row col value):\n";
    for(int i=0;i<t1;i++){
        cin>>a[i][0]>>a[i][1]>>a[i][2];
    }
```

```
    cout<<"Enter number of non-zero elements in second matrix: ";
    cin>>t2;
    int b[100][3];
    cout<<"Enter triplet form (row col value):\n";
    for(int i=0;i<t2;i++){
        cin>>b[i][0]>>b[i][1]>>b[i][2];
    }
    int c[200][3]; // result
    int i=0,j=0,k=0;
    while (i<t1 && j<t2){
        if (a[i][0]<b[j][0] || (a[i][0]==b[j][0] && a[i][1] < b[j][1])){
            c[k][0]=a[i][0];
            c[k][1]=a[i][1];
            c[k][2]=a[i][2];
            i++;
            k++;
        }
        else if (b[j][0]<a[i][0] || (b[j][0]==a[i][0] && b[j][1]<a[i][1])){
            c[k][0]=b[j][0];
            c[k][1]=b[j][1];
            c[k][2]=b[j][2];
            j++;
            k++;
        }
        else{
            int sum=a[i][2]+b[j][2];
            if (sum!=0){
                c[k][0]=a[i][0];
                c[k][1]=a[i][1];
                c[k][2]=sum;
                k++;
            }
            i++;
            j++;
        }
    }
    while (i<t1){
        c[k][0]=a[i][0];
        c[k][1]=a[i][1];
        c[k][2]=a[i][2];
        i++;
        k++;
    }
    while (j< 2){
        c[k][0]=b[j][0];
        c[k][1]=b[j][1];
        c[k][2]=b[j][2];
        j++;
        k++;
    }
    cout<<"\nResultant Sparse Matrix (Triplet form):\n";
    for(int p=0;p<k;p++){
        cout<<c[p][0]<<" "<<c[p][1]<<" "<<c[p][2]<<endl;
    }
}
```

Output:

```
Enter rows and cols of matrices: 4
4
Enter number of non-zero elements in first matrix: 3
Enter triplet form (row col value):
0
1
4
1
1
5
3
3
8
Enter number of non-zero elements in second matrix: 4
Enter triplet form (row col value):
0
1
5
1
0
4
3
3
1
2
0
8

Resultant Sparse Matrix (Triplet form):
0 1 9
1 0 4
1 1 5
3 3 9
```

(C)

```
#include <iostream>
using namespace std;

int main() {
    int m,n,p,t1,t2;
    cout<<"Enter rows and cols of first matrix: ";
    cin>>m>>n;
    cout<<"Enter number of non-zero elements in first matrix: ";
    cin>>t1;
    int a[100][3];
    cout<<"Enter triplet form (row col value):\n";
    for(int i=0;i<t1;i++){
        cin>>a[i][0]>>a[i][1]>>a[i][2];
    }
    cout<<"Enter cols of second matrix: ";
    cin>>p;
    cout<<"Enter number of non-zero elements in second matrix: ";
    cin>>t2;
    int b[100][3];
    cout<<"Enter triplet form (row col value):\n";
    for (int i=0; i<t2; i++){
        cin>>b[i][0]>>b[i][1]>>b[i][2];
    }
    int c[200][3];
    int k=0;
    for(int i=0;i<t1;i++){
        for (int j=0;j<t2;j++) {
            if (a[i][1]==b[j][0]){
```

```
            int row=a[i][0];
            int col=b[j][1];
            int val=a[i][2]*b[j][2];
            int found=-1;
            for (int x=0;x<k;x++){
                if (c[x][0]==row && c[x][1]==col){
                    found=x;
                    break;
                }
            }
            if (found !=-1){
                c[found][2]+=val;
            } else {
                c[k][0]=row;
                c[k][1]=col;
                c[k][2]=val;
                k++;
            }
        }
    }
}
cout<<"\nResultant Matrix in Triplet form (row col value):\n";
for (int i=0;i<k;i++){
    if (c[i][2]!=0)
        cout<<c[i][0]<<" "<<c[i][1]<<" "<<c[i][2]<<endl;
}
}
```

Output:

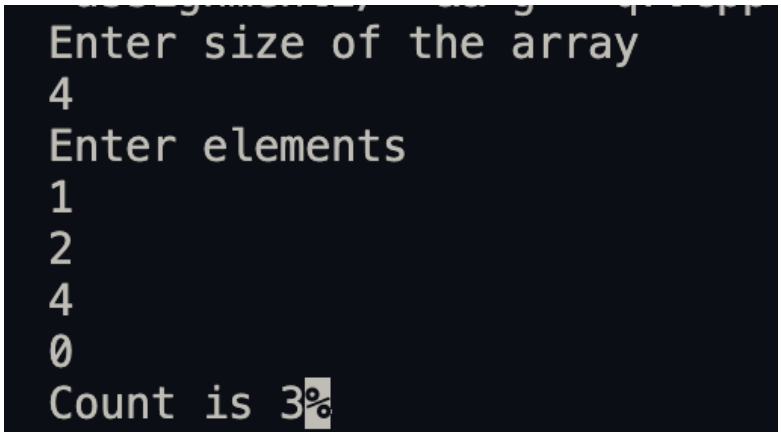
```
Enter rows and cols of first matrix: 3
3
Enter number of non-zero elements in first matrix: 2
Enter triplet form (row col value):
0
1
3
2
2
4
Enter cols of second matrix: 3
Enter number of non-zero elements in second matrix: 1
Enter triplet form (row col value):
1
1
5

Resultant Matrix in Triplet form (row col value):
0 1 15
```

Answer 7:

//Let A[1 …. n] be an array of n real numbers. A pair (A[i], A[j]) is said to be an inversion if these numbers are out of order, i.e., i < j but A[i]>A[j]. Write a program to count the number of inversions in an array.

```
#include <iostream>
using namespace std;
int main(){
    int size,i,j;
    cout<<"Enter size of the array\n";
    cin>>size;
    int count=0;
    int a[size];
    cout<<"Enter elements\n";
    for(i=0;i<size;i++){
        cin>>a[i];
    }
    for(i=0;i<size-1;i++){
        for(j=i+1;j<size;j++){
            if(a[i]>a[j]){
                count++;
            }
        }
    }
    cout<<"Count is "<<count;
}
```

Output:



Answer 8:

// Program to count distinct number of elements in array
// My approach is to delete all the duplicates and then print the size of new array

```
#include <iostream>
using namespace std;
int main(){
    int size,j,k,i;
    cout<<"Enter size of the array\n";
    cin>>size;
    int a[size];
```

```
    cout<<"Start filling the array\n";
    for(i=0;i<size;i++){
        cin>>a[i];
    }
    for(i=0;i<size;i++){
        for(j=i+1;j<size;j++){
            if(a[i]==a[j]){
                for(k=j;k<size-1;k++){
                    a[k]=a[k+1];
                }
                size--;
                j--;
                i--;

            }
        }
    }
    cout<<"\nNumber of distinct elements is "<<size;
}
```

Output:

```
Enter size of the array
8
Start filling the array
1
2
2
7
8
7
10
0

Number of distinct elements is 6%
```

Additional Question:

// Write a program to find a saddle point in a two-dimensional array. A saddle point in a numerical array is a number that is larger than or equal to every number in its column, and smaller than or equal to every number in its row.

```
#include <iostream>
using namespace std;
int main(){
    int count=1;
    int a[3][3]={{3,5,7},{1,4,9},{2,6,8}};
    int i=0,j,min[3],max[3],k;
    while(i<3){ // we are finding maximum number in each column in this loop
        j=0;
        k=a[j][i];
        while(j<2){
            if(a[j+1][i]>k){
```

```
            k=a[j+1][i];
        }
        j++;
    }
    max[i]=k;
    i++;
}
i=0;
while(i<3){
    j=0;
    k=a[i][j];
    while(j<2){
        if(a[i][j+1]<k){
            k=a[i][j+1];
        }
        j++;
    }
    min[i]=k;
    i++;
}
i=0;
while(i<3){
    j=0;
    while(j<3){
        if(max[i]==min[j]){
            count=0;
            break;
        }
        j++;
    }
    if(count==0){
        break;
    }
    i++;
}
if(count==0){
    cout<<"Saddle point is "<<max[i];
}
else{
    cout<<"There exists no saddle point";
}
}
```

Output:

```
Saddle point is 3%
```