S.no	**WEEK 2**
1.	Write a program in C++ perform various operations on Sparse Matrix a. Write a program in C++ to perform Addition of Two Sparse Matrices b. Write a program in C++ to perform Transpose of a Matrix c. Write a program in C++ to Multiply Two Matrices
2.	Write a program in C++ to perform various operations on strings a. Write a program in C++ to find Length of a String. b. Write a program in C++ to Copy String c. Write a program in C++ to Concatenate Strings d. Write a program in C++ to find Frequency of Character in String e. Write a program in C++ to Reverse String f. Write a program in C++ to Compare Two String

1. SPARSE MATRIX MULTIPLICATION

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>

const int max1=3;
const int max2=3;
const int maxsize=20;
#define TRUE 1
#define FALSE 2

class sparse
{
private:
int *sp;
int row;
int *result;
public:
sparse();
void create_array();
int count();
void display();
void create_tuple(sparse &s);
void display_tuple();
void prodmat(sparse &a,sparse &b);
static void searchina(int *sp,int ii,int *p,int *flag);
static void searchinb(int *sp,int jj,int colofa,int *p,int *flag);
void display_result();
~sparse();
};

//initialises data members
sparse::sparse()
{
sp=NULL;
result=NULL;
}

void sparse::create_array()
{
int n;
//allocate memory
sp=new int[max1*max2];

//add elements to array
for(int i=0;i<max1 * max2;i++)
{
cout<<"\nenter element no:"<<i<<":";
cin>>n;
*(sp+i)=n;
}
}
void sparse::display()
{
```

```
//traverse the entire matrix
for(int i=0;i< max1 * max2;i++)
{
//positions the cursor to the new line for every new row
if(i%3==0)
cout<<endl;
cout<<*(sp+i)<<"";
}
}
int sparse::count()
{
int cnt=0;
for(int i=0;i<max1*max2;i++)
{
if(*(sp+i)!=0)
cnt++;
}
return cnt;
}
void sparse::create_tuple(sparse &s)
{
int r=0,c=-1,l=-1;
//get total number of non-zero elements
//add1 store total no.of rows,cols and non zero values
row=s.count()+1;

//allocate memory
sp=new int[row * 3];
//store information about total no. of rows,cols and non zero values

*(sp+0)=max1;
*(sp+1)=max2;
*(sp+2)=row-1;
l=2;

//scan the array and store info about non zero values in tuple
for(int i=0;i<max1*max2;i++)
{
c++;
//set the row and column values
if(((i%3)==0)&&(i!=0))
{
r++;
c=0;
}
//check for non zero element value
if(*(s.sp+i)!=0)
{
l++;
*(sp+l)=r;
l++;
*(sp+l)=c;
l++;
*(sp+l)=*(s.sp+i);
}
}
}
```

```
void sparse::display_tuple()
{
//travers the entire matrix
cout<<"\nelemets in a 3-tuple:"<<endl;
int j=(*(sp+2)*3)+3;
for(int i=0;i<j;i++)
{
//position the cursor to the new line for every row
if(i%3==0)
cout<<endl;
cout<<*(sp+i)<<"";
}
}

void sparse::prodmat(sparse &a,sparse &b)
{
int sum k,position,posi,flaga,flagb;
k=1;
result=new int[maxsize*3];
for(int i=0;i<*(a.sp+0*3+0);i++)
{
for(int j=0;j<*(b.sp+0*3+1);j++)
{
//search if an element present at ith row
searchina(a.sp,i,&position,&flaga);
if(flaga==TRUE)
{
sum=0;
//run looptill there are element at ith row in first 3 tuple
while(*(a.sp+position*3+0)==i)
{

//search if an element present at ith col in second 3-tuple
searchinb(b.sp,j,*(a.sp+position*3+1),&posi,&flagb);
//if found then multiply
if(flagb==TRUE)
sum=sum+*(a.sp+position*3+2)* *(b.sp+posi*3+2);
position=position+1;
}

//add result
if(sum!=0)
{
*(result+k*3+0)=i;
*(result+k*3+1)=j;
*(result+k*3+2)=sum;
k=k+1;
}
}
}
}

//add total no.of rows cols and non zero values
*(result+0*3+0)=*(a.sp+0*3+0);
*(result+0*3+1)=*(b.sp+0*3+1);
*(result+0*3+2)=k-1;
}
```

```
//search if an element is present at ith row
void sparse::searchina(int *sp,int ii,int *p,int *flag)
{
int j;
*flag=FALSE;
for(j=1;j<=*(sp+0*3+2);j++)
{
if(*(sp+j*3+0)==ii)
{
*p=j;
*flag=TRUE;
return;
}
}
}



void sparse::searchinb(int *sp,int jj,int colofa,int *p,int *flag)
{
int j;
*flag=FALSE;
for(j=1;j<=*(sp+0*3+2);j++)
{
if(*(sp+j*3+1)==jj&&*(sp+j*3+0)==colofa)
{
*p=j;
*flag=TRUE;
return;
}
}
}


void sparse::display_result()
{
//traverse the entire matrix
for(int i=0;i<(*(result+0+2)+1)*3;i++)
{
//positions the cursor to the new line for every new row
if(i%3==0)
cout<<endl;
cout<<*(result+i)<<"";
}
}

sparse::~sparse()
{
if(sp!=NULL)
delete sp;
if(result!=NULL)
delete result;
}

void main()
{
clrscr();
sparse s1;
```

```
s1.create_array();

sparse s2;
s2.create_tuple(s1);
s2.display_tuple();

sparse s3;
s3.create_array();

sparse s4;
s4.create_tuple(s3);
s4.display_tuple();

sparse s5;
s5.prodmat(s2,s4);
cout<<endl<<"result of multiplication of two matrices:";
s5.display_result();
getch();
}
```

2. Write a program in C++ to implement different string manipulation operations

a. Write a program in C++ to find Length of a String.

```
#include<iostream.h>
#include<string.h>
#include<conio.h>

const int max=10;

class string
{
private:
char str[max];
public:
string();
string(char *s);
int xstrlen();

void show();
};

//initialise data members
string::string()
{
}
//initialise data members
string::string(char *s)
{
```

```
strcpy(str,s);
}

//finds the length of the string
int string::xstrlen()
{
int l=0;
char *s=str;
while(*s)
{
l++;
s++;
}
return l;
}


void string::show()
{
cout<<str<<endl;
}

void main()
{
clrscr();
string s1("hello");

cout<<endl;
cout<<"string s1:";
s1.show();
cout<<"\nlength of the string s1: "<<s1.xstrlen()<<endl;

getch();
}
```

a. Write a program in C++ to Copy String

```
#include<iostream.h>
#include<string.h>
#include<conio.h>

const int max=10;

class string
{
private:
char str[max];
public:
string();
string(char *s);
static void xstrcpy(string &t,string &s);
```

```
void show();
};

//initialise data members
string::string()
{
}
//initialise data members
string::string(char *s)
{
strcpy(str,s);
}


//copies source string s to the target string t
void string::xstrcpy(string &t,string &s)
{
char *s1=t.str;
char *s2=s.str;

while(*s2)
{
*s1=*s2;
s1++;
s2++;
}
*s1='\0';
}

void string::show()
{
cout<<str<<endl;
}

void main()
{
clrscr();
string s1("hello");
string s2("world");

cout<<endl;
cout<<"string s1:";
s1.show();

cout<<"\nstring2 ";
s2.show();

string s3;
string::xstrcpy(s3,s1);
cout<<"\nstring s3 after copying s1 to it: ";
s3.show();


getch();
}
```

c. Write a program in C++ to Concatenate Strings

```
#include<iostream.h>
#include<string.h>
#include<conio.h>

const int max=10;

class string
{
private:
char str[max];
public:
string();
string(char *s);
static void xstrcat(string &t,string &s);


void show();
};

//initialise data members
string::string()
{
}
//initialise data members
string::string(char *s)
{
strcpy(str,s);
}

//concatenates the two strings
void string::xstrcat(string &t,string &s)
{
char *s1=t.str;
char *s2=s.str;

while(*s1)
s1++;
while(*s2)
*s1++=*s2++;
*s1='\0';

}

void string::show()
{
cout<<str<<endl;
}

void main()
{
clrscr();
string s1("hello");
string s2("world");
```

```
cout<<endl;
cout<<"string s1:";
s1.show();

cout<<"\nstring2 ";
s2.show();


string::xstrcat(s1,s2);
cout<<"\nstring s3 after concatenation ";
s1.show();

getch();
}
```

NOTE: Assignment to be done by students in the Lab

Q1.

a. Write a program in C++ to perform Addition of Two Sparse Matrices
b. Write a program in C++ to perform Transpose of a Matrix

Q2.

d. Write a program in C++ to find Frequency of Character in String
e. Write a program in C++ to Reverse String
f. Write a program in C++ to Compare Two String