Sparse Matrix using C++

```cpp
#include <iostream>

using namespace std;

class Element
{
public:
    int i;
    int j;
    int x;
};

class Sparse
{
private:
    int m;
    int n;
    int num;
    Element *ele;
public:
    Sparse(int m,int n,int num)
    {
        this->m=m;
        this->n=n;
        this->num=num;
        ele=new Element[this->num];
    }
    ~Sparse()
    {
        delete [] ele;
    }

    Sparse operator+(Sparse &s);

    friend istream & operator>>(istream &is,Sparse &s);
    friend ostream & operator<<(ostream &os,Sparse &s);


};

Sparse Sparse::operator+(Sparse &s)
{
    int i,j,k;
    if(m!=s.m || n!=s.n)
        return Sparse(0,0,0);
    Sparse *sum=new Sparse(m,n,num+s.num);

    i=j=k=0;
    while(i<num && j<s.num)
    {
        if(ele[i].i<s.ele[j].i)
            sum->ele[k++]=ele[i++];
        else if(ele[i].i > s.ele[j].i)
            sum->ele[k++]=s.ele[j++];
        else
        {
            if(ele[i].j<s.ele[j].j)
                sum->ele[k++]=ele[i++];
            else if(ele[i].j > s.ele[j].j)
                sum->ele[k++]=s.ele[j++];
            else
            {
                sum->ele[k]=ele[i];
                sum->ele[k++].x=ele[i++].x+s.ele[j++].x;
            }

        }
    }
    for(;i<num;i++)sum->ele[k++]=ele[i];
    for(;j<s.num;j++)sum->ele[k++]=s.ele[j];
    sum->num=k;

    return *sum;

}






    istream & operator>>(istream &is,Sparse &s)
    {
        cout<<"Enter non-zero elements";
        for(int i=0;i<s.num;i++)
            cin>>s.ele[i].i>>s.ele[i].j>>s.ele[i].x;
        return is;
    }

    ostream & operator<<(ostream &os,Sparse &s)
    {
        int k=0;
        for(int i=0;i<s.m;i++)
        {
            for(int j=0;j<s.n;j++)
            {
                if(s.ele[k].i==i && s.ele[k].j==j)
                    cout<<s.ele[k++].x<<" ";
                else
                    cout<<"0 ";
            }
            cout<<endl;
        }
        return os;
    }


int main()
{
    Sparse s1(5,5,5);
    Sparse s2(5,5,5);

    cin>>s1;
    cin>>s2;

    Sparse sum=s1+s2;

    cout<<"First Matrix"<<endl<<s1;
    cout<<"Second MAtrix"<<endl<<s2;
    cout<<"Sum Matrix"<<endl<<sum;

    return 0;
}
```