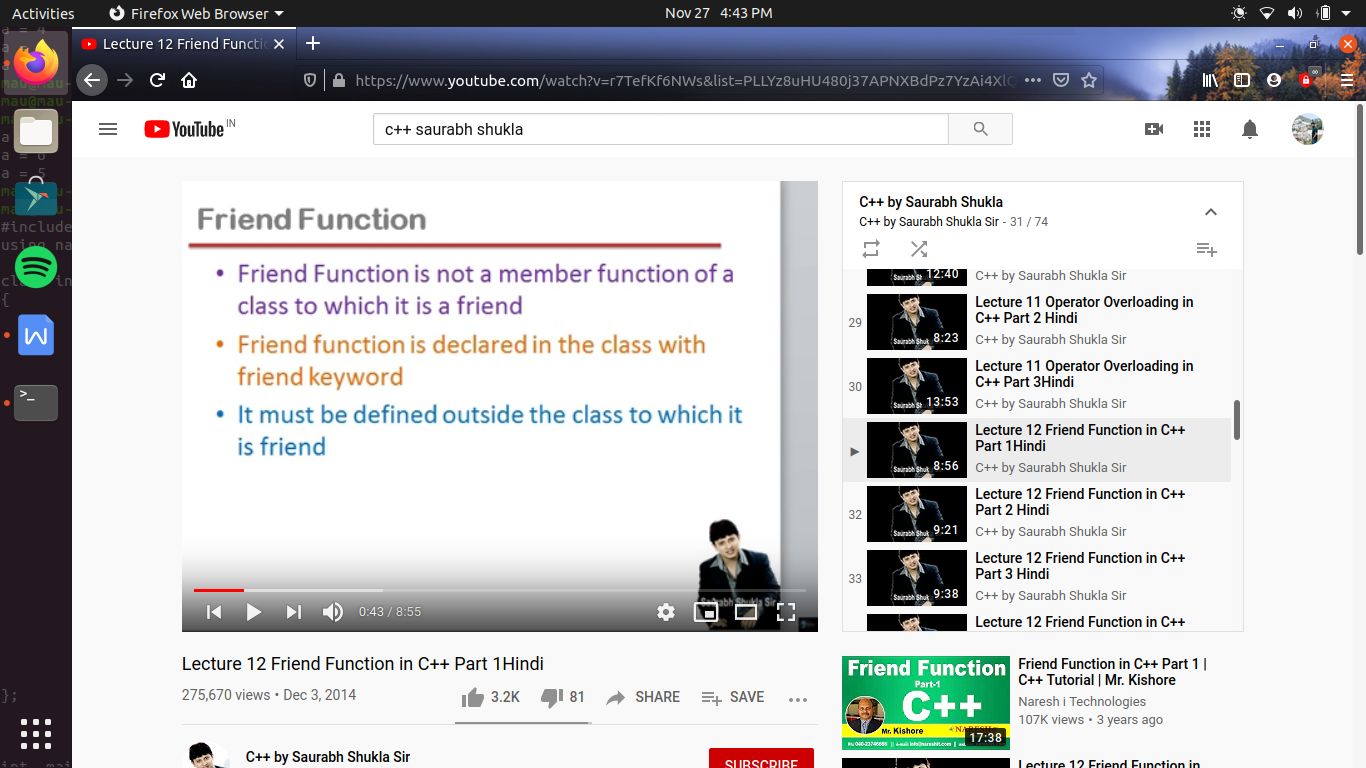
friend function in c++

friend function ek aisa function hai jo class ka member function hai , jo ki us ka friend hai

friend ko bhi class ke andar declare karate hai

lekin define class ke andar nhi balaki bahar karate hai



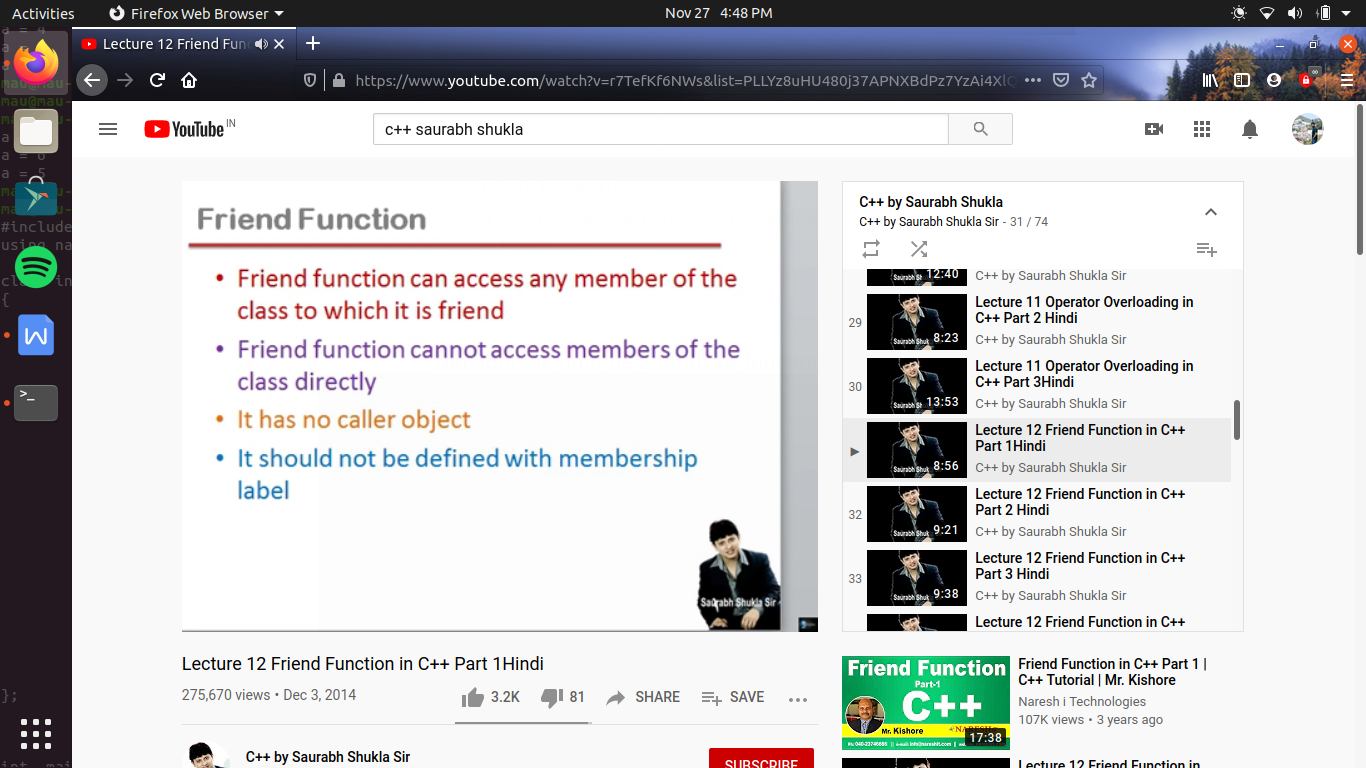
toh abhi tak hum ne 2 tarah ke function dekh ke hai ya toh wo member function hoga class ka or ya toh non member function hota hai

toh yeh friend function en dono ke beech ki chij hai , mtlb yeh class ke leye hai toh non member function lekin yeh class ka friend hai , jisse es ko aisa kayi sare adhikar mil jate hai jo ki us ke member function ko mile hota hai

jaise member function class ke kisi bhi member ko access kr sakaata hai , friend function bhi class ke kisi bhi member ko access kar sakata hai

i.e friend member us class ke private member ko bhi access kr sakata hai lekin sirf non-member function class ke sirf public member ko access kr sakata hai toh but function can not access member function directly as member function do

friend function ka koe call object nhi hoga kyu ki yeh member function nhi hai or es ko define karate time membership label bhi nhi lagaya jayega



#include<iostream>

using namespace std;

class complex

{

private :

int a,b;

public:

void set\_data (int x, int y)

{a=x;b=y;}

void show\_data()

{cout<<"a = "<<a<<"\nb = "<<b<<endl;}

friend void fun(complex); //agar yeh function ko declare karate waqat friend keyword likh diya jaye toh yeh friend function mana jayega

//friend function class me keval declare hota hai , or es ko bahar define karana padata hai

};

void fun(complex c1) //ab decleare kaarte time nahi hum yaha dobara frined likhege na hi membership label kagaye ge

{

// cout<<"sum is "<<a+b; //toh yeh bhi galat hai kyu ki friend fuction ko class ke private memeber ko access kaarne ka adhikar hai pr direclty nhi

// ab directly kyu nhi kyu ki yeh a or b kis object ke hai yeh pata nhi chalega c1 ke hai ya c2, c3 ke hai , kyu ki es ka koe caller object nhi hai

// toh es ke leye hum es me argument pass karege us ke leye hame declare karate hue or define karate hue batana padega

cout<<"sum is "<<c1.a+c1.b; //toh hum private member ko toh access kr pa rahe hai but direclty nhi kr pa rahe hai

}

int main()

{

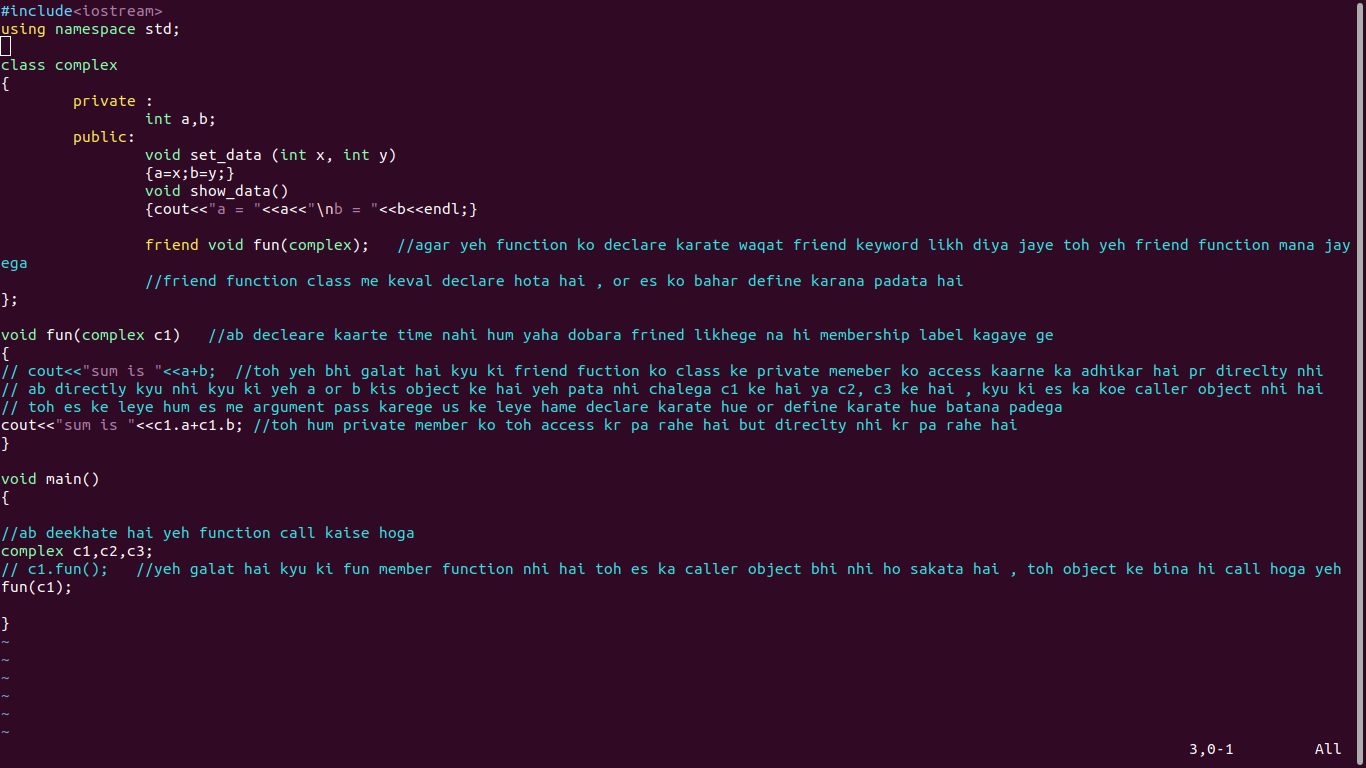
//ab deekhate hai yeh function call kaise hoga

complex c1,c2,c3;

// c1.fun(); //yeh galat hai kyu ki fun member function nhi hai toh es ka caller object bhi nhi ho sakata hai , toh object ke bina hi call hoga yeh

fun(c1);

}



friend function ek hi sath kayi sari classes ka friend banaya ja sakata hai or yeh ek friend function ka unique behaviour hoga , jis ka ek fayada hoga

#include<iostream>

using namespace std;

class B; //forward decleration

class A

{ private:

int a;

public:

void set\_data(int x)

{a=x;}

friend void fun(A,B); //yaha pr class B kyu ki badme bana hai toh compiler samjh nhi pa raha hai ki yeh B kya hai toh es ke leye

//hum class B ka forward decleration kar dege Class A se pahale

};

class B

{

private:

int b;

public:

void set\_data(int y)

{b=y;}

friend void fun(A,B);

};

void fun(A a1, B b2) //yeh ek aisa friend function hai jo dono class ke private member ko ek hi sath use kr raha hai

//toh agar kabhi aisi situation ati hai ki hum ek se jayada class ke priavte member ko ek sath use karana hai toh yeh kam sirf friend function kr sakata hai

//yeh friend function ka pahala faida hai , kul 3 fayade important hai

{

cout <<"sum is "<<a1.a+b2.b<<endl;

}

//friend function chahe toh private declare kare ya public declare kare ese koe fark nhi padega kyu ki yeh ek non member function hai

// or hum ne phichale lecture me yeh bat samajhi thi ki agar friend function me agar koe argument pass nhi karate hai toh yeh etana meaning full nhi rahega

// kyu ki es function ko call karane ke leye koe caller object toh ho ga nhi toh hum direclty use nhi kr payege class ke member ko

// toh es ke leye hame argument pass karana jaruri hai

int main()

{

A a1;

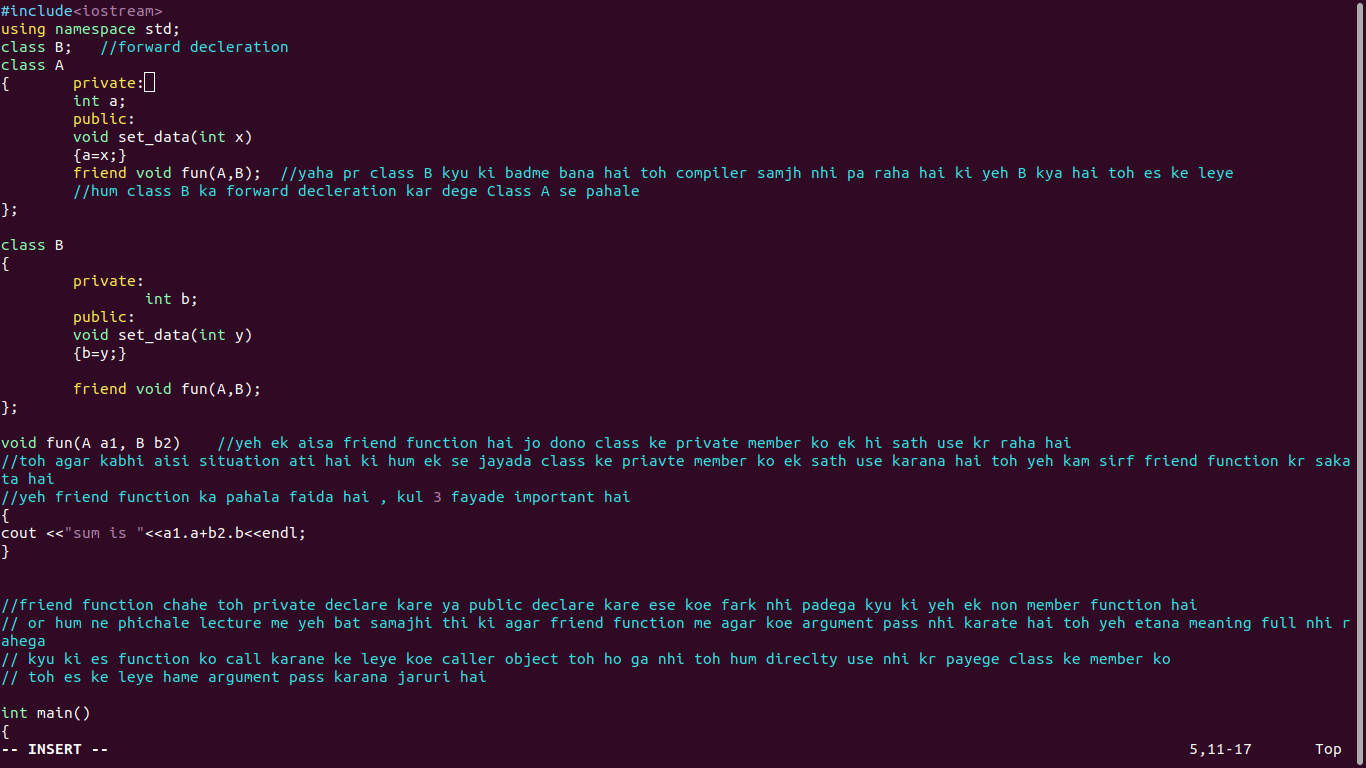
B b2;

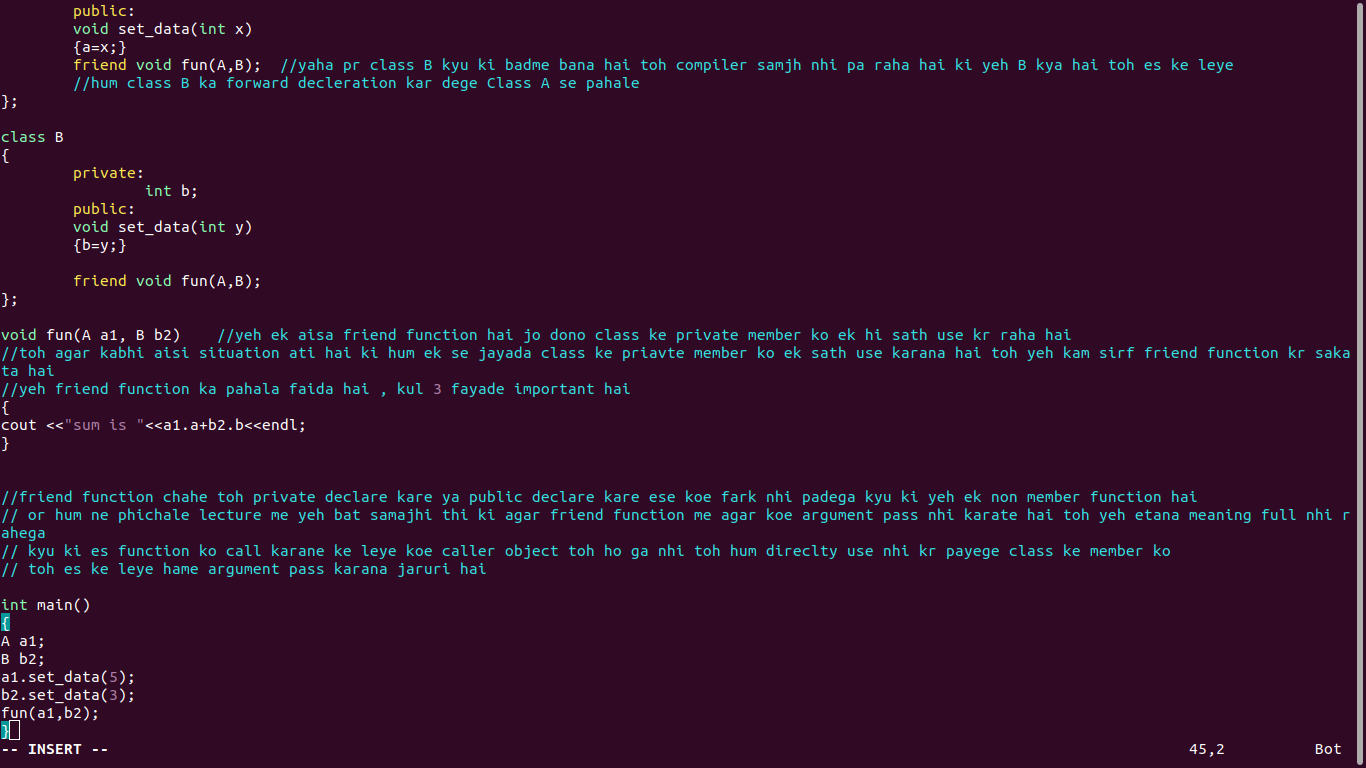
a1.set\_data(5);

b2.set\_data(3);

fun(a1,b2);

}





how to define friend function using operator

i.e operator ki overloading as an friend function kaise karege

#include<iostream>

using namespace std;

class complex

{

private :

int a,b;

public:

void set\_data (int x, int y)

{a=x;b=y;}

void show\_data()

{cout<<"a = "<<a<<"\nb = "<<b<<endl;}

friend complex operator + (complex, complex);

};

complex operator +(complex b1, complex b2)

{

complex tempt;

tempt.a=b1.a+b2.a;

tempt.b=b1.b+b2.b;

return tempt;

}

int main()

{

complex b1,b2,b3;

b1.set\_data(3,4);

b2.set\_data(5,6);

b3=b1+b2; // hum yaha chahte hai ki yeh + operator ko hum as an friend function define kare toh kyun kyu ki friend function member nhi hota class ka

//toh hum ese andar define nhi kr sakate , andar sirf es ka decleration ayega

//toh friend function banane ke bad es ka explanation change ho jayega b3=b1+b2; b3= b1.operator(b2) abhi tak hum yeh bol rahe the but es ke bad

//ab hum bolege + function call hua , or us me c1 and c2 as an argument pass hue

//i.e c3= operator + (c1,c2);

//ab asia kyu hai , ki pahale caller object tha , and + operator i.e binary operator ko apana kam karane ke leye chahiye 2 opearant toh pahale jo caller object tha wo or sath me ek argument pass karana padata tha

//but friend as a non-member function so it has no object , or hame abhi bhi 2 operant chahiye binary operator ko perform karane ke leye

//toh es leye hame 2 argument pass karane padege

b3.show\_data();

}



how to overload unary operator as a friend function :-

#include<iostream>

using namespace std;

class complex

{

private :

int a,b;

public:

void set\_data (int x, int y)

{a=x;b=y;}

void show\_data()

{cout<<"a = "<<a<<"\nb = "<<b<<endl;}

/\* complex operator -() //if we have to define it as a member function

{

complex temp;

temp.a=-a;

temp.b=-b;

return temp;

} \*/

friend complex operator - (complex);

};

complex operator -(complex b1)

{

complex temp;

temp.a=-b1.a;

temp.b = - b1.b;

return temp;

}

int main()

{

complex b1,b2;

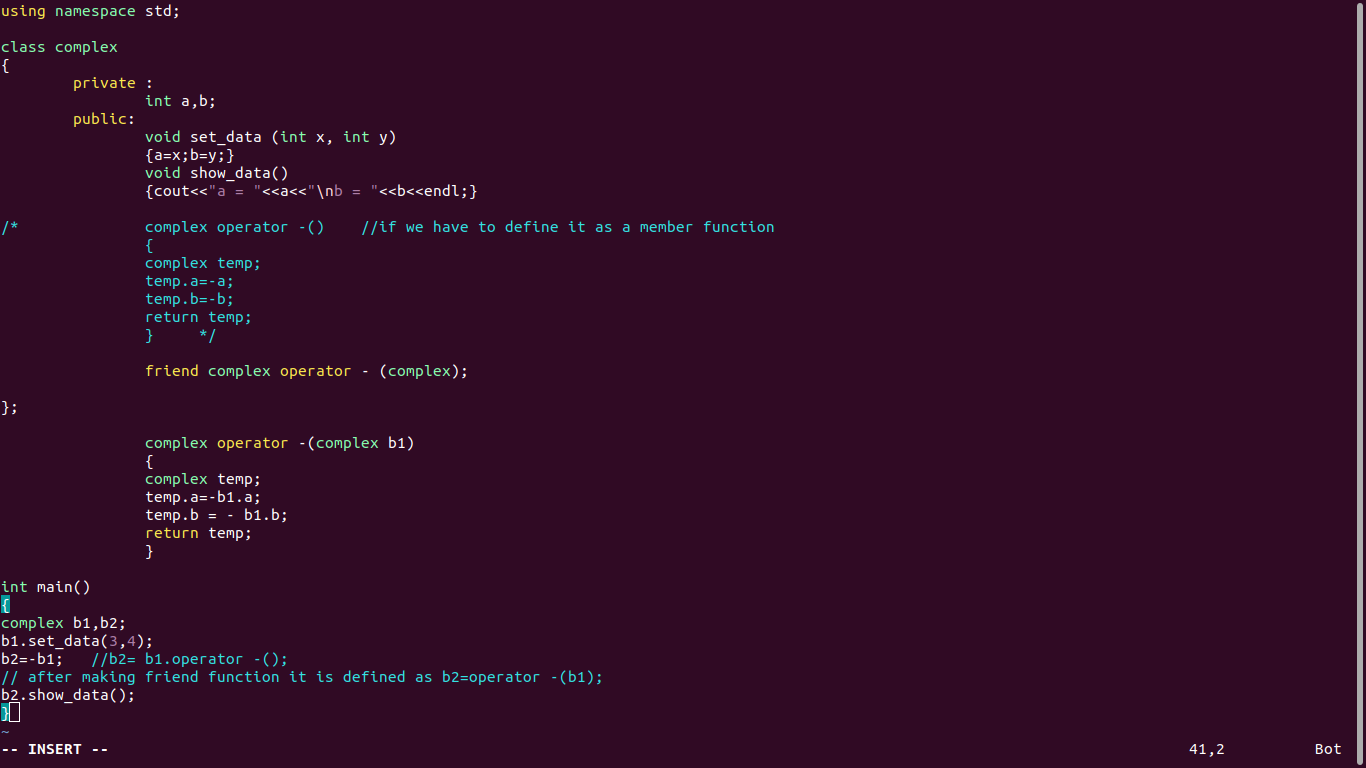
b1.set\_data(3,4);

b2=-b1; //b2= b1.operator -();

// after making friend function it is defined as b2=operator -(b1);

b2.show\_data();

}



now we see insertion and extraction ki overloading friend function ke dawara kaise ki jati hai

#include<iostream>

using namespace std;

class complex

{

private :

int a,b;

public:

void set\_data (int x, int y)

{a=x;b=y;}

void show\_data()

{cout<<"a = "<<a<<"\nb = "<<b<<endl;}

friend ostream & operator <<(ostream & , complex); // yaha ostream cout object ka return type hai i.e output stream

//or as because hum yaha object nhi bana sakate es leye hum reference bana dege yani hum reference le kr karege

friend istream & operator >>(istream & , complex &); // yaha cin ka data type hai istream i.e input stream

// yaha complex ka bhi reference banana padega kyu ki hame actural argument me changes karwane hai

};

ostream& operator <<(ostream & dout, complex c)

{

cout <<"\n a="<<c.a<<"\n b="<<c.b<<endl;

return dout;

}

istream& operator>>(istream & din , complex & d) //here &d is just another name of

{

cin>>d.a>>d.b;

return din; // yaha return sirf ek ke leye karwa rahe hai ki es ki cascating ban sake

}

int main()

{

complex c1;

cout<<"enter a complex no."<<endl;

cin>>c1;

// yeh hame error degi , yeh error kyu degi toh pahale es line ko explain karate hai

// cin kya hai cin ek object hai or >> yeh ek extraction operator es ka member funtion hai

// mtlb cin jis class ke andar define kiya hoga , usi class ke andar hi define kiya hoga >> extraction operator

// es ko hum dusari tarah se bhi likh sakate hai like

// cin.operator >>(c1); or es me c1 as an argument pass hua

// toh jab hum dot nhi lagate . toh hame operator bhi nhi likhana hota or hame parenthesis bhi lagana zaruri nhi hai toh it becomes cin>>c1;

// or ab hame yeh bhi samajhana chahiye ki cin jis class ka object hai us class me extraction operator >> ke ek se jayada operator bane hue hai

//jaise kabhi hai int type ka variable likhate hai, kabhi float type ka toh kabhi char type ka ;

//es ka mtlb extraction operator me pass hone wala argument alag alag hai

//or yeh tabhi munkin hai ki jis class me extraction operator ke ek se jayada version bane hoge or sabhi me arguments me fark hoga

//yeh arguments primitive type ke hisab se bane hoge

//but non primitive type jis hum user define data type bhi kahate hai us ka argument lene ke leye aisa koe version nhi hoga

//toh ese leye yeh line me ayegi error

cout<<"entered value "<<endl;

cout<<c1; //toh yaha bhi wohi bat ho rahi hai , hum c1 pass kar rahe hai insertion operator me

//incresion operator ke bhib bahot sare version bane hoge primitive type ke leye

//but non-primitive type ke leye , aisa koe argumrnt receice karane wala function nhi bana hoga

//toh es ke leye hum friend function bante hai

//cout<<c1; toh es ka mtlb yeh tha ki cout ke class ka object hai jis ne insertion operator ko call kiya or us me ko c1 complex type ki value pass kar raha hai , yani hum le kr chl rahe hai ki jis class ka cout object hai waha pr << increment operator me ek aisa version bhi hoga jis me complex variable bhi define hai, but aisa nhi hai

//toh toh yeh likhana cout.operator <<(c1); yeh galat hai kyu ki aisa koe incremental operator exist nhi karata jo complex value le

//so hum aisa object se us ko access bhi nhi kr sakate

//toh es ke leye hum ek function define karege , ley say hum ek friend function define kr rahe hai

//toh friend function ek non member function hai , jis ke kisi object se access nhi kiya ja sakata

//ab << increment operator kyu ki 2 operant leta hai toh , as because yaha koe object nhi hai toh cout ko bhi as an argument pass karana hoga

//toh hum cout<<c1; ko define aise karege as << increamet operator call hua or usme cout and c1 as an argument pass hua like

//operator<<(cout,c1); toh yeh cout<<c1; ek hi bat hai but yeh valid tabhi hoga jab ese as a friend function define karege

//ab es me cout kyu pass kar rahe hai ki lihane wala ese cout<<c1; waise likh sake

//ab friend function me return type kya hoga toh , cout jis type ka function hai toh wohi return type hona chahiye

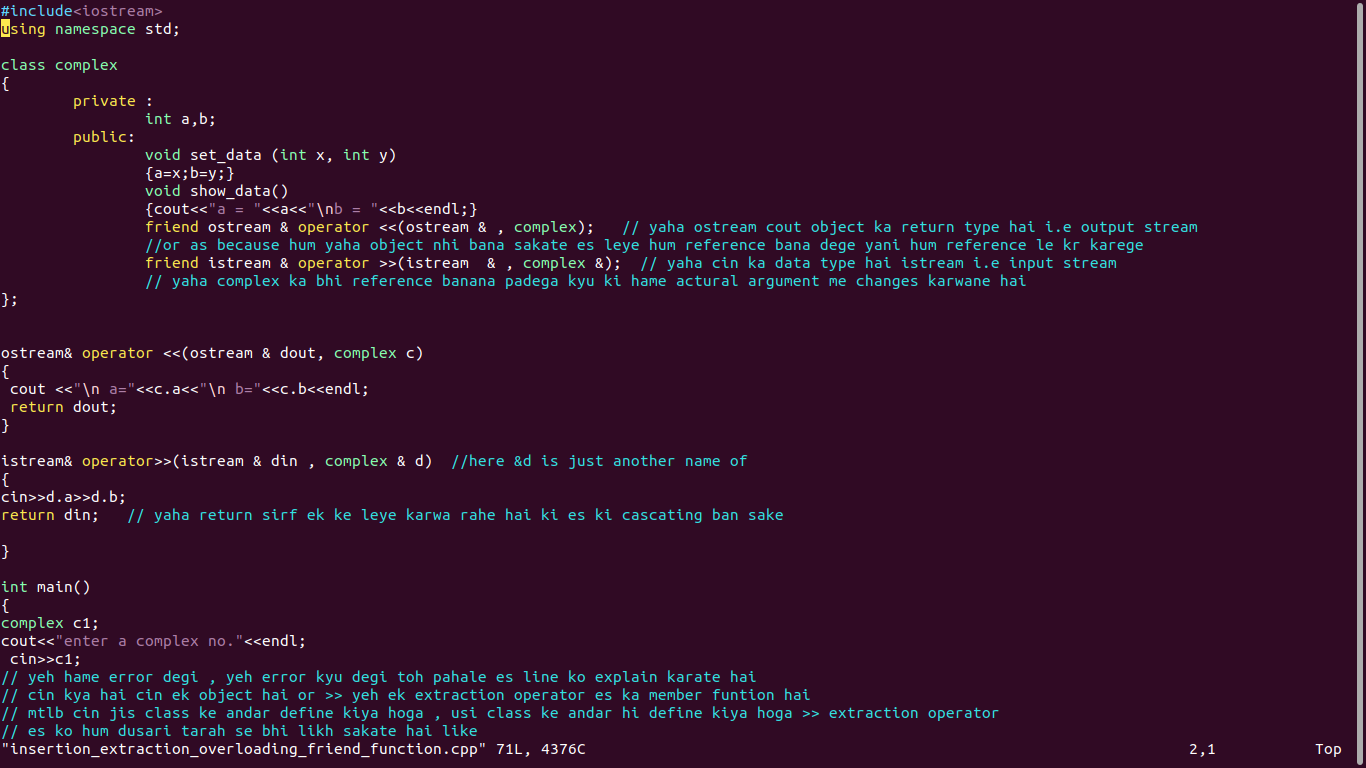
//kyu , kyu ki ho sakata hai hum cascading kare(i.e multiple use of input and output operator in one statement i.e either << or >> )

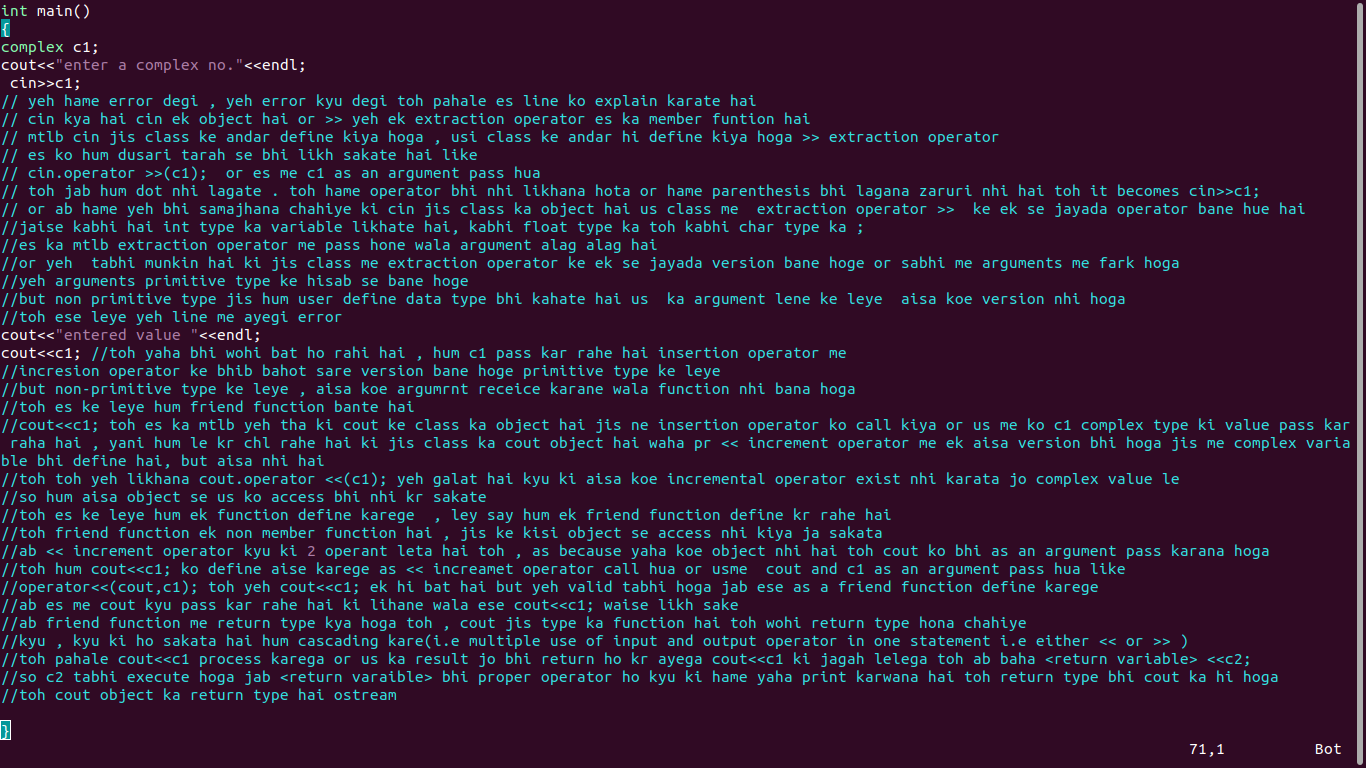
//toh pahale cout<<c1 process karega or us ka result jo bhi return ho kr ayega cout<<c1 ki jagah lelega toh ab baha <return variable> <<c2;

//so c2 tabhi execute hoga jab <return varaible> bhi proper operator ho kyu ki hame yaha print karwana hai toh return type bhi cout ka hi hoga

//toh cout object ka return type hai ostream

}





-> member function of one class can become friend of another class

(ek function kabhi do class ka member ho hi nhi sakata )

#include<iostream>

using namespace std;

class A

{

public :

void fun()

{........}

void food()

{.........}

};

class B

{

friend void A:: fun(); // yeh ek galat decleration hai yeh syantax error nhi hai balaki , lekin ek logical mistake hai

//pr jis tarah se hum ne decleration kiya hai fun() ka , es se yeh toh samajh a raha hai ki fun() class B ka friend hoga

//liken yeh fun() class A ke andar jo hai wo wala fun yeh nhi pata chl raha

//es me aisa bhi toh ho sakata hai ki ek fun() or ho jo kisi bhi class ka member na ho

//ho sakata hai ek or koe class ho or us ke andar bhi fun() function bana ho

//pr es ko dhek ke aisa lag raha hai ki yeh wo fun function hai wo jisi bhi class ka function nhi hai yai yeh jo bahar bana hai

//but hame banana hai jo class A ka fun() hai wo class B ka friend hai

//toh es ko aisa batane ke leye hame scope resolution operator lagana padega <name of class>::<function name>

//yeh scpoe ko resolve kar raha hai jo yeh bata raha hai ki yeh A class wala fun hai

//toh yeh hame likhana padata hai return type or function name ke beech me

//or agar hum yeh nhi lagate hai toh yeh mana jayega ki yeh wo wala fun hai jo kisi bhi class ka function nhi hai

friend void A:: food();

// ab agar ek or hai toh fir se wohi tarika use karege

// but agar hame class A me present sare ke sare funtion ko class B ka friend banana hai

// toh ek ek kr ke bhi bana sakate hai pr es ke leye ek or tarika bhi hai toh hum likhege

friend class A;

// but agar class A me 10 function hai or hame 9 function ko class B me friend banana hai toh hum aisa nhi likhege

// tab hame ek ek kr ke bhi sab ko declare karana padega

};

void fun()

{

}

