# **C++тилинде киргизүү/чыгаруу системи**

## **Си++тилинде киргизүү/чыгаруу системинин негизги түшүнүктөрү**

Си++ тилинде киргизүү/чыгаруу системи агым (stream) аркылуу ишке ашырылат. Програмда колдонгон маалыматты кабыл алган жана чыгарган логикалык түзүлүштү ***киргизүү/чыгаруу агымы*** деп аталат. Агым физикалык түзүлүш менен киргизүү/чыгаруу систем аркылуу байланышат. Киргизүү/чыгаруу системи киргизүү/чыгаруу агымдарга ар кандай түзүлүштөр менен иштөөгө ыңгайлуу интерфейс түзүп берет. Стандарттуу агымдар баскыч менен экрандын ортосундагы байланышты түзүү үчүн колдонот. Бирок ал агымдарды башка түзүлүштөргө жөнөтсө болот. Мисалы, маалыматты экранга чыгарган функцияны маалыматты файл жана принтерге чыгаруу үчүн колдонсо болот.

Си++тилинде програма иштей баштаганда автоматтык түрдө төмөнкү төрт агым ачылат:

|  |  |  |
| --- | --- | --- |
| **Агым** | **Маани** | **Түзүлүш** |
| cin | киргизүү | Баскыч |
| cout | чыгаруу | Экран |
| cerr | ката | Экран |
| clog | cerr агымдын буфердик версиясы | Экран |

Си++тилинде киргизүү/чыгаруу програмдын башына <iostream> бөрк файлын кошуу аркылуу ишке ашырылат. Бул файлда киргизүү/чыгаруу амалдарын ишке ашырган класстардын татаал иерархы аныкталган.

## **Калып менен киргизүү/чыгаруу**

Си++тилинде киргизүү/чыгаруу системи аркылуу маалыматтарды ар кандай калыпта чыгарса болот. Андан сырткары, маалыматты киргизүүдө киргизүүнүн кээ бир параметрлерин өзгөртсө болот.

Киргизүү/чыгаруунун ар бир агымы бир катар калып байрактары (*format flags*) менен байланышта турат. Калып байрактары маалыматтарды калыптоо ыкмасын башкарат жана биттик маскалардан (*bitmasks*) турат. Бул маскалар *ios* классында *fmtflags* аталуучу тип катары жарыяланган. Анда төмөнкү маанилер аныкталган:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| skipws | left | right | internal | oct | hex |
| dec | showbase | scientific | fixed | uppercase | showpos |
| showpoint | unitbuf | boolalpha | basefiels | adjustfield | floatfield |

Бул маанилер *ios* классында аныкталган жана калып туусун орнотуу же алып таштоо үчүн колдонулат.

Маалыматты киргизүү учурунда *skipws* туусу орнотулса, анда башында жайгашкан көрүнбөгөн белгилер (боштуктар, табулөө жана жаңы сап белгилери) жокко чыгарылат. Эгерде *skipws* туусу алынып ташталса, анда көрүнбөгөн белгилер жокко чыгарылбайт.

*left* туусу орнотулган кезде тегиздөө сол жак бойунча жүргүзүлөт.*right*туусу орнотулган кезде тегиздөө оң жак бойунча жүргүзүлөт. Эгерде *internal* туусу орнотулса, анда чыгаруу талааны толтуруу үчүн санарип жана сандын белгисинин ортосуна боштук белгилер кыстарылат. Бул байрактардын бири да орнотулбаса, анда тегиздөө сол жагы бойунча жүргүзүлөт.

Эч кандай туу орнотулбаса сандар ондук эсептөө системинде чыгат. *oct* туусу сегиздик, *hex* туусу он алтылык эсептөө системинде чыгарат. Ондук эсептөө системине кайтып келүү үчүн *dec* туусу орнотулат.

*showbase* туусу аркылуу санды эсептөө системинин негизи менен чыгарса болот. Мисалы, он алтылык эсептөө системинде берлиген *1F* саны *Ох1F* калыпта чыгат.

*scientific* туусу орнотулган кезде анык сандар илимий жазылышта чыгат, ал эми *fixed* туусу орнотулса анык сандар кадимки жазылышта чыгат. Эгерде бул байрактардын бириси да орнотулбаса, анда чыгаруунун ыңгайлуу калыбын компилегич өзү тандап алат.

Илимий жазылышта чыгаруу учурунда «е» белгиси төмөнкү регистрде чыгат. Ошондой эле он алтылык системде чыгаруу учурунда «х» белгиси да төмөнкү регистрде чыгат. *uppercase* туусун орнотсок, анда ал белгилер жогорку регистрде чыгат.

*showpos* туусу орнотулган учурда оң сандын алдына «+» белгиси койулат.

*showpoint* туусу анык сандарды чыгаруу учурунда ондук чекит жана 0 белгисин чыгарууга алып келет.

*unitbuf* туусу орнотулса, анда ар бир кыстаруу амалынан кийин буфер тазаланат.

*boolalpha* туусу орнотулса, анда логик типтин маанилери *true* жана *false*кызматчы сөз менен чыгарылат.

*oct***,***dec* жана *hex* байрактары менен аныкталган талааларга бир мезгилде *basefiels* туусу аркылуу кайрылса болот. Ошондой эле *left***,***right*жана *internal* байрактары менен аныкталган талааларга *adjustfield* туусу менен шилтеп салса болот. Ал эми *scientific* жана *fixed* байрактары менен аныкталган талааларга *floatfield* туусу аркылуу кайрылса болот.

Калыптын туусун орнотуу үчүн *setf()* функциясы колдонот. Ал *ios* классынын мүчөсү болот. Төмөндө анын негизги формасы көрсөтүлгөн:

fmtflags setf(fmfcflags байрактар);

Бул функция алдыда орнотулган калып байрактардын көрсөтмөлөрүн кайтарып берет жана жаңыларды орнотот. Мисалы, *showpos* туусун орнотуу үчүн төмөнкү көрсөтмөнү колдонсо болот:

киргизүү/чыгаруу\_агымы.setf(ios::showpos);

Мында*киргизүү/чыгаруу\_агымы*биз таасир берип жаткан агым.

*setf()*функцияга бир кайрылуу аркылуу бир нече тууну орнотсо болот. Керектүү болгон байрактарды бириктирүү үчүн *OR* операторун колдонуш керек. Төмөнкү кайрылууда *cout* агымы үчүн *showbase* жана *hex* байрактары орнотулат:

cout.setf(ios::showbase | ois::hex);

Төмөндө калыптын бир нече байрактарын койуга мисал келтирилген:

#include <iostream>  
using namespace std;  
  
int main()  
{  
cout<<158.43<<" Салaм "<<1000<<"\n";  
cout<<25<<' '<<-25<<"\n";  
cout<<150.0<<"\n";  
cout.unsetf(ios::dec);  
cout.setf(ios::hex | ios::scientific);  
cout<<158.43<<" Салaм "<<1000<<"\n";  
cout.setf(ios::showpos);  
cout<<25<<' '<<-25<<"\n";  
cout.setf(ios::showpoint | ios::fixed);  
cout<<150.0<<"\n";  
return 0;  
}

Програма төмөнкү жыйынтыкты берет:

158.43 Салaм 1000  
25 -25  
150  
1.584300e+02 Салaм 3e8  
19 ffffffe7  
+150.000

*unsetf()* функция аркылуу биз *dec* туусун алып таштадык.

Төмөнкү мисалда *uppercase* туусунун таасири көрсөтүлгөн. Алды менен *uppercase*, *showbase* жана *hex* байрактары орнотулат дагы, 112 саны он алтылык эсептөө системинде чыгарылат. Жыйынтыкта он алтылык эсептөө системинин белгиси «Х» жогорку регистрде чыгат. Андан кийин *unsetf()* функция аркылуу *uppercase* туусу алынып ташталат дагы «х» белгиси төмөнкү регистрде чыгат.

#include <iostream>  
using namespace std;  
  
int main()  
{  
cout.unsetf(ios::dec);  
cout.setf(ios::uppercase | ios:: showbase | ios::hex);  
cout<<112<<"\n";  
cout.unsetf(ios::uppercase);  
cout<<112<<"\n";  
return 0;  
}

Жыйынтык төмөнкү:

0X70  
0x70

Төмөнкү програмда кээ бир калып байрактарынын абалдарын билүү үчүн *flags()* функциясы колдонулган. *tuu\_korsot()* функциясына көңүл буруш керек, себеби ал програмдарды иштеп чыгууда керек болуп калышы мүмкүн.

#include <iostream>  
using namespace std;  
void tuu\_korsot();  
  
int main()  
{  
    tuu\_korsot();//байрактардын абалдарын чыгаруу  
    cout.setf(ios::oct | ios::showbase | ios::fixed);  
    tuu\_korsot();  
    return 0;  
}  
  
//Төмөнкү функция калыптын байрактарынын абалын көрсөтөт  
void tuu\_korsot()  
{  
    ios::fmtflags tuu;  
    tuu =cout.flags();//байрактардын калыбын алуу  
    if(tuu & ios::skipws) cout<<"skipws орнотулган \n";  
        else cout<<"skipws алынган \n";  
    if(tuu & ios::left) cout<<"left орнотулган \n";  
        else cout<<"left алынган \n";  
    if(tuu & ios::right) cout<<"right орнотулган \n";  
        else cout<<"right алынган \n";  
    if(tuu & ios::internal) cout<<"internal орнотулган \n";  
        else cout<<"internal алынган \n";  
    if(tuu & ios::oct) cout<<"oct орнотулган \n";  
        else cout<<"oct алынган \n";  
    if(tuu & ios::fixed) cout<<"fixed орнотулган \n";  
        else cout<<"fixed алынган \n";  
         
       cout<<"\n";  
    };

Бул програмдан төмөнкүдөй натыйжа алабыз:

skipws орнотулган  
left алынган  
right алынган  
internal алынган  
oct алынган  
fixed алынган  
  
skipws орнотулган  
left алынган  
right алынган  
internal алынган  
oct орнотулган  
fixed орнотулган

Төмөнкү програмда *flags()* функциясынын экинчи варианты каралган. Алды менен *showpos***,***showbase, oct*жана*right* байрактарын орнотуп, биз байрактардын маскасын түзөбүз. Андан кийин *flags()* функциясынын жардамы менен *cout* агымы үчүн байрактардын өзгөрмөсү маска менен байланыштырылат. *tuu\_korsot()* функциянын жардамы менен байрактардын абалдары текшерилет.

#include <iostream>  
using namespace std;  
void tuu\_korsot();  
  
int main()  
{  
    tuu\_korsot();//байрактардын абалдарын чыгаруу  
    ios::fmtflags tuu=ios::showpos | ios::showbase | ios::oct | ios::right;  
   cout.flags(tuu);  
    tuu\_korsot();  
    return 0;  
}  
  
//Төмөнкү функция калыптын байрактарынын абалын көрсөтөт  
void tuu\_korsot()  
{  
    ios::fmtflags tuu;  
    tuu=cout.flags();//байрактардын калыбын алуу  
    if(tuu & ios::showpos) cout<<"showpos орнотулган \n";  
        else cout<<"showpos алынган \n";  
    if(tuu & ios::left) cout<<"left орнотулган \n";  
        else cout<<"left алынган \n";  
    if(tuu & ios::showbase) cout<<"showbase орнотулган \n";  
        else cout<<"showbase алынган \n";  
    if(tuu & ios::oct) cout<<"oct орнотулган \n";  
        else cout<<"oct алынган \n";  
    if(tuu & ios::right) cout<<"right орнотулган \n";  
        else cout<<"right алынган \n";  
    if(tuu & ios::fixed) cout<<"fixed орнотулган \n";  
        else cout<<"fixed алынган \n";  
         
        cout<<"\n";  
    };

Програм төмөнкү жыйынтыкты берет:

showpos алынган  
left алынган  
showbase алынган  
oct алынган  
right алынган  
fixed алынган  
  
showpos орнотулган  
left алынган  
showbase орнотулган  
oct орнотулган  
right орнотулган  
fixed алынган

## **width(), precision() жана fill() функциялары**

Калып байрактарынан сырткары *ios* классында үч мүчө-функция аныкталган. Бул функциялар калыптын төмөнкү параметрлерин аныктайт: талаанын узундугу, тактык жана толуктоо белгиси. Ирети менен бул *width(), precision()*жана*fill()*функциялары.

Унчукпаганда чыгарылып жаткан маани үчүн белгилердин санына жараша орун берилет. Бирок *width()* функциясы аркылуу ал узундукту өзгөртсө болот. Төмөндө ал функциянын прототиби көрсөтүлгөн:

streamsize width(streamsize w);

Талаанын узундугу *w* параметр аркылуу берилди.

Эгерде талаанын узундугу орнотулгандан кийин чыгарылуучу маани ал узундуктан кыска болсо, анда калган жер учурдагы толуктоо белгиси менен толукталат. Унчукпаганда ал белги катары боштук белгиси алынат. Эгерде чыгарылып жаткан маанинин узундугу орнотулган талаанын узундугунан узун болсо, анда маани аныкталган узундукта кесилип чыгат.

Анык сандарды чыгарууда унчукпаганда бөлчөк бөлүгүнө алты орун берилет. *precision()*функциясынын жардамы менен ал санды өзгөртсө болот. Төмөндө ал функциянын прототиби көрсөтүлгөн:

streamsize precision(streamsize p);

Тактык *р* параметри аркылуу берилет.

Эгерде талаанын бош жерлерин кандайдыр белги менен толуктоого туура келсе, анда *fill()*функциясы колдонот. Ал функциянын прототиби төмөнкүчө аныкталат:

char fiil(char belgi);

*fiil()*функциясы чакырылгандан кийин *belgi*белгиси толуктоонун жаңы белгиси болот.

Төмөнкү програм калыптоо функцияларын сүрөттөйт.

#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout.width(10);  
    cout<<"Салам"<<’\n’;  
    cout.fill('$');  
    cout.width(10);  
    cout<<"Салам"<<’\n’;  
    cout.setf(ios::left);  
    cout.width(10);  
    cout<<"Салам"<<’\n’;  
    cout.width(10);  
    cout.precision(10);  
    cout<<123.456789<<’\n’;  
    cout.width(10);  
    cout.precision(5);  
    cout<<123.456789<<’\n’;  
    return 0;  
}

Програма аткарылып бүткөндөн кийин экранга төмөнкү чыгат:

|  |
| --- |
| Салам |
| $Салам |
| Салам  $ |
| 123.456789 |
| 123.46 |

Талаанын узундугу ар бир чыгаруунун алдында берилип жатканына көңүл буруш керек.

Төмөнкү програмда Си++ тилинде киргизүү/чыгаруунун калыбына байрактарды орнотуу функцияларды колдонуу аркылуу сандардын тегизделген жадыбалын кандайча алууга мүмкүндүк бар экендиги көрсөтүлгөн.

#include <iostream>  
#include <cmath>  
using namespace std;  
  
int main()  
{  
    double x;  
    cout.precision(4);  
    cout<<"      x sqrt(x)    x^2 \n";  
    for(x=2.0;x<=10.0;x++)  
    {cout.width(7);  
        cout<<x;  
        cout.width(7);  
        cout<<sqrt(x);  
        cout.width(7);  
        cout<<x\*x;  
        cout<<"\n";  
    }  
    return 0;  
}

Програма төмөнкү натыйжаны берет:

|  |  |  |
| --- | --- | --- |
| x | sqrt(x) | x^2 |
| 2 | 1.414 | 4 |
| 3 | 1.732 | 9 |
| 4 | 2 | 15 |
| 5 | 2.236 | 25 |
| 6 | 2.449 | 36 |
| 7 | 2.646 | 49 |
| 8 | 2.828 | 64 |
| 9 | 3 | 81 |
| 10 | 3.162 | 100 |

## **Киргизүү/чыгаруу манипуляторлору**

Си++ тилинде киргизүү/чыгаруу системинде маалыматты калыптоонун дагы бир ыкмасы бар. Ал ыкма атайын киргизүү/чыгаруу манипуляторлор функцияларын колдонууну болжолдойт.

Киргизүү/чыгаруу манипуляторлору киргизүү/чыгаруунун атайын функциялары болуп саналат жана *ios* классынын мүчөлөрүнөн айырмаланып, киргизүү/чыгаруу көрсөтмөлөрдүн ичинде жайгашышы мүмкүн. Эгерде програмда параметрлүү манипулятор колдонсо (мисалы *setw()*функциясы), анда програмдын башына *iomanip* бөрк файлын кошуу зарыл. Манипуляторлорду киргизүү/чыгаруу амалдардын ичинде көрсөтсө болот. Мисалы,

cout<<oct<<100<<hex<<100;  
cout<<setw(6)<<100;

Бул мисалда биринчи көрсөтмө *cout* агымына санды сегиздик эсептөө системинде, андан кийин он алтылык эсептөө системинде чыгарууга буйрук берет. Экинчи көрсөтмөдө сан үчүн алты орун бөлөнүп, берилген сан он алтылык эсептөө системинде чыгат.

Төмөнкү жадыбалда стандарттуу манипуляторлордун тизмеси келтирилген:

|  |  |  |
| --- | --- | --- |
| **Манипулятор** | **Милдети** | **Киргизүү/чыгаруу** |
| boolalpha | boolalpha тууну орнотуу | киргизүү/чыгаруу |
| dec | dec туусун орнотуу | киргизүү/чыгаруу |
| endl | жаңы сапка өтүү | чыгаруу |
| ends | Null маанини чыгаруу | чыгаруу |
| fixed | fixed туусун орнотуу | чыгаруу |
| flush | агымды тазалоо | чыгаруу |
| hex | hex туусун орнотуу | киргизүү/чыгаруу |
| internal | internal туусун орнотуу | чыгаруу |
| left | left туусун орнотуу | чыгаруу |
| noboolalpha | boolalpha туусун алып таштоо | киргизүү/чыгаруу |
| noshowbase | showbase туусун алып таштоо | чыгаруу |
| noshowpoint | showpoint туусун алып таштоо | чыгаруу |
| noshowpos | showpos туусун алып таштоо | чыгаруу |
| noskipws | skipws туусун алып таштоо | киргизүү |
| nounitbuf | unitbuf туусун алып таштоо | чыгаруу |
| nouppercase | uppercase туусун алып таштоо | чыгаруу |
| oct | oct туусун орнотуу | киргизүү/чыгаруу |
| resetiosflags(fmtflags t) | t параметри менен берилген байрактарды жойу | киргизүү/чыгаруу |
| right | right туусун орнотуу | чыгаруу |
| scientific | scientific туусун орнотуу | чыгаруу |
| setbase(int негиз) | эсептөөнүн негизин берүү | киргизүү/чыгаруу |
| setfill(int белги) | Толуктоонун белгисин аныктоо | чыгаруу |
| setiosflags(fmtflags t) | t параметри менен берилген байрактарды аныктоо | киргизүү/чыгаруу |
| setprecision(int p) | тактыкты аныктоо | чыгаруу |
| setw(int w) | талаанын узундугун аныктоо | чыгаруу |
| showbase | showbase туусун орнотуу | чыгаруу |
| showpoint | showpoint туусун орнотуу | чыгаруу |
| showpos | showpos туусун орнотуу | чыгаруу |
| skipws | skipws туусун орнотуу | киргизүү |
| unitbuf | unitbuf туусун орнотуу | чыгаруу |
| uppercase | uppercase туусун орнотуу | чыгаруу |
| ws | баштапкы боштуктарды жойу | киргизүү |

Төмөнкү програмда киргизүү/чыгаруу манипуляторлорго мисал келтирилген.

#include <iostream>  
#include <iomanip>  
using namespace std;  
  
int main()  
{  
cout<<oct<<100<<"  "<<hex<<100;  
cout<<setw(5)<<100<<endl;  
cout<<setfill('$')<<setw(8);  
cout<<100<<"Салам"<<endl;  
    return 0;  
}

Програм аткарылып бүткөндөн кийин төмөнкү жыйынтык чыгат:

144  64   64

$$64Салам

Төмөнкү програмда 2ден 10 чейин сандардын квадраттык тамыры жана квадраты тегизделген жадыбал түрүндө чыгаруу маселеси каралган. Бул програмда мүчө-функциялардын жана калыптын байрактарынын ордуна киргизүү/чыгаруу манипуляторлору колдонду.

#include <iostream>  
#include <iomanip>  
#include <cmath>  
using namespace std;  
  
int main()  
{  
double x;  
    cout<<setprecision(4);  
    cout<<"      x  sqrt(x)   x^2 \n";  
    for(x=2.0;x<=10.0;x++)  
    {  
        cout<<setw(7)<<x;  
        cout<<setw(7)<<sqrt(x);  
        cout<<setw(7)<<x\*x<<"\n";  
    }  
    return 0;  
}

Жыйынтык төмөнкү калыпта чыгат:

|  |  |  |
| --- | --- | --- |
| x | sqrt(x) | x^2 |
| 2 | 1.414 | 4 |
| 3 | 1.732 | 9 |
| 4 | 2 | 15 |
| 5 | 2.236 | 25 |
| 6 | 2.449 | 36 |
| 7 | 2.646 | 49 |
| 8 | 2.828 | 64 |
| 9 | 3 | 81 |
| 10 | 3.162 | 100 |

Киргизүү/чыгаруунун калып байрактарынын эң кызыктуусу *boolalpha* туусу саналат. Ал логик типти киригизүү/чыгарууда 1 жана 0 маанилердин ордуна *true* жана *false* кызматчы сөздөрүн колдонууга мүмкүндүк берет. Төмөнкү мисалда ал ачык-айкын көрсөтүлгөн.

#include <iostream>  
using namespace std;  
  
int main()  
{  
bool b;  
cout<<"boolalpha туусу койула элек: ";  
b=true;  
cout<<b<<" ";  
b=false;  
cout<<b<<"\n";  
cout<<"boolalpha туусу койулду: ";  
b=true;  
cout<<boolalpha<<b<<" ";  
b=false;  
cout<<b<<" "<<endl;  
cout<<"Логик типтин маанисин киргиз: ";  
cin>>boolalpha>>b;//бул жерде true же false киргизсе болот  
cout<<"\n Киргизилген маани: "<<b;  
return 0;  
}

Жыйынтык төмөнкүдөй болушу мүмкүн:

boolalpha туусу койула элек: 1 0  
boolalpha туусу койулду: true false  
Логик типтин маанисин киргиз: true  
Киргизилген маани: true

## **Чыгаруунун колдонуучу функциялары**

Си++тилинде чыгарууну кээде *кыстаруу (insertion)*, ал эми **<<** операторун ***кыстаруу оператору*** *(insertion operator*) деп аташат. Маалыматты чыгаруу үчүн **<<** операторун колдонгондо, ***кыстаруу функция*** (*inserter function же inserter*) түзүлөт. Бул термин маалыматты агымга кыстарат дегенди түшүндүрөт. Терминдердин мындай баш аламандыгынан кутулуу үчүн кыстаруу функцияны *чыгаруунун* *колдонуучу функциясы*деп аталат.

Бардык чыгаруунун колдонуучу функцияларында төмөнкү негизги калып колдонулат:

ostream & operator<<(ostream &stream, класстын\_ысымы объект)  
{  
//Чыгаруунун колдонуучу функциясынын тулкусу  
return stream;  
}

Биринчи аргумент *ostream* тибиндеги объектке шилтеме болот, ошондуктан *stream* агымы чыгаруучу агым болуш керек. Экинчи параметр чыгарылуучу объекти кабыл алат. Эгерде кайрадан жүктөлүүчү << оператору киргизүү/чыгаруу туйунтмасында удаалаш колдонсо, анда чыгаруунун колдонуучу функциясы *ostream* тибиндеги *stream* агымга шилтемени кайтарып бериши зарыл:

cout<<objec1<< objec2<< objec3;

Чыгаруунун колдонуучу функциясынын ичинде каалагандай процедурду аткарса болот. Ал функциянын аткарган амалдары толугу менен колдонуучудан көз каранды. Бирок, кандайдыр бир класс менен иштөөгө багытталган чыгаруучу функция ал класстын мүчөсү боло албайт. Анын себеби төмөнкүдө: эгерде каалаган типтеги оператор-функция класстын мүчөсү болсо, анда сол операнды айкын эмес *this* көрсөткүчү аркылуу берилет дагы, оператор-функцияны чакыруусун козгогон объект болот. Сол операнд ал класстын объеги болушун шарттайт. Ошондуктан кайрадан-жүктөлгөн оператор-функция класстын мүчөсү болсо, сол операнд ошол класстын мүчөсү болуш керек. Бирок, чыгаруунун колдонуучу функциясын колдонгондо сол операнд класстын объеги болбой агымга айланат, ал эми оң операнд чыгарууга зарыл болгон объектке айланат. Ушул себептен чыгаруунун колдонуучу функциясы мүчө-функция боло албайт. Бирок андай функциялар класска шериктеш функция болушу мүмкүн.

Төмөнкү мисалда класс үчүн чыгаруунун колдонуучу функциясы түзүлөт.

//объектилерди чыгаруу үчүн шериктеш функцияларды колдонуу  
#include <iostream>  
using namespace std;  
  
class misal{  
int i,j;  
public:  
misal(){i=0; j=0;}  
misal(int x, int y){i=x; j=y;}  
friend ostream &operator<<(ostream &stream, misal object);  
};  
  
ostream &operator<<(ostream &stream, misal object)  
{stream<<object.i<<","<<object.j<<"\n";  
return stream;  
}  
  
int main()  
{  
misal a(3,5), b(10,15);  
cout<<a<<b;  
return 0;  
}

Програм төмөнкү жыйынтыкты берет:

3,5  
10,15

Төмөнкү програмда чыгаруунун колдонуучу функциясы класска шериктеш функция болбойт. Чыгаруунун колдонуучу функциясы класстын жабык бөлүгүнө кирүүгө мүмкүндүгү жок болгондуктан, өзгөрмөлөрдү ачык кылып жарыялайбыз.

/\*объектилерди чыгаруу үчүн шериктеш эмес функцияларды колдонуу\*/  
#include <iostream>  
using namespace std;  
  
class misal{  
public:  
int i,j;  
misal(){i=0; j=0;}  
misal(int x, int y){i=x; j=y;}  
};  
  
ostream &operator<<(ostream &stream, misal object)  
{stream<<object.i<<","<<object.j<<"\n";  
return stream;  
}  
  
int main()  
{  
misal a(3,5), b(10,15);  
cout<<a<<b;  
return 0;  
}

Програм аткарылганда төмөнкү жыйынтык алынат:

3,5  
10,15

## **Киргизүүнүн колдонуучу функциялары**

Чыгаруучу операторду кайрадан жүктөгөндөй эле, киргизүүчү операторду кайрадан жүктөсө болот. Си++тилинде киргизүүчү операторду кээде *сууруп алуу оператору*, ал эми ал операторду кайрадан жүктөгөн функцияны ***сууруп алуу функциясы*** деп аталат. Төмөндө киргизүүнүн колдонуучу функциясынын жалпы калыбы келтирилген:

istream &operator(istream &stream, класстын\_ысымы &объект)  
{  
//киргизүүнүн колдонуучу функциясынын денеси  
return stream;  
}

Киргизүүнүн колдонуучу функциялары шилтемени киргизүүлүчү *istream*агымына кайтарып берет. Биринчи параметри киргизүүлүчү агымга шилтеме болот, экинчи параметри киргизилүүчү маалыматты алган объектке шилтеме болот. Киргизүүнүн колдонуучу функциясы мүчө-функция боло албайт.

Төмөнкү програмда киргизүүнүн колдонуучу функциясы кошулган.

#include <iostream>  
using namespace std;  
  
class misal{  
int i,j;  
public:  
misal(){i=0; j=0;}  
misal(int x, int y){i=x; j=y;}  
friend ostream &operator<<(ostream &stream, misal object);  
friend istream &operator>>(istream &stream, misal &object);  
};  
  
ostream &operator<<(ostream &stream, misal object)  
{stream<<object.i<<","<<object.j<<"\n";  
return stream;  
}  
  
istream &operator>>(istream &stream, misal &object)  
{  
cout<<" Эки сан киргиз: ";  
stream>>object.i>>object.j;  
return stream;  
}  
  
int main()  
{  
misal a(3,5), b(10,15);  
cout<<a<<b;  
cin>>a;  
cout<<a;  
 return 0;  
}

Програм төмөнкү жыйынтыктарды берет:

3,5  
10,15  
Эки сан киргиз: 15 20  
15,20

Төмөнкү мисалда буйумдун атын, санын жана бир буйумдун баасын сактаган класс түзүлөт. Програмда киргизүү жана чыгаруунун колдонуучу функциялары кошулган.

#include <iostream>  
#include <cstring>  
using namespace std;  
  
class buyum{  
char ati[25];//Буйумдун аты  
int sani; //Колдогу буйумдун саны  
double baasi; // Буйумдун баасы  
public:  
buyum (char \* a, int s, double b)  
{  
    strcpy(ati,a);  
    sani=s;  
    baasi=b;  
}  
friend ostream &operator<<(ostream &stream, buyum object);  
friend istream &operator>>(istream &stream, buyum &object);  
};  
  
ostream &operator<<(ostream &stream, buyum object)  
{stream<<object.ati<<" "<<object.sani<<" ";  
stream<<object.baasi<<"\n";  
return stream;  
}  
  
istream &operator>>(istream &stream, buyum &object)  
{  
cout<<" Буйумдун атын киргиз: ";  
stream>>object.ati;  
cout<<" Буйумдун санын киргиз: ";  
stream>>object.sani;  
cout<<" Буйумдун баасын киргиз: ";  
stream>>object.baasi;  
return stream;  
}  
  
int main()  
{  
buyum object("Калем",3,1.5);  
cout<<object;  
cin>>object;

cout<<object;  
return 0;  
}

Програм төмөнкү натыйжаларды берет:

Калем 3 1.5  
 Буйумдун атын киргиз: Дептер  
 Буйумдун санын киргиз: 5  
 Буйумдун баасын киргиз: 2.35  
Дептер 5 2.35