1 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

|  |
| --- |
| **BÀI 3: ĐỊNH NGHĨA CHỒNG** (hàm và) **PHÉP TOÁN: Overload Operator** |

**1. Định nghĩa chồng phép toán**

Những phép toán có sẵn được định nghĩa lại để có thể áp dụng với những kiểu do ta xây dựng gọi là định  nghĩa chồng phép toán. Định nghĩa chồng phép toán làm cho chương trình dễ đọc, dễ sửa lỗi.

**Ví dụ: Tìm chỗ sai trong chương trình sau**

struct phanso

{

int ts,ms;

};

int main()

{

phanso ps1, ps2, ps3;

cout<<”nhap ps1, ps2:”;

cin>>ps1>>ps2;

ps3=ps1+ps2;

cout<<”tong la:”<<ps3;

}

cin>>ps1>>ps2; //sai vì phép toán nhập (>>) chỉ cho nhập kiểu số, chuỗi;

 không thể nhập kiểu phanso (struct)

ps3=ps1+ps2; //sai vì phép toán cộng chỉ dùng để cộng hai số,

 không thể cộng hai phân số (struct)

cout<<”tong la:”<<ps3; //sai vì phép toán xuất (<<) chỉ dùng để xuất kiểu số, chuỗi;  không thể xuất kiểu phanso (struct)

Để có thể sử dụng các phép toán nhập (cin), cộng (+), xuất (cout) với kiểu phân số, C++ cho phép ta  định nghĩa lại các phép toán này.

**2. Các loại phép toán và cách cài đặt**

**Các loại phép toán:**

Có hai loại phép toán là phép toán hai ngôi (có hai toán hạng) và phép toán một ngôi (có một toán hạng). **ví dụ:**

- Phép toán số học, phép gán kép (+=,-=,…), phép so sánh (>,>=,…) là phép toán hai ngôi:   VD: a+b, a-b, a\*b, a/b, a+=b, a<b, a-=b, a\*=b, a/=b, a%=b, a>b, a>=b, a<=b, a==b,

- Phép ++, --, - (phép toán đổi dấu), ! (phép toán phủ định) là phép toán một ngôi:  VD: ++a, --b, -c, !(a>b)

Thuật ngữ:

Operator = toán tử, VD + - \* / ++ --

Operand = toán hạng, VD: 3 + 5 => 3 và 5: toán hạng, + toán tử

 a += b; <=> a = a + b;

**Cách cài đặt lại phép toán**

Viết phép toán ở dạng phương thức thuộc lớp

Cú pháp: giống cú pháp ở cách 1 nhưng ít hơn một tham số.

Nghĩa là nếu phép toán hai ngôi thì số tham số là một, nếu phép toán một ngôi thì số tham số sẽ là 0. VD:

a PhepToan b => a.PhepToan(b);

PhepToan a => a.PhepToan();

2 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

\* Ví dụ: Xây dựng lớp phân số và định nghĩa lại (chồng phép toán) đối với các phép toán sau: - **Phép toán hai ngôi:**

**+, += (phép gán kép), >(phép so sánh), << (phép toán “xuất”), >> (phép toán “nhập”)** - **Phép toán 1 ngôi: ! (phép nghịch đảo ps), ++(phép toán tăng 1), --(giảm 1)**

#include <iostream>

using namespace std;

class PHANSO

{

 int tuso, mauso;

public:

 PHANSO(int t =0, int m =1){ tuso =t; mauso =m; } //hàm thiết lập

 int USCLN();

 void ToiGian();

 //4 phep toan 2 ngoi: + - \* /

 PHANSO& operator+(PHANSO& p);

 PHANSO& operator-(PHANSO& p);

 PHANSO& operator\*(PHANSO& p);

 PHANSO& operator/(PHANSO& p); //chu y: Xu ly truong hop MauSo cua Kq co the = 0

 //4 phep toan 1 ngoi: ++ -- - !

 PHANSO& operator++();

 PHANSO& operator--();

 PHANSO& operator-();

 PHANSO& operator!(); // Nghich dao PhanSo; Xu ly: truong hop Mauso cua KQ co the =0

 //4 phep toan GOP/GHEP [2 ngoi]: += -= \*= /=

 PHANSO& operator+=(PHANSO& p);

 PHANSO& operator-=(PHANSO& p);

 PHANSO& operator\*=(PHANSO& p);

 PHANSO& operator/=(PHANSO& p); //chu y: Xu ly truong hop MauSo cua Kq co the = 0

 //6 phep toan SO SANH [2 ngoi]: > < >= <= == != : KetQua là int theo nghia TRUE: 1, FALSE: 0  int operator>(PHANSO& p);

 int operator<(PHANSO& p);

 int operator>=(PHANSO& p);

 int operator<=(PHANSO& p);

 int operator==(PHANSO& p);

 int operator!=(PHANSO& p);

 //2 ham friend : nhap (tham chieu istream&) /xuat(tham chieu ostream&) PhanSo [2 ngoi: ngoi dau la thu vien: i/o:stream]  friend istream& operator>>(istream& i, PHANSO& p); //i chinh la cin: cin>>p:Ham toan tu tu do (ko thuoc lop)  friend ostream& operator<<(ostream& o, PHANSO& p); //o chinh la cout: cout<<p: Ham toan tu tu do (ko thuoc lop)

};

3 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

int PHANSO::USCLN()

{ int a = tuso, b = mauso, t;

while(b!=0) //thuc hien cho den khi mau so b = 0

{

t = a; //giu lai tu so a vao bien tam t

a = b; // tu so a = mau so b

b = t%b; // mau so b = du so % cua phep chia tu so kq t cho mau so b

}

return a; //ket qua la tu so sau cung a khi chay vong lap tren (while) cho den khi mau so b = 0 }

void PHANSO::ToiGian()

{ int t = USCLN();

tuso = tuso/t;

mauso = mauso/t;

}

//4 phep toan 2 ngoi: + - \* /

PHANSO& PHANSO::operator+(PHANSO& p)

{

tuso = tuso\*p.mauso + p.tuso\*mauso;

mauso = mauso\*p.mauso;

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator-(PHANSO& p)

{

tuso = tuso\*p.mauso - p.tuso\*mauso;

mauso = mauso\*p.mauso;

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator\*(PHANSO& p)

{

tuso = tuso\*p.tuso;

mauso = mauso\*p.mauso;

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator/(PHANSO& p) //chu y: Xu ly truong hop MauSo cua Kq co the = 0 {

if(p.tuso == 0) { cout<<"Loi phep chia phan so"<<endl;} // khong chia duoc vì phan so thu 2 là 0 else // co the viet theo cach nhan(\*: da viet) voi nghich dao (!: da viet) cua phan so p [tham so cua ham toan tu] {

 tuso = tuso\*p.mauso;

 mauso =p.tuso\*mauso;

 ToiGian();

}

return \*this;

}

4 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

//4 phep toan 1 ngoi: ++ -- - !

PHANSO& PHANSO::operator++()

{ tuso = tuso + mauso;

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator--()

{ tuso = tuso - mauso;

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator-() // dao dau + <-> -

{

 mauso = -mauso; // chi dao dau tuso or mauso ; nhuung chi nen thuc hien voi mauso (khong nen lam voi tuso; vi co the no = 0), nhung mauso luon != 0  return \*this;

}

PHANSO& PHANSO::operator!() // Nghich dao PhanSo; Xu ly: truong hop Mauso cua KQ co the =0 { if(tuso == 0) { cout<<"Loi phep NGHICH DAO phan so"<<endl;} //khong nghich dao duoc vì tu so là 0 bi chuyen mauso  else

 {

 int t = tuso;

 tuso = mauso;

 mauso = t;

 ToiGian();

 }

 return \*this;

}

//4 phep toan GOP/GHEP [2 ngoi]: += -= \*= /=

PHANSO& PHANSO::operator+=(PHANSO& p)

{ \*this = \*this + p; // ...operator+(...) da viet o tren

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator-=(PHANSO& p)

{ \*this = \*this - p; // ...operator-(...) da viet o tren

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator\*=(PHANSO& p)

{ \*this = \*this \* p; // ...operator\*(...) da viet o tren

ToiGian();

return \*this;

}

PHANSO& PHANSO::operator/=(PHANSO& p) //chu y: Xu ly truong hop MauSo cua Kq co the = 0 { \*this = \*this / p; // ...operator/(...) da viet o tren; TRONG ....operator/.... da xu ly vu tu so cua phanso thu 2 = 0, nen o day KO can thuc hien them  ToiGian();

 return \*this;

}

5 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

//6 phep toan SO SANH [2 ngoi]: > < >= <= == != : KetQua là int theo nghia TRUE: 1, FALSE: 0 int PHANSO::operator>(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return ((t\*m>0)? 1:0);

}

int PHANSO::operator<(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return ((t\*m)<0? 1:0);

}

int PHANSO::operator>=(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return (t\*m? 1:0);

}

int PHANSO::operator<=(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return (t\*m? 0:1);

}

int PHANSO::operator==(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return ((t\*m==0)? 1:0);

}

int PHANSO::operator!=(PHANSO& p)

{

int t = tuso\*p.mauso - mauso\*p.tuso, m = mauso\*p.mauso;

return ((t\*m!=0)? 1:0);

}

//2 ham friend : nhap (tham chieu istream&) /xuat(tham chieu ostream&) PhanSo [2 ngoi: ngoi dau la thu vien: i/o:stream] istream& operator>>(istream& i, PHANSO& p) //i chinh la cin: cin>>p :Ham toan tu tu do (ko thuoc lop) {

cout<<"Nhap tu so = "; i>>p.tuso; //dung duoc p.tuso nho friend

do{

cout<<"Nhap mau so ="; i>>p.mauso;

if(p.mauso == 0) cout<<"Nhap mau so khac 0"<<endl;

}while(p.mauso ==0);//dung duoc p.mauso nho friend

return i;

}

ostream& operator<<(ostream& o, PHANSO& p) //o chinh la cout: cout<<p: Ham toan tu tu do (ko thuoc lop) {

o<<p.tuso<<"/"<<p.mauso<<endl;

return o;

}

6 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

int main()

{

PHANSO p1, p2;

//NHAP 2 PHAN SO

cout<<"Nhap phan so thu 1:"<<endl; cin>>p1;

cout<<"Nhap phan so thu 2:"<<endl; cin>>p2;

//XUAT 2 PHAN SO VUA NHAP

cout<<p1;

cout<<p2;

//KIEM TRA: CAC PHEP TOAN SO SANH: KIEM TRA LAI

if(p1>p1) cout<<"PHAN SO THU 1 LON HON PHAN SO THU 2"<<endl;

else if(p1==p2) cout<<"HAI PHAN SO BANG NHAU"<<endl;

else cout<<"PHAN SO THU 2 LON HON PHAN SO THU 1"<<endl;

//KIEM TRA CAC PHEP TOAN TU HAI NGOI

p1 = p1 + p2;

cout<<"TONG CUA 2 PHAN SO = "<<p1<<endl;

//KIEM TRA CAC PHEP TOAN TU 1 NGOI ++

++p1;

cout<<"++ TANG LEN 1 CHO KET QUA SAU KHI + 2 PHAN SO BAN DAU = "<<p1<<endl;

//KIEM TRA CAC PHEP NGHICH DAO

!p1;

cout<<" NGHICH DAO CHO KET QUA SAU KHI + 2 PHAN SO BAN DAU VA ++ LEN 1 = "<<p1<<endl;

//KIEM TRA CAC PHEP GOP/GHEP

p1+=p2;

cout<<" KET QUA PHAN SO SAU CUNG += PHAN SO THU 2 = "<<p1<<endl;

//KIEM TRA PHEP MO RONG CONG MOT LUC 3 TOAN HANG

PHANSO p3;

cout<<"Nhap Phan so thu 3 = "<<endl; cin>>p3;

p1 = p1 + p2 + p3; // p1: la ket qua cuoi cung cua cac phep toan tren:.... += p2(NSD nhap) P3(4,5) cout<<" KET QUA CONG 3 PHAN SO SAU CUNG = "<<p1<<endl;

//KIEM TRA PHEP MO RONG NGHICH DAO PHAN SO 2 LAN => NHU CU

!!p1;

cout<<" KET QUA DAO PHAN SO SAU CUNG 2 LAN -> NHU CU = "<<p1<<endl;

//KIEM TRA PHEP CONG PHAN SO VOI HANG SO NGUYEN

p1 = (PHANSO)2 + p2;

cout<<" KET QUA CONG PHAN SO SAU CUNG VOI SO NGUYEN 2 = "<<p1<<endl; }

7 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

**Nhận xét & Chú ý:** Operand1(p1) + Operand2(p2) Operand1 thuộc Class : viết thành Method của Class   đt hiện tại tham số

 p1 + p2 => p1.operator+(p2)

- Phép toán nhập xuất không thể là phép toán thuộc lớp vì thành phần bên trái của phép toán không phải  là đối tương phân số mà là đối tượng thuộc lớp istream (đối tượng nhập cin) hoặc đối tượng thuộc lớp  ostream (đối tượng xuất cout). Do đó phép toán nhập xuất phải định nghĩa là hàm không thuộc lớp nhưng  để có thể truy xuất đến dữ liệu private (ts,ms) cần khai báo là hàm bạn của lớp phân số

GIẢI THÍCH: cin[TH1] >> p1[TH2]

|  |
| --- |
| cout thuộc lớp đối tượng ostream  cin thuộc lớp đối tượng istream  cout<<x : cout và x là 2 toán hạng (operand), << toán tử (operator)  ---vì vậy toán tử >> hoặc << PHẢI sử dụng friend (không phải method) Nếu x << cout ===> x.operator(ostream)   cout<<x : không dùng cách trên được |

- Phép toán ps1+ps2 có thể ghi là ps1.operator+(ps2), ps1 gọi là đt hiện tại đang gọi phép toán cộng. - Phép toán + hai toán hạng nhưng khai báo chỉ có toán hạng bên phải [làm tham số của hàm], toán hạng  bên trái chính là đối tượng hiện đang gọi phép toán. Tương tự phép toán nghịch đảo (!) có một toán hạng  nhưng không có khai báo tham số.

- Có thể sử dụng phép toán mở rộng như sau:

|  |
| --- |
| ps4=ps1+ps2+ps3 <==> ps4=(ps1.operator+(ps2)).operater+(ps3); |

ps2=!!ps1 <==> ps2=(ps1.operartor!()).operator!() 

|  |
| --- |
| ps2=ps1+2 : |

KO kể cả ép kiểu <==> khi đó 2 chính là phân số 2/1 và ps2=ps1.operartor+(2/1). 

Nhưng không thể ghi ps2=2+ps1 vì phép toán cộng chỉ có thể được gọi bởi một đối tượng phân số  

mà phải [ép kiểu] ghi

|  |
| --- |
| ps2=(phanso)2+ps1 |

- Phép toán ++ cần trả về tham chiếu [&] để khi thực hiện phép toán nhiều lần, đt ps được thực sự thay  đổi. Phép toán này sử dụng trước hay sau phân số đều tăng ps lên 1.: KHÔNG ĐƯỢC=>MUỐN ĐƯỢC  THÌ PHẢI THÊM HÀM TOÁN TỬ TỰ DO (có 1 tham số PS) LÀ friend của Class

**=================**

8 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân) **MINH HỌA CÁC PHÉP TOÁN**

//4 phep toan 2 ngoi: + - \* /

a.operator+(b) <=> a + b

3 4 3\*7 + 4\*5

-- + ---- = --------------: tối giản

5 7 5\*7

a.operator\*(b) <=> a \* b

3 4 3\*4

-- \* ---- = --------------: tối giản

5 7 5\*7

a.operator/(b) <=> a / b

3 0 3 7 3\*7

-- / ---- = --- \* ---- = ------: LỖI [tối giản]

5 7 5 0 5\*0

========================================================== //4 phep toan 1 ngoi: ++ -- - !

a.operator!() <=> !a NGHỊCH ĐẢO PHÂN SỐ

 0 5

! -- = ----[LỖI: XỬ LÝ] : tối giản

 5 0

----------------------------------------------

a.operator++() <=> ++a

3 3 5 3+5

-- + 1 = --- + ---- = ------: tối giản

5 5 5 5

a.operator-() <=> -a ĐẢO DẤU PHÂN SỐ - <------> + -------------

ban đâu 3

 --- đảo dấu + <--> - thì sẽ có kết quả là

 5

-------------

 3

 - -- KHÔNG CẦN tối giản

 5

================

//KIEM TRA PHEP MO RONG CONG MOT LUC 3 TOAN HANG

17/18 + 1/2 + 4/5 = 101/45 ?????

9 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

**CÁC GIẢI THÍCH**

**TỐI GIẢN & USCLN = ước số chung lớn nhất**

8/4 ==để tối giản==>trước hết phải tìm USCLN(8,4) = 4 :

------------------

x = USCLN(“tử”, “mẫu”) = số nguyên lớn nhất mà “tử” và “mẫu” cùng chia hết cho x ---------------------tức là:

“tử” % x = 0

“mẫu” % x = 0

x lớn nhất thõa mãn 2 đk trên

**//VỀ PHÉP CHIA / TRONG C/C++ có 2 Ý NGHĨA : CHIA NGUYÊN & CHIA THẬP PHÂN** TRONG C/C++, chú ý: phép /

a / b : là phép chia NGUYÊN nếu cả a và b đều là số nguyên

a / b : là phép chia THẬP PHÂN nếu ít nhất a hoặc b là số thực

 VD 9 / 5 : chia nguyen ==nên kết quả ==> 1

 9.0 / 5 hoặc 9/5.0 hoặc 9.0/5.0 ===> 1.8

 int a =9 , b = 5;

 int c;

 c = a/b; CHIA NGUYEN : c= 1

 ===============

 float c = a/b; c = 1.0 chú ý: trong lập trình 1 và 1.0 là khác nhau ===============

 float c = 1.0\*a/b; c = 1.8 *phép \* và / là ngang hàng: nên thực hiện trái->phải*float c = a/b\*1.0; c = 1.0 *phép \* và / là ngang hàng: nên thực hiện trái->phải*float c = 1.0\*(a/b); c = 1.0 *phép \* và / là ngang hàng: nên thực hiện trái->phải* float c = (float)(a/b); c = 1.8 *ép kiểu được ưu tiên thực hiện trước => nên đúng* float c = (float)a/b; c = 1.8

 float c = a/(float)b; c = 1.8