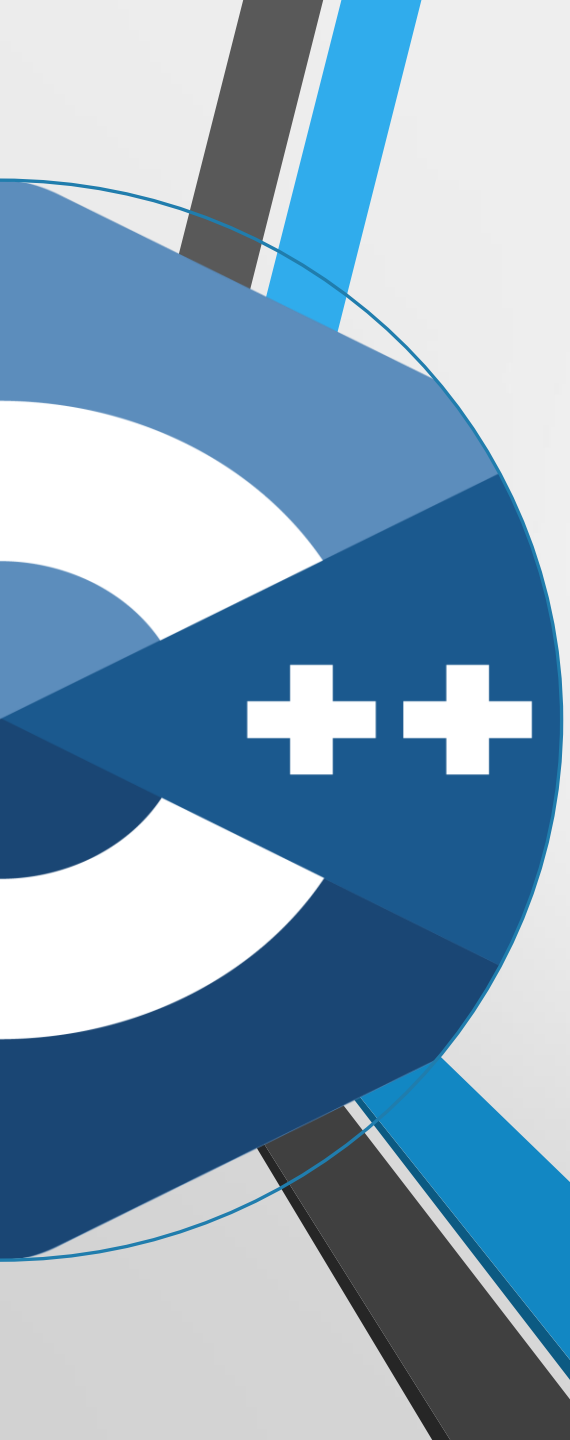

Thiết kế lớp số phức

THIẾT KẾ LỚP CSoPhuc

THIẾT KẾ LỚP CSoPhuc

- **Thuộc tính:**
  - Phần thực.
  - Phần ảo

- **Phương thức:**
  - Nhóm các phương thức khởi tạo.
  - Nhóm các phương thức cung cấp thông tin.
  - Nhóm phương thức cập nhật thông tin.
  - Nhóm các phương thức kiểm tra.
  - Nhóm các phương thức xử lý.

THIẾT KẾ LỚP CSoPhuc

- **Nhóm các phương thức khởi tạo:**
  - 1. Phương thức khởi tạo mặc định.
  - 2. Phương thức khởi tạo sao chép.
  - 3. Phương thức khởi tạo khi biết đầy đủ thông tin.
  - 4. Phương thức thiết lập mặc định.
  - 5. Phương thức thiết lập sao chép.
  - 6. Phương thức thiết lập khi biết đầy đủ thông tin.
  - 7. Phương thức Nhập.
  - 8. Toán tử vào.

THIẾT KẾ LỚP CSoPhuc

- **Nhóm các phương thức cung cấp thông tin:**
  - 1. Phương thức Xuất.
  - 2. Toán tử ra.
  - 3. Phương thức cung cấp phần thực.
  - 4. Phương thức cung cấp phần ảo.

THIẾT KẾ LỚP CSoPhuc

- **Nhóm các phương thức cập nhật thông tin:**
  - 1. Toán tử gán.
  - 2. Phương thức cập nhật phần thực.
  - 3. Phương thức cập nhật phần ảo.

THIẾT KẾ LỚP CSoPhuc

- **Phương thức kiểm tra:**
  - Kiểm tra có bằng số 0 không?
  - Kiểm tra hai số phức có trùng không?
  - Kiểm tra hai số phức có không trùng không?
  - Kiểm tra có phải là số thực không?
  - Kiểm tra có phải là số thuần ảo không?

THIẾT KẾ LỚP CSoPhuc

- **Nhóm các phương thức xử lý:**
  - 1. Tính mô-đun của số phức
  - 2. Toán tử so sánh bằng
  - 3. Toán tử so sánh khác
  - 4. Toán tử so sánh lớn hơn
  - 5. Toán tử so sánh nhỏ hơn
  - 6. Toán tử so sánh lớn hơn bằng
  - 7.Toán tử so sánh nhỏ hơn bằng

THIẾT KẾ LỚP CSoPhuc

- **Nhóm các phương thức xử lý:**
  - 7. Toán tử cộng
  - 8. Toán tử trừ
  - 9. Toán tử nhân
  - 10. Toán tử chia
  - 11. Phương thức phá hủy

KHAI BÁO LỚP CSoPhuc

KHAI BÁO LỚP CSoPhuc

- **Thuộc tính:**
  - Phần thực.
  - Phần ảo.

KHAI BÁO LỚP CSoPhuc

```
class CSoPhuc
{
   private:
      float thuc;
      float ao;
   public:
```

KHAI BÁO LỚP CSoPhuc

- **Thuộc tính:**
  - Phần thực.
  - Phần ảo.

- **Phương thức:**
  - Nhóm các phương thức khởi tạo.
  - Nhóm các phương thức cung cấp thông tin.
  - Nhóm phương thức cập nhật thông tin.
  - Nhóm các phương thức kiểm tra.
  - Nhóm các phương thức xử lý.

KHAI BÁO LỚP CSoPhuc

```
class CSoPhuc
{
   private:
      float thuc;
      float ao;
   public:
      // Nhóm phương thức khởi tạo
      // Nhóm phương thức cung cấp thông tin
      // Nhóm phương thức cập nhật thông tin
      // Nhóm phương thức kiểm tra
      // Nhóm phương thức xử lý
```

KHAI BÁO LỚP CSoPhuc

- **Nhóm các phương thức khởi tạo:**
  - 1. Phương thức khởi tạo mặc định.
  - 2. Phương thức khởi tạo sao chép.
  - 3. Phương thức khởi tạo khi biết đầy đủ thông tin.
  - 4. Phương thức thiết lập mặc định.
  - 5. Phương thức thiết lập sao chép.
  - 6. Phương thức thiết lập khi biết đầy đủ thông tin.
  - 7. Phương thức Nhập.
  - 8. Toán tử vào.

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

KHAI BÁO LỚP CSoPhuc

- **Nhóm các phương thức cung cấp thông tin:**
  - 1. Phương thức Xuất.
  - 2. Toán tử ra.
  - 3. Phương thức cung cấp phần thực.
  - 4. Phương thức cung cấp phần ảo.

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cung cấp thông tin
void Xuat();
friend ostream& operator<<(ostream&, CSoPhuc&)
float GetThuc();
float GetAo();
```

KHAI BÁO LỚP CSoPhuc

- **Nhóm các phương thức cập nhật thông tin:**
  - 1. Toán tử gán.
  - 2. Phương thức cập nhật phần thực.
  - 3. Phương thức cập nhật phần ảo.

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cập nhật thông tin
CHonSo& operator = (const CHonSo&);
void SetThuc(float);
void SetAo(float);
```

KHAI BÁO LỚP CSoPhuc

- **Phương thức kiểm tra:**
  - Kiểm tra số phức có bằng số 0 không?
  - Kiểm tra hai số phức có trùng nhau không?
  - Kiểm tra hai số phức có không trùng nhau không?
  - Kiểm tra có phải là số thực không?
  - Kiểm tra có phải số thuần ảo không?

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

KHAI BÁO LỚP CSoPhuc

- **Nhóm các phương thức xử lý:**
  - 1. Tính mô-đun của số phức
  - 2. Toán tử so sánh bằng
  - 3. Toán tử so sánh khác
  - 4. Toán tử so sánh lớn hơn
  - 5. Toán tử so sánh nhỏ hơn
  - 6. Toán tử so sánh lớn hơn bằng
  - 7. Toán tử so sánh nhỏ hơn bằng

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
    float ModunSoPhuc();
    int operator == (CSoPhuc&);
    int operator != (CSoPhuc&);
    int operator > (CSoPhuc&);
    int operator >= (CSoPhuc&);
    int operator < (CSoPhuc&);
    int operator <= (CSoPhuc&);
```

KHAI BÁO LỚP CSoPhuc

- **Nhóm các phương thức xử lý:**
  - 7. Toán tử cộng
  - 8. Toán tử trừ
  - 9. Toán tử nhân
  - 10. Toán tử chia
  - 11. Phương thức phá hủy

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~ CSoPhuc();
```

PHƯƠNG THỨC KHỞI TẠO

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc &);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc &);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
void CSoPhuc::KhoiTao()
{
    thuc = 0;
    ao = 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
void CSoPhuc::KhoiTao(const CSoPhuc&P)
{
    thuc = P.thuc;
    ao = P.ao;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
void CSoPhuc::KhoiTao(float tt, float aa)
{
    thuc = tt;
    ao = aa;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc &);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
CSoPhuc::CSoPhuc()
{
    thuc = 0;
    ao = 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CHonSo KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
CSoPhuc::CSoPhuc(const CSoPhuc &P)
{
    thuc = P.thuc;
    ao = P.ao;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&, CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
CSoPhuc::CSoPhuc(float tt, float aa)
{
    thuc = tt;
    ao = aa;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&,CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
void CSoPhuc::Nhap()
{
    cout << "Nhap phan thuc: ";
    cin >> thuc;
    cout << "Nhap phan ao: ";
    cin >> ao;
}
```

KHAI BÁO LỚP CHonSo

```
// Nhóm phương thức khởi tạo
void KhoiTao();
void KhoiTao(const CSoPhuc&);
void KhoiTao(float, float);
CSoPhuc();
CSoPhuc KhoiTao(const CSoPhuc&);
CSoPhuc(float, float);
void Nhap();
friend istream& operator>>(istream&,CSoPhuc&);
```

NHÓM PHƯƠNG THỨC KHỞI TẠO

```
istream& operator >> (istream& is, CSoPhuc &P)
{
    cout << "Nhap phan thuc: " ;
    is >> P.thuc;
    cout << "Nhap tu: ";
    is >> P.ao;
    return is;
}
```

PHƯƠNG THỨC CUNG CẤP THÔNG TIN

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cung cấp thông tin
void Xuat();
friend ostream& operator<<(ostream&, CSoPhuc&);
float GetThuc();
float GetAo();
```

Nhóm phương thức cung cấp thông tin

```
void CSoPhuc::Xuat()
{
    cout<<thuc<<" + ("<<ao<<")i ";
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cung cấp thông tin
void Xuat();
friend ostream& operator<<(ostream&,CSoPhuc&);
float GetThuc();
float GetAo();
```

Nhóm phương thức cung cấp thông tin

```
ostream& operator << (ostream& os,CSoPhuc& P)
{
    os<<P.thuc<<" + ("<<P.mau<< ")i";
    return os;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cung cấp thông tin
void Xuat();
friend ostream& operator<<(ostream&, CSoPhuc&);
float GetThuc();
float GetAo();
```

Nhóm phương thức cung cấp thông tin

```
float CSoPhuc::GetThuc()
{
    return thuc;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cung cấp thông tin
void Xuat();
friend ostream& operator<<(ostream&,CSoPhuc&);
float GetThuc();
float GetAo();
```

Nhóm phương thức cung cấp thông tin

```
float CSoPhuc::GetAo()
{
    return ao;
}
```

PHƯƠNG THỨC CẬP NHẬT THÔNG TIN

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cập nhật thông tin
CSoPhuc& operator = (const CSoPhuc&);
void SetThuc(float);
void SetAo(float);
```

Nhóm phương thức cập nhật thông tin

```
CSoPhuc& CSoPhuc::operator = (const CSoPhuc &P)
{
    thuc = P.thuc;
    ao = P.ao;
    return *this;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cập nhật thông tin
CHonSo& operator = (const CHonSo&);
void SetThuc(float);
void SetAo(float);
```

Nhóm phương thức cập nhật thông tin

```
void CSoPhuc::SetThuc(float tt)
{
    thuc = tt;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức cập nhật thông tin
CSoPhuc& operator = (const CSoPhuc&);
void SetThuc(float);
void SetAo(float);
```

Nhóm phương thức cập nhật thông tin

```
void CSoPhuc::SetAo(float aa)
{
    ao = aa;
}
```

PHƯƠNG THỨC KIỂM TRA

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

Nhóm phương thức kiểm tra

```
int CSoPhuc::isBangKhong()
{
    if (thuc == 0 && ao == 0)
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

Nhóm phương thức kiểm tra

```
int CSoPhuc::isTrung(const CSoPhuc &P)
{
    if (thuc == P.thuc && ao == P.ao)
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

Nhóm phương thức kiểm tra

```
int CSoPhuc::isKhongTrung(const CSoPhuc &P)
{
    if (!(thuc == P.thuc && ao == P.ao))
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

Nhóm phương thức kiểm tra

```
int CSoPhuc::ktSoThuc()
{
    if (ao != 0)
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức kiểm tra
int isBangKhong();
int isTrung(const CSoPhuc&);
int isKhongTrung(const CSoPhuc&);
int ktSoThuc();
int ktThuanAo();
```

Nhóm phương thức kiểm tra

```
int CSoPhuc::ktThuanAo()
{
    if (thuc == 0)
        return 1;
    return 0;
}
```

PHƯƠNG THỨC XỬ LÝ

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
float ModunSoPhuc();
int operator == (CSoPhuc &);
int operator != (CSoPhuc &);
int operator > (CSoPhuc &);
int operator >= (CSoPhuc &);
int operator < (CSoPhuc &);
int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
float CSoPhuc::ModunSoPhuc(CSoPhuc &P)
{
    float Modulus = sqrt((thuc*thuc)+(ao*ao));
    return Modulus;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator==(CSoPhuc &P)
{
    if(ModunSoPhuc() == P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator!=(CSoPhuc &P)
{
    if(ModunSoPhuc() != P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator>(CSoPhuc &P)
{
    if(ModunSoPhuc() > P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CHonSo

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator>=(CSoPhuc &P)
{
    if(ModunSoPhuc() >= P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator<(const CSoPhuc &P)
{
    if(ModunSoPhuc() < P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
 float ModunSoPhuc();
 int operator == (CSoPhuc &);
 int operator != (CSoPhuc &);
 int operator > (CSoPhuc &);
 int operator >= (CSoPhuc &);
 int operator < (CSoPhuc &);
 int operator <= (CSoPhuc &);
```

Nhóm phương thức xử lý

```
int CSoPhuc::operator<=(const CSoPhuc &P)
{
    if(ModunSoPhuc() <= P.ModunSoPhuc())
        return 1;
    return 0;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~ CSoPhuc();
```

Nhóm phương thức xử lý

```
CSoPhuc CSoPhuc::operator+(CSoPhuc P)
{
    CSoPhuc temp;
    temp.thuc = thuc + P.thuc;
    temp.ao = ao + P.ao;
    return temp;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~ CSoPhuc();
```

Nhóm phương thức xử lý

```
CSoPhuc CSoPhuc::operator-(CSoPhuc P)
{
    CSoPhuc temp;
    temp.thuc = thuc - P.thuc;
    temp.ao = ao - P.ao;
    return temp;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~ CSoPhuc();
```

Nhóm phương thức xử lý

```
CSoPhuc CSoPhuc::operator*(CSoPhuc P)
{
     CSoPhuc temp;
     temp.thuc = (thuc * P.thuc) - (ao * P.ao);
     temp.ao = (thuc * P.ao) + (ao * P.thuc);
     return temp;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~CSoPhuc();
```

Nhóm phương thức xử lý

```
CSoPhuc CSoPhuc::operator/(CSoPhuc P)
{
     CSoPhuc temp;
     temp.thuc = ((thuc*P.thuc)+(ao*P.ao))
                 /((P.thuc*P.thuc)+(P.ao*P.ao));
     temp.ao = ((P.thuc*ao)-(thuc*P.ao))
               /((P.thuc*P.thuc)+(P.ao*P.ao));
     return temp;
}
```

KHAI BÁO LỚP CSoPhuc

```
// Nhóm phương thức xử lý
CSoPhuc operator+(CSoPhuc);
CSoPhuc operator-(CSoPhuc);
CSoPhuc operator*(CSoPhuc);
CSoPhuc operator/(CSoPhuc);
~CSoPhuc();
```

Nhóm phương thức xử lý

```
CSoPhuc::~CSoPhuc()
{
    return;
}
```

Kết thúc.

- Nguyễn Văn Linh – 20520613
- Hồng Gia Hy – 20520561
- Nguyễn Huy Trí Dũng -20520459
- Đỗ Trọng Tình – 20520318
- Nguyễn Công Đoàn - 20520447