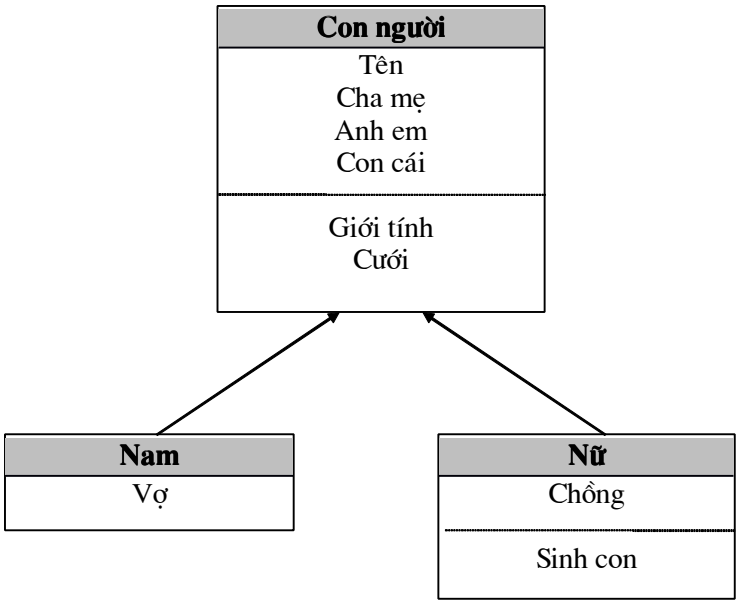
BÀI TOÁN QUAN HỆ GIA ĐÌNH

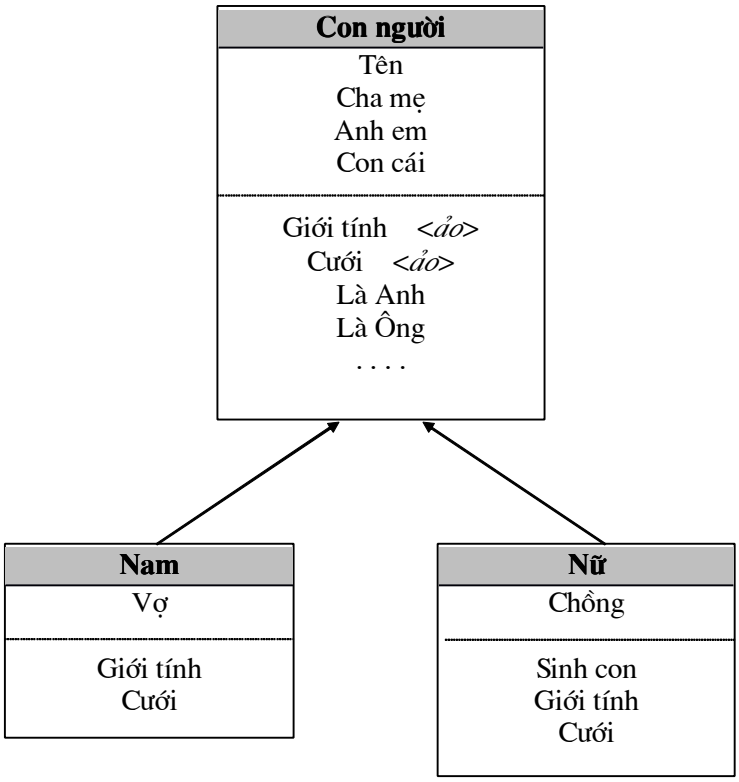
Trong mục này, ta sẽ xây dựng chương trình cho bài toán quan hệ gia đình đã được phân tích ở chương một. Theo như sự phân tích ban đầu của bài toán, ta có một tập các cá thể và mô tả bằng lớp **Con người** bao gồm các thuộc tính tên, anh em, cha mẹ, ... và các phương thức sinh, cưới, ... Nhưng ta có nhận xét rằng phương thức sinh chỉ thực hiện được trên những cá thể là nữ và phương thức cưới chỉ xảy ra cho hai cá thể khác giới. Như vậy có sự phân chia tập đối tượng của bài toán thành hai lớp khác nhau là **Nam** và **Nữ**. Rõ ràng, hai lớp này phải kế thừa từ lớp Con người. Lớp con người sẽ chứa các thuộc tính và phương thức chung, dù cá thể là Nam hay Nữ cũng đều phải có. Ngoài cá thành phần được kế thừa từ lớp Con người, lớp Nam có thêm thuộc tính **Vợ**, lớp Nữ có thêm thuộc tính **Chồng** và phương thức **Sinh con**. Thiết kế các lớp ban đầu của bài toán như hình dưới đây.



Thiết kế sơ bộ các lớp của bài toán

Phương thức **Giới tính** dùng để trả lời xem một cá thể đó là Nam hay Nữ. Nếu là Nam kết quả là 1 còn là kết quả là 0. Rõ ràng tại lớp Con người

phương thức Giới tính không thể trả lời được đó là Nam hay Nữ. Câu trả lời chỉ xác định tại các phương thức Giới tính ở lớp kế thừa Nam và Nữ. Phương thức Giới tính ở lớp Nam trả kết quả là 1 còn ở lớp Nữ trả kết quả là 0. Để thực hiện được kĩ thuật này, ta dùng kĩ thuật hàm ảo trong LTHĐT. Lập luận tương tự cho phương thức **Cưới**, bởi vì phương thức này cần biết cưới chồng hay cưới vợ. Để trả lời cho các câu hỏi về mối quan hệ gia đình chúng ta cũng cần phải xây những phương thức để trả lời các câu hỏi như **Là Anh**, **Là Ông**(X), v.v... Hình dưới đây thiết kế các lớp của bài toán.



Thiết kế các lớp của bài toán

Sau đây là thể hiện của các lớp dưới ngôn ngữ C++ có bổ sung thêm một số thuộc tính và phương thức phục vụ việc cài đặt lớp.

```
class Nguoi {
    friend class Nam;
    friend class Nu;
```

```
    char Ten[25];
    Nam *Bo;
    Nu *Me;
    Nguoi *AnhChi[10], *CacEm[10], *CacCon[10];
    int SoAnhChi, SoEm, SoCon;

    Nguoi(char *ten, Nam *bo, Nu *me) :
        Bo(bo), Me(me), SoAnhChi(0), SoEm(0), SoCon(0)
    {
        strcpy(Ten, ten);
    }
    void ThemAnhChi(Nguoi* nguoi)
    {
        AnhChi[SoAnhChi++] = nguoi;
    }
    void ThemEm(Nguoi* nguoi)
    {
        CacEm[SoEm++] = nguoi;
    }
    void ThemCon(Nguoi* nguoi)
    {
        CacCon[SoCon++] = nguoi;
    }
public:
    // 1 la Nam, 0 la Nu
    virtual int GioiTinh()=0;
    virtual int Cuoi(Nguoi*)=0;
    int LaCha(Nguoi *);
    int LaMe(Nguoi *);
    int LaCon(Nguoi *);
```

```
    int LaAnh(Nguoi *);
    int LaChi(Nguoi *);
    int LaEm(Nguoi *);
    int LaCo(Nguoi *);
    int LaDi(Nguoi *);
    int LaChu(Nguoi *);
    int LaCau(Nguoi *);
    int LaMo(Nguoi *);
    int LaBac(Nguoi *);
    int LaOngNoi(Nguoi *);
    int LaBaNoi(Nguoi *);
    int LaOngNgoai(Nguoi *);
    int LaBaNgoai(Nguoi *);
    int LaAnhHo(Nguoi *);
    int LaChiHo(Nguoi *);
    int LaEmHo(Nguoi *);
    virtual int LaVo(Nguoi*)=0;
    virtual int LaChong(Nguoi*)=0;
};

class Nam : public Nguoi
{
    Nu *Vo;
    int LaVo(Nguoi *) { return 0; }
public:
    Nam(char *ten, Nam *bo=0, Nu *me=0) :
          Nguoi(ten, bo, me), Vo(0) {}
int GioiTinh() { return 1; }
    int Cuoi(Nguoi *vo);
    int LaChong(Nguoi * nguoi);
```

```
};

class Nu : public Nguoi
{
    Nam *Chong;
    int LaChong(Nguoi *) { return 0; }
public:
    Nu(char *ten, Nam *bo=0, Nu *me=0):
        Nguoi(ten, bo, me), Chong(0) {}
    int GioiTinh() { return 0; }
    int Cuoi(Nguoi *chong);
    void SinhCon(char* ten, int gioitinh);
    int LaVo(Nguoi * nguoi);
};
```

Phương thức cưới chỉ thực hiện đối với cá thể chưa lập gia đình. Trong trường hợp đã lập gia đình thì nó sẽ trả ra 0.

```
int Nam::Cuoi(Nguoi *vo)
{
    if (Vo||vo->GioiTinh()) return 0;
    Vo = (Nu*)vo;
    Vo->Cuoi(this);
    return 1;
}
int Nu::Cuoi(Nguoi *chong)
{
    if (Chong||chong->GioiTinh()==0) return 0;
    Chong = (Nam*)chong;
    Chong->Cuoi(this);
    return 1;
}
```

```
void Nu::SinhCon(char *ten, int gioitinh)
{
    Nguoi* nguoi = TaoNguoi(ten, gioitinh, Chong, this);
    ThemCon(nguoi);
    if (Chong) Chong->ThemCon(nguoi);
    for (int i=0; i<SoCon; i++)
    {
        CacCon[i]->ThemEm(nguoi);
        nguoi->ThemAnhChi(CacCon[i]);
    }
}
```

Trong phương thức sinh con đã dùng hàm TaoNguoi để tạo ra một thể hiện của lớp Nam hay Nu phụ thuộc vào giới tính. Đối tượng mới tạo ra sẽ được gia nhập vào cộng đồng và được xem xét các mối quan hệ sau này, chi tiết về hàm này chúng ta sẽ bàn luận sau. Sau khi đã có đối tượng thì các mối quan hệ gia đình cha, mẹ, con cái, anh, chị, em phải được xác lập.

Thực hiện cài đặt các phương thức trả lời câu hỏi quan hệ của lớp Con người. Đối với các mối quan hệ gần như LaAnh, LaCha thì việc kiểm tra rất đơn giản thông qua các thuộc tính Bo, Me, AnhChi,... của đối tượng. Nhưng đối với mối quan hệ xa hơn chút ít như LaOngNoi, LaCo, LaChu,... thì trở nên khó khăn hơn nhiều. Chúng ta dùng một phương pháp kiểm tra đơn giản theo mô hình toán học. Ví dụ, A là ông nội của B khi và chỉ khi trong cộng đồng tồn tại một đối tượng X mà A là cha của X và X là cha của B. Như vậy để thực hiện được kiểm tra này thì cần phải lưu được toàn bộ các đối tượng trong cộng đồng để phục vụ tìm kiếm X. Trong chương trình dùng một mảng tĩnh các con trỏ tới đối tượng Con người để quản lý cộng đồng người này. Mảng này được khai báo như một thành phần tĩnh của lớp Con người. Phương thức **TaoNguoi** sẽ tạo một đối tượng là Nam hoặc Nữ phụ thuộc giới tính truyền vào. Đối tượng mới tạo ra sẽ được thêm vào cộng đồng. Phương thức **TimNguoi** tìm đối tượng trong cộng đồng có tên như tên đưa vào, nếu không tìm thấy trả về NULL. Dưới đây là một số bổ sung cho lớp Con người.

```
class Nguoi
{
    . . .
```

```
    static Nguoi* NhanDan[100];
    static int SoDan;
public:
    . . .
    static int LaySoDan() { return SoDan; }
    static Nguoi* ThemDan(Nguoi* nguoi)
    {
        return NhanDan[SoDan++] = nguoi;
    }
    static Nguoi* TaoNguoi(char*, int, Nam *bo=0, Nu *me=0);
    static Nguoi* TimNguoi(char* ten);
};

Nguoi* Nguoi::NhanDan[];
int     Nguoi::SoDan = 0;

Nguoi*
Nguoi::TaoNguoi(char* ten, int gioitinh, Nam *bo, Nu *me)
{
    return gioitinh ? ThemDan(new Nam(ten, bo, me))
                    : ThemDan(new Nu(ten, bo, me));
}
Nguoi* Nguoi::TimNguoi(char* ten)
{
    for (int i=0; i<SoDan; i++)
        if (strcmp(ten, NhanDan[i]->LayTen())==0)
        return NhanDan[i];
    return 0;
}
```

Sau khi đã tổ chức được dữ liệu lưu trữ con người chúng ta có thể xây dựng các hàm kiểm tra quan hệ như dưới đây. A là đối tượng gọi hàm kiểm tra, B là đối tượng truyền vào, X là đối tượng tìm kiếm NhanDan[i] (dùng vòng **for** để duyệt).

```
int Nguoi::LaOngNoi(Nguoi *nguoi)
{
    if (GioiTinh()==0) return 0;
    for (int i=0; i<SoDan; i++)
        if (LaCha(NhanDan[i])&&NhanDan[i]->LaCha(nguoi))
return 1;
    return 0;
}
int Nguoi::LaBaNoi(Nguoi *nguoi)
{
    if (GioiTinh()) return 0;
    for (int i=0; i<SoDan; i++)
        if (LaMe(NhanDan[i])&&NhanDan[i]->LaCha(nguoi))
            return 1;
    return 0;
}
```

Mục đích của chương trình là sau khi đã có dữ liệu phải trả lời được câu hỏi quan hệ khi đưa tên hai người vào. Sau khi người sử dụng đưa tên vào chúng ta sẽ tìm đối tượng đó trong cộng đồng. Nếu tìm thấy thì gọi hàm đưa ra thông báo quan hệ của hai đối tượng vừa tìm thấy. Trong hàm dưới đây thực hiện bằng cách kiểm tra các quan hệ bề trên nếu tất cả không thoả mãn thì lại đổi ngược lại đối tượng gọi phương thức để kiểm tra.

```
char qh[256];
// đưa ra thông báo về quan hệ của 2 đối tượng
char* QuanHe(Nguoi* A, Nguoi* B)
{
    for (int i=1; i<=2; i++)
    {
```

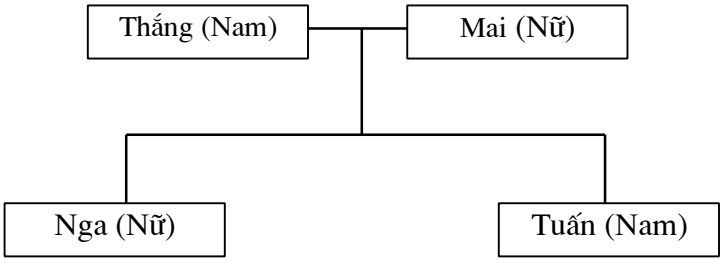
```
    strcpy(qh, A->LayTen());
    strcat(qh, " va ");
    strcat(qh, B->LayTen());
    strcat(qh, " co quan he ");
    if (A->LaOngNoi(B))
        return strcat(qh, "ong chau noi");
    if (A->LaBaNoi(B))
        return strcat(qh, "ba chau noi");
    if (A->LaOngNgoai(B))
        return strcat(qh, "ong chau ngoai");
    if (A->LaBaNgoai(B))
        return strcat(qh, "ba chau ngoai");
    if (A->LaCha(B))
        return strcat(qh, "cha con");
    if (A->LaMe(B))
        return strcat(qh, "me con");
    if (A->LaCo(B))
        return strcat(qh, "co chau");
    if (A->LaDi(B))
        return strcat(qh, "di chau");
    if (A->LaChu(B))
        return strcat(qh, "chu chau");
    if (A->LaBac(B))
        return strcat(qh, "bac chau");
    if (A->LaAnh(B))
        return strcat(qh, "anh em");
    if (A->LaChi(B))
        return strcat(qh, "chi em");
    if (A->LaAnhHo(B))
        return strcat(qh, "anh em ho");
```

```
        if (A->LaChiHo(B))
            return strcat(qh, "chi em ho");
        if (A->LaVo(B))
            return strcat(qh, "vo chong");

        Nguoi* temp = A;
        A = B;
        B = temp;
    }
    strcpy(qh, A->LayTen());
    strcat(qh, " va ");
    strcat(qh, B->LayTen());
    return strcat(qh, " khong co quan he gia dinh");
}
// tìm quan hệ của hai đối tượng có tên nhập vào từ bàn phím
void TimQuanHe()
{
    clrscr();
    char ten1[25];
    cout << "Ten nguoi thu nhat: ";
    gets(ten1);
    Nguoi *A = Nguoi::TimNguoi(ten1);
    if (A==0)
    {
        cout << "Khong co nguoi ten " << ten1 << endl;
        getch();
        return;
    }
    char ten2[25];
    cout << "Ten nguoi thu hai: ";
```

```
    gets(ten2);
    Nguoi *B = Nguoi::TimNguoi(ten2);
    if (B==0)
    {
        cout << "Khong co nguoi ten " << ten2 << endl;
        getch();
        return;
    }
    cout << QuanHe(A, B) << endl;
    getch();
}
```

Để đơn giản trong việc thiết kế nhập dữ liệu cho chương trình, chúng ta dùng phương pháp nhập dữ liệu từ tệp. Việc hình thành con người và các mối quan hệ của chúng là thông qua các sự kiện chính: Tạo một con người, Đám cưới của hai người, Sinh con của người phụ nữ. Do vậy tệp dữ liệu phải thể hiện được điều này. Dùng tệp văn bản để mô tả các sự kiện tỏ ra thích hợp trong trường hợp này. Dưới đây là một tệp văn bản mô tả cho một quan hệ gia đình đơn giản.



Tệp văn bản dữ liệu:

Tao Thang 1	Tạo người tên Thắng là nam
Tao Mai 0	Tạo người tên Mai là nữ
Cuoi Thang Mai	Thắng cưới Mai
Sinh Mai Nga 0	Mai sinh con gái tên là Nga
Sinh Mai Tuan 1	Mai sinh con trai tên là Tuấn

Hàm nhập dữ liệu từ tệp văn bản.

```
void NhapDuLieu() {
    clrscr();
    char s[80];
    cout << "Ten tep nhap du lieu: ";
    cin >> s;
    ifstream input(s, ios::in|ios::nocreate);
    input.seekg(0L, ios::end );
    if ( input.tellg() < 0)  {
        cout << "Loi mo tep ! \n";
        getch();
        return;
    }
    input.seekg(0L, ios::beg);
    cout << "Dang nhap du lieu........\n";
    int dong = 1;
    while (1)  {
        input.getline(s, sizeof(s));
        if (input.gcount()==0) break;
        if (strcmp(s, "")==0)        {
            dong++;
            continue;
```

```
    }
    if (strcmp(s, "Tao")==0) {
        char ten[25];
        input.getline(ten, sizeof(ten));
        if (strcmp(ten, "")) {
            int gt;
            input >> gt;
            cout << "Tao nguoi ten " << ten << endl;
            if (Nguoi::TimNguoi(ten))
                cout << "Da co nguoi ten la " << ten << endl;
            else
                Nguoi::TaoNguoi(ten, gt);
            dong += 2;
            continue;
        }
    }
    if (strcmp(s, "Cuoi")==0) {
        char ten1[25];
        input.getline(ten1, sizeof(ten1));
        char ten2[25];
        input.getline(ten2, sizeof(ten2));
        Nguoi* A = Nguoi::TimNguoi(ten1);
        Nguoi* B = Nguoi::TimNguoi(ten2);
        cout << "Cuoi " << ten1 << " va "<< ten2 << endl;
        if (A==0)
     cout << "Khong co nguoi ten " << ten1 << endl;
        if (B==0)
     cout << "Khong co nguoi ten " << ten2 << endl;
        if (A&&B&&A->Cuoi(B)==0)
            cout << "Khong cuoi duoc\n";
```

```
        dong += 2;
        continue;
    }
    if (strcmp(s, "Sinh")==0)
    {
        char ten1[25];
        input.getline(ten1, sizeof(ten1));
        char ten2[25];
        input.getline(ten2, sizeof(ten2));
        if (strcmp(ten2, "")) {
            cout << ten1 << " sinh con "<< ten2 << endl;
            Nguoi* A = Nguoi::TimNguoi(ten1);
            if (A==0)
      cout << "Khong co nguoi ten " << ten1 << endl;
            int gt;
            input >> gt;
            if (Nguoi::TimNguoi(ten2))
                cout << "Da co nguoi ten la " << ten2 << endl;
            else {
                if ( A )
                    if ( A->GioiTinh() )
             cout << ten1 <<" la nam khong sinh con duoc\n";
                    else
                        ((Nu*)A)->SinhCon(ten2, gt);
            }
            dong += 3;
            continue;
        }
    }
    cout << "Loi o dong thu " << dong << endl;
```

```
        break;
    }
    cout << "Ket thuc nhap.\n";
    getch();
}
```

Viết thêm menu cho chương trình có dạng như sau:

Lua chon cong viec theo so

1. Nhap du lieu
2. Tim quan he
3. Ket thuc

```
void Menu() {
    clrscr();
    cout << "\n\n Lua chon cong viec theo so\n\n";
    cout << "  1. Nhap du lieu\n";
    cout << "  2. Tim quan he\n";
    cout << "  3. Ket thuc\n";
}
void main(){
  int i;
    Menu();
    do
    {
        i = getch();
        switch (i)    {
            case '1':
                Nguoi::XoaDuLieu();
                NhapDuLieu();
                Menu();
                break;
```

```
            case '2':
                TimQuanHe();
                Menu();
        }
    }while (i!='3');
    Nguoi::XoaDuLieu();
}
```

Ví dụ sau khi nhập dữ liệu cho chương trình từ tệp dữ liệu như ta có ở trên và thực hiện tìm quan hệ màn hình sẽ có dạng như sau

```
Ten nguoi thu nhat: Thang
Ten nguoi thu hai: Mai
Mai va Thang co quan he vo chong
```