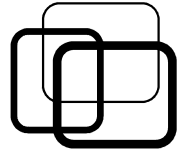
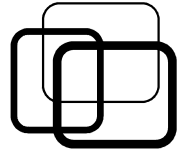


Hàm và cách tổ chức chương trình C

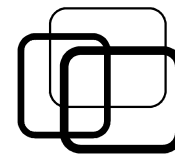
GV. Nguyễn Minh Huy



- Khái niệm hàm.
- Truyền tham số và tầm vực.
- Tổ chức chương trình C.



- **Khái niệm hàm.**
- Truyền tham số và tầm vực.
- Tổ chức chương trình C.



■ Xét chương trình sau:

- Nhập vào 3 số nguyên $a, b, c \geq 0$.

- Tính và xuất $S = a! + b! + c!$.

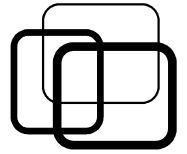
→ Hãy chỉ ra những phần trùng lặp của chương trình.

■ Điểm yếu của chương trình trùng lặp:

- Tốn thời gian, công sức.

- Khi có thay đổi → sửa nhiều chỗ.

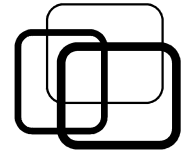
→ Viết 1 lần, tái sử dụng lại nhiều lần.



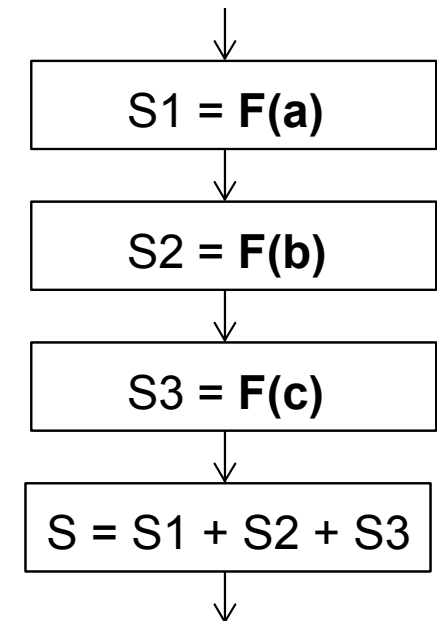
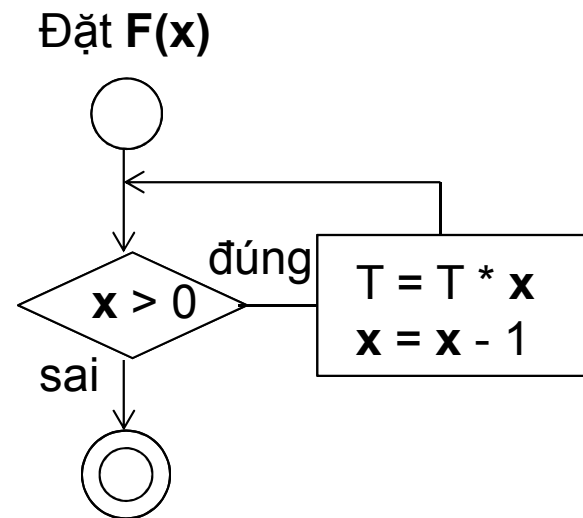
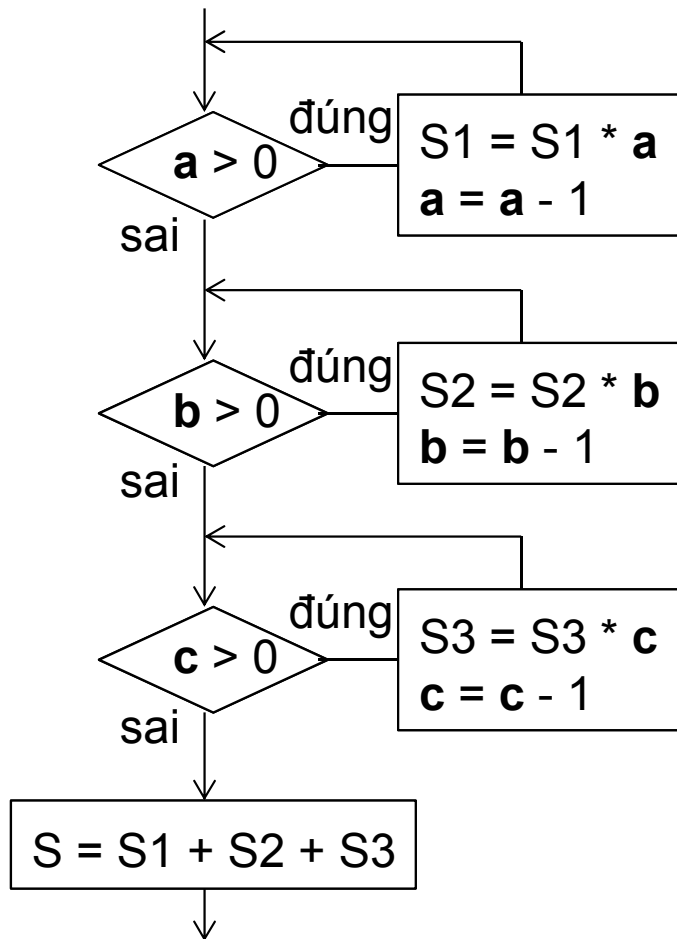
■ Phương pháp đặt hàm:

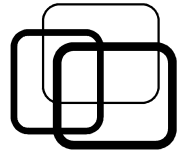
- B1: $S1 = 1$.
 - B2: Lặp $a > 0$
 $S1 = S1 * a$.
 $a = a - 1$.
 - B3: $S2 = 1$.
 - B4: Lặp $b > 0$
 $S2 = S2 * b$.
 $b = b - 1$.
 - B5: $S3 = 1$.
 - B6: Lặp $c > 0$
 $S3 = S3 * c$.
 $c = c - 1$.
 - B7: $S = S1 + S2 + S3$.
- B0: Đặt $F(x)$ như sau:
 - B0.1: $T = 1$.
 - B0.2: Lặp $x > 0$
 $T = T * x$.
 $x = x - 1$.
 - B1: $S1 = F(a)$.
 - B2: $S2 = F(b)$.
 - B3: $S3 = F(c)$.
 - B4: $S = S1 + S2 + S3$.

Khái niệm hàm



■ Phương pháp đặt hàm:





■ Hàm trong ngôn ngữ C:

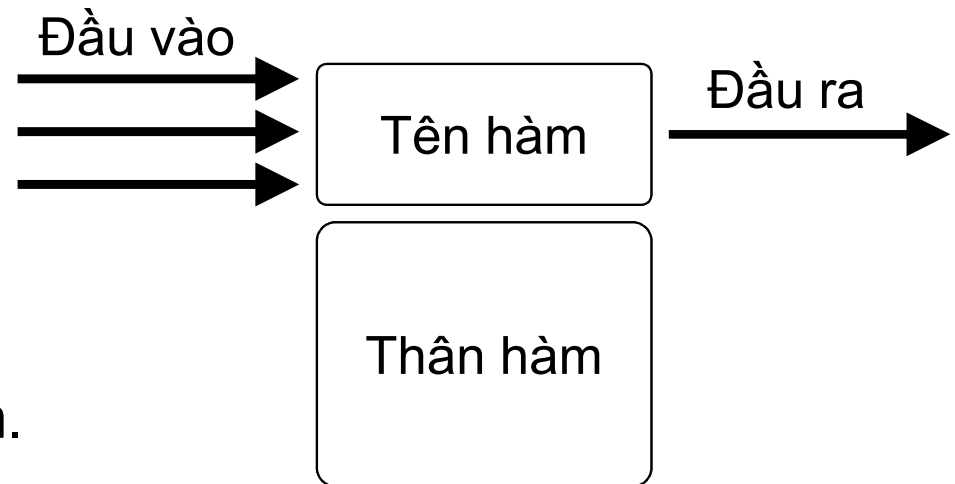
- Một khối lệnh được đặt tên.
- Có thể gọi từ bất kỳ đâu trong chương trình.
- Có thể gọi nhiều lần với tham số khác nhau.

■ Cấu trúc của hàm:

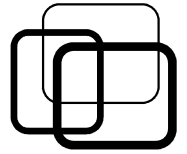
➤ Phần khai báo:

- Tên hàm.
- Tham số đầu vào.
- Kết quả đầu ra.
- ➔ Định danh hàm.

➤ Phần cài đặt: thân hàm.



Khái niệm hàm



■ Các bước sử dụng hàm trong C:

■ Khai báo hàm (prototype):

<Kiểu trả về> <Tên hàm>(<Khai báo tham số>);

*<Kiểu trả về>: int, float, char, ..., **void** (không trả về).*

*float **tinhDTB**(float van, float toan);*

*void **xuatKetQua**();*

■ Cài đặt hàm:

<Kiểu trả về> <Tên hàm>(<Khai báo tham số>)

{

[Các câu lệnh]

*[**return** <giá trị trả về>;]*

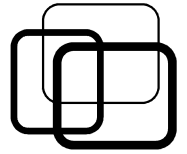
}

■ Gọi thực hiện hàm:

<Tên hàm>(<Các tham số>);

*float dtb = **tinhDTB**(7, 8.5);*

Khái niệm hàm



■ Các bước sử dụng hàm trong C:

```
#include <stdio.h>
```

```
// Khai báo hàm.
```

```
long tingGT(int n);
```

```
void main()
```

```
{
```

```
    /* Viết khai báo biến.
```

```
    Viết lệnh nhập a, b, c. */
```

```
    // Gọi thực hiện hàm.
```

```
    S1 = tingGT(a);
```

```
    S2 = tingGT(b);
```

```
    S3 = tingGT(c);
```

```
    S = S1 + S2 + S3;
```

```
}
```

```
// Cài đặt hàm.
```

```
long tingGT(int n)
```

```
{
```

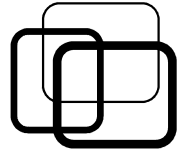
```
    long s = 1;
```

```
    for ( ; n > 0; n--)
```

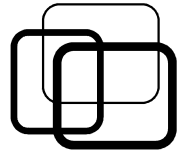
```
        s = s * n;
```

```
    return s;
```

```
}
```



- Khái niệm hàm.
- **Truyền tham số và tầm vực.**
- Tổ chức chương trình C.



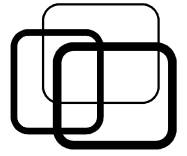
■ Cách truyền tham số vào hàm:

■ Truyền tham trị (pass-by-value):

- Truyền giá trị vào hàm.
- Hàm chỉ nhận bản sao của tham số.
- Tham số KHÔNG THAY ĐỔI sau khi truyền.
- Tham số: biến, hằng, biểu thức.

```
float  tinhDTB( float van, float toan )  
{  
    van = van * 2;  
    toan = toan * 3;  
    return (van + toan) / 5;  
}
```

```
void main()  
{  
    float van, toan, dtb;  
  
    dtb =  tinhDTB( van, toan );  
    dtb =  tinhDTB( 6, 8.5 );  
    dtb =  tinhDTB( van + 1, toan );  
    // Biến van, toan không thay đổi.  
}
```



■ Cách truyền tham số vào hàm:

■ Truyền tham chiếu (pass-by-reference) (C++):

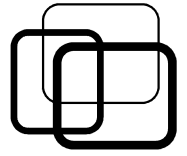
- Còn gọi là truyền tham biến.
- Hàm nhận bản gốc của tham số.
- Tham số CÓ THỂ THAY ĐỔI sau khi truyền.
- Tham số chỉ có thể là biến.
- Khai báo tham số: **&**<tên tham số>.

```
float  tinhDTB( float &van, float toan )    void main()
{
    van = van * 2;
    toan = toan * 3;
    return (van + toan) / 5;
}

{
    float van, toan, dtb;

    dtb =  tinhDTB( van, toan );
    // Biến van bị thay đổi.
    dtb =  tinhDTB( 6, 8.5 );           //Sai
    dtb =  tinhDTB(van + 1, toan); //Sai
}
```

Truyền tham số và tầm vực



■ Cách truyền tham số vào hàm:

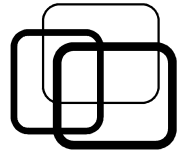
■ Ghi chú:

➤ Dùng truyền tham chiếu để trả về giá trị.

➔ Hàm có nhiều giá trị trả về.

```
void nhapDiem( float &diem1, float &diem2 )  
{  
    printf("Nhap diem van = ");  
    scanf("%d", &diem1);  
    printf("Nhap diem toan = ");  
    scanf("%d", &diem2);  
}  
  
void  tinhDTB( float van, float toan, float &dtb )  
{  
    van = van * 2;  
    toan = toan * 3;  
    dtb = (van + toan) / 5;  
}
```

```
void main()  
{  
    int    van, toan;  
    float dtb;  
  
    // van, toan thay đổi  
    nhapDiem(van, toan);  
  
    // dtb thay đổi  
     tinhDTB(van, toan, dtb);  
}
```



■ Tầm vực:

- Phạm vi hoạt động của biến và hàm.

- Phân loại:

 - Toàn cục: hoạt động trên toàn chương trình.

 - Cục bộ: hoạt động trong một khối lệnh.

- Hàm → phạm vi toàn cục.

- Biến:

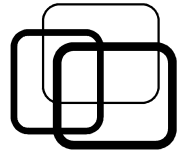
 - Biến toàn cục: khai báo ngoài hàm (kể cả hàm main).

 - ➔ Hoạt động trên toàn chương trình.

 - Biến cục bộ: khai báo trong thân hàm hoặc khối lệnh.

 - ➔ Hoạt động trong thân hàm hoặc khối lệnh.

Truyền tham số và tầm vực



■ Tầm vực:

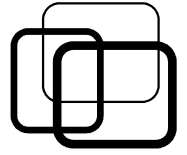
```
float S;           // Khai báo biến toàn cục.  
int tingh();      // Khai báo hàm.
```

```
void main()
```

```
{  
    int a = S + tingh();    // Biến cục bộ hàm main.  
    while (a > 0)  
    {  
        int b = S + tingh(); // Biến cục bộ vòng lặp.  
    }  
}
```

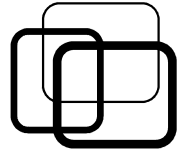
```
int hamXYZ()
```

```
{  
    int y = S * 2;           // Biến cục bộ hàm tingh.  
}
```



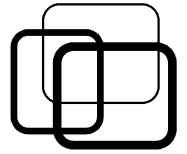
- Khái niệm hàm.
- Truyền tham số và tầm vực.
- **Tổ chức chương trình C.**

Tổ chức chương trình C



- Một quyển sách được tổ chức thế nào?
 - Không thể viết tất cả trên một trang giấy!!
 - ➔ Chia làm nhiều chương.
 - ➔ Có tóm tắt ở đầu.
 - ➔ Nội dung các chương ở sau.

Tổ chức chương trình C



■ Cách tổ chức chương trình C:

■ Tổ chức giống một quyển sách.

- Các chương ~ các file mã nguồn.
- Tóm tắt ~ hàm main.

➔ Làm sao kết nối các file mã nguồn?

// File **main.cpp**

```
void main()
{
    nhap();
    tinhToan1();
    tinhToan2();
    xuat();
}
```

// File **nhapxuat.cpp**

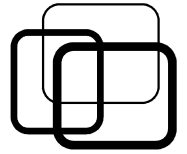
```
void nhap()
{
}

void xuat()
{
}
```

// File **xuly.cpp**

```
int tinhToan1()
{
}

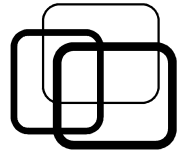
int tinhToan2()
{
}
```



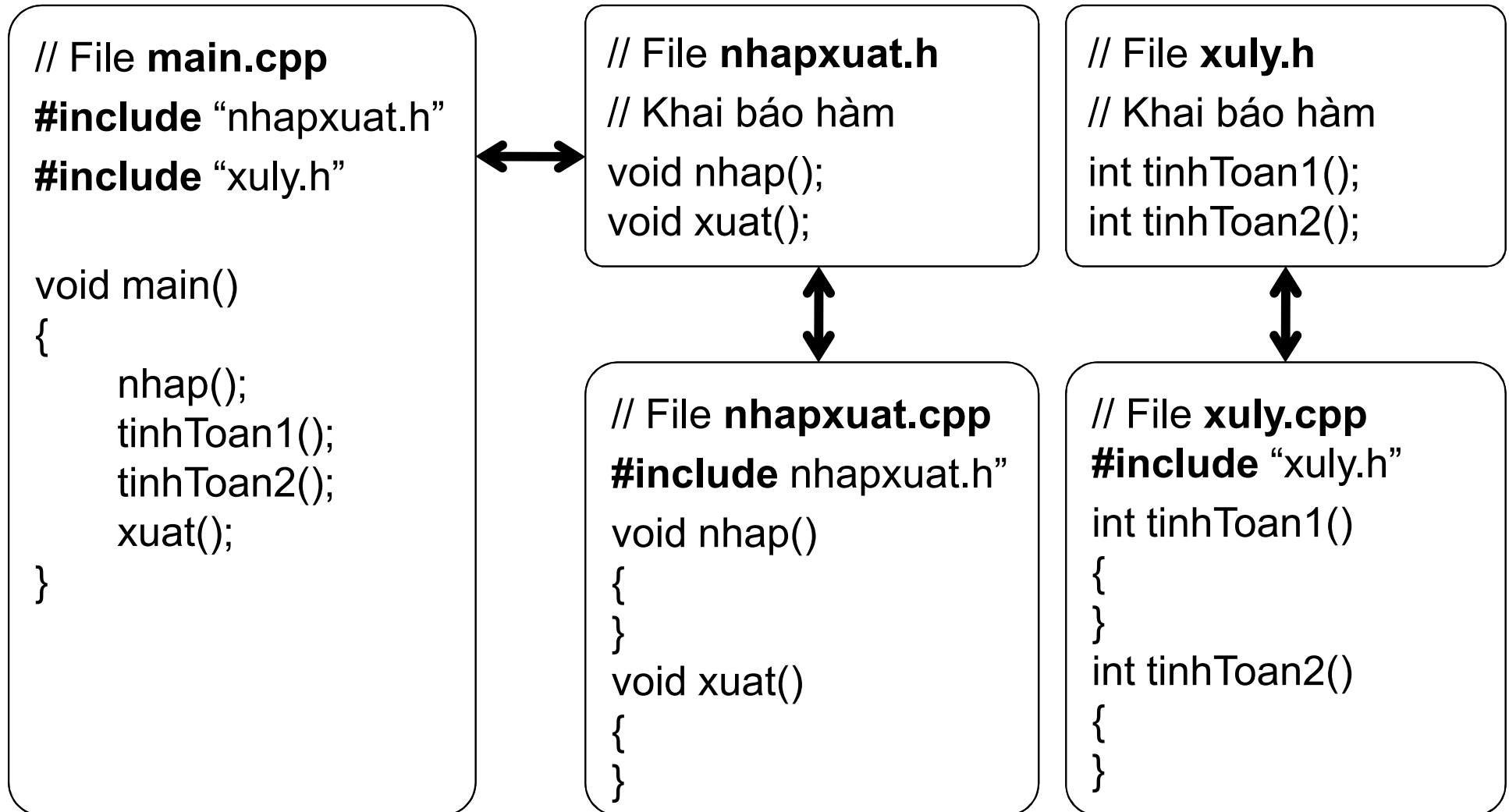
■ Header file:

- Kết nối các file mã nguồn trong chương trình.
- Làm mã nguồn trên các file hiểu lẫn nhau.
- Có đuôi file **.h**.
- Cách sử dụng:
 - Tạo file **.h** cho file mã nguồn **.cpp**.
 - File **.h** chỉ chứa **khai báo** hàm và biến toàn cục.
 - File **.cpp** chỉ chứa **cài đặt** hàm.
 - Để A.cpp hiểu mã nguồn của B.cpp
 - ➔ Trong A.cpp dùng **#include** “<Đường dẫn B.h>”

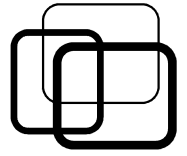
Tổ chức chương trình C



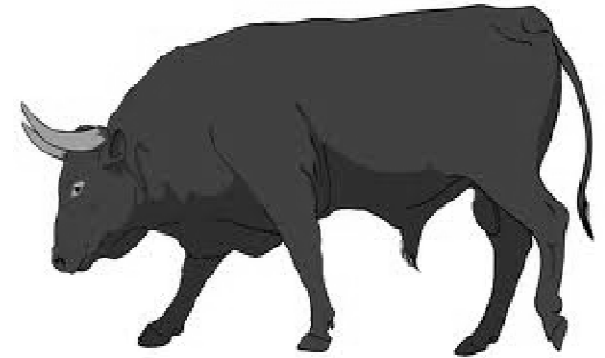
■ Header file:



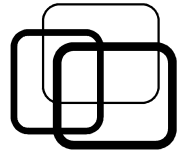
Tổ chức chương trình C



- Phương pháp “chia để trị”:
 - Làm sao để ăn hết một con bò?
 - ➔ Chia thành từng phần nhỏ.
 - ➔ “Giải quyết” từng phần.
 - Phần thế nào là đủ nhỏ?
 - ➔ “Nhai, nuốt” được.
 - Để viết một chương trình:
 - Chia thành từng hàm nhỏ.
 - Xử lý từng hàm.
 - Mỗi hàm nên chỉ 10-20 câu lệnh!!

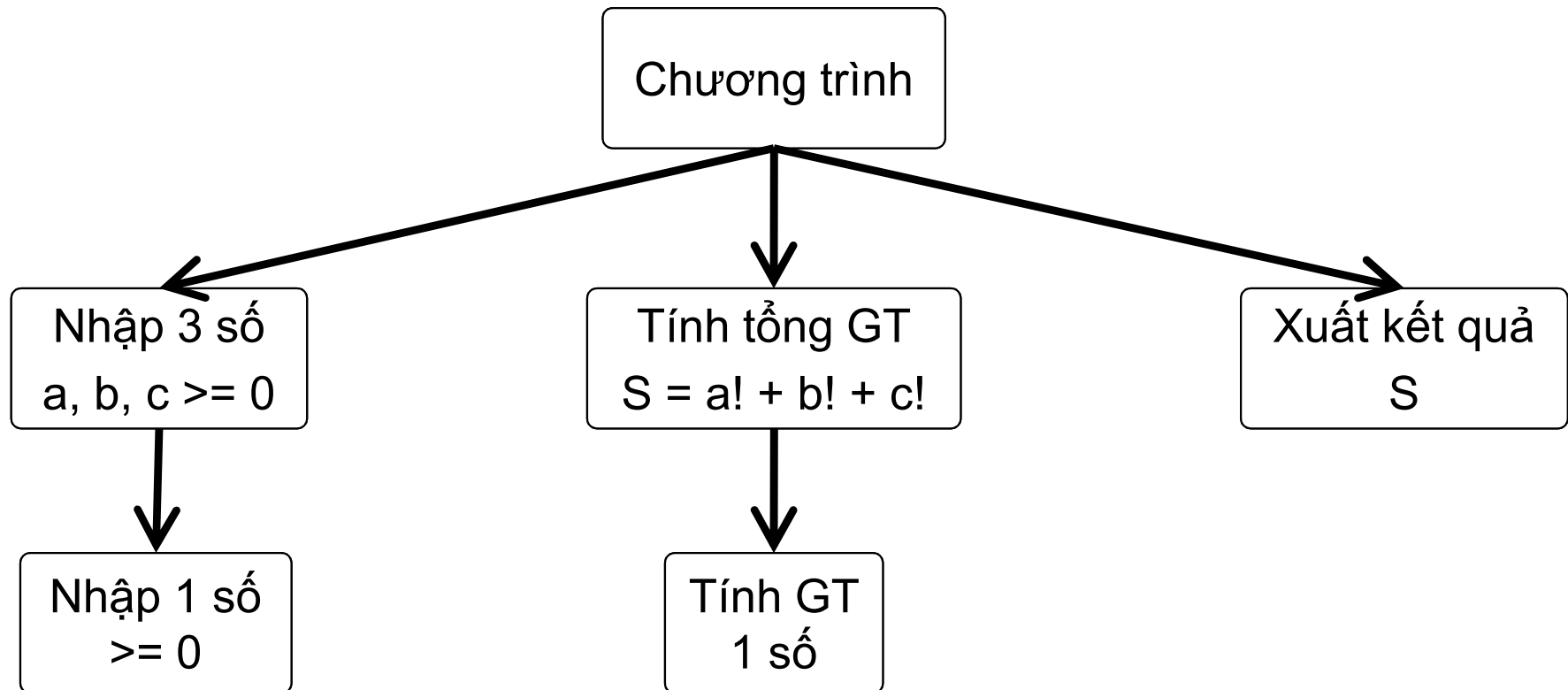


Tổ chức chương trình C

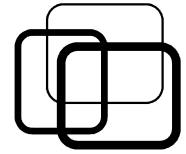


■ Cây phân rã chương trình:

- Nhập vào 3 số nguyên $a, b, c \geq 0$.
- Tính và xuất $S = a! + b! + c!$.



Tổ chức chương trình C



```
// File xuly1.h
void nhap3so(int a, int b, int c);
long tinhTGT(int a, int b, int c);
void xuatKQ(long ketqua);
```

```
// File xuly2.h
void nhap1so(int x);
long tinhGT(int n);
```

```
// File main.cpp
#include "xuly1.h"

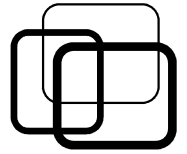
void main()
{
    int a, b, c;
    long S;

    nhap3so(a, b, c);
    S = tinhTGT(a, b, c);
    xuatKQ(S);
}
```

```
// File xuly1.cpp
#include "xuly1.h"
#include "xuly2.h"
void nhap3so(int a, int b, int c)
{
    nhap1so(a);
    nhap1so(b);
    nhap1so(c);
}
long tinhTGT(int a, int b, int c)
{
    return tinhGT(a) +
        tinhGT(b) + tinhGT(c);
}
void xuat(long ketqua)
{
    printf("S = %ld", ketqua);
}
```

```
// File xuly2.cpp
#include "xuly2.h"
#include <stdio.h>
void nhap1so(int &x)
{
    do {
        printf("Nhap 1 so = ");
        scanf("%d", &x);
    } while (x < 0);
}

long tinhGT(int n)
{
    long S = 1;
    for ( ; n > 0; n--)
        S = S * n;
    return S;
}
```



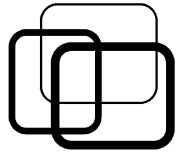
■ Khái niệm hàm:

- Một khối lệnh được đặt tên.
- Có thể gọi từ nhiều nơi trong chương trình.

■ Cách truyền tham số:

- Cách thức đưa giá trị từ bên ngoài vào hàm.
- Truyền tham trị:
 - Truyền giá trị vào hàm.
 - Tham số: biến, hằng, biểu thức.
- Truyền tham chiếu:
 - Truyền biến vào hàm.
 - Tham số: biến.





■ Tầm vực:

- Phạm vi hoạt động của biến và hàm.
- Hàm: toàn cục.
- Biến: toàn cục hoặc cục bộ.

■ Tổ chức chương trình C:

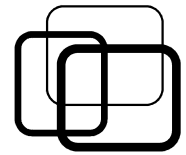
■ Header file:

- File đuôi .h, dùng kết nối các file mã nguồn.
- A.cpp muốn hiểu B.cpp → #include "B.h"

■ “Chia để trị”:

- Chia chương trình thành các hàm.
- Mỗi hàm nên tối đa 10 câu lệnh.
- Cây phân rã chương trình.

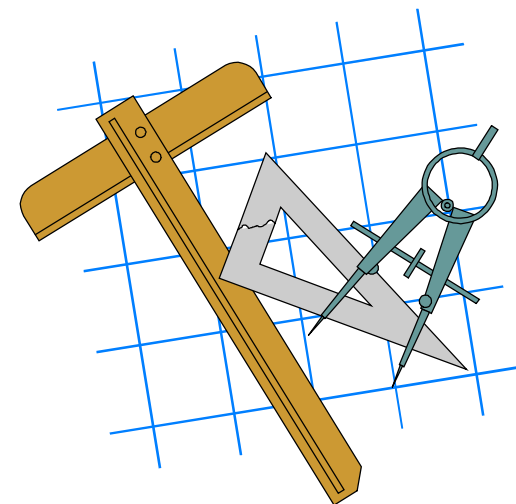


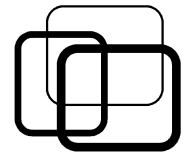


■ Bài tập 6.1:

Viết chương trình C tìm số nguyên tố như sau:
(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào số nguyên $N \geq 0$.
- Xuất tất cả các số nguyên tố $\leq N$.

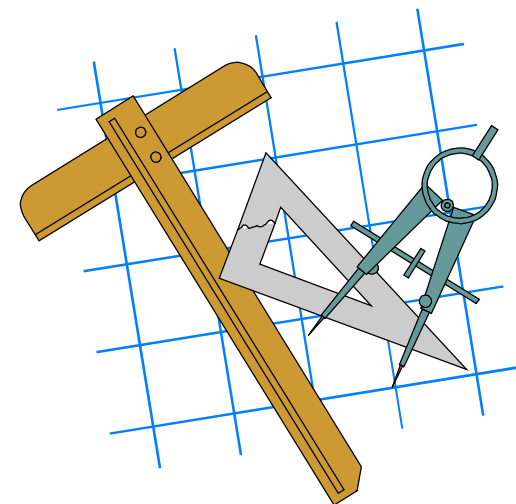


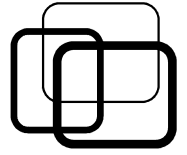


■ Bài tập 6.2:

Viết chương trình C tính toán trên phân số như sau:
(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào 2 phân số a/b và c/d .
- Nhập vào phép tính (+, -, *, /).
- Thực hiện phép tính trên 2 phân số và xuất kết quả.

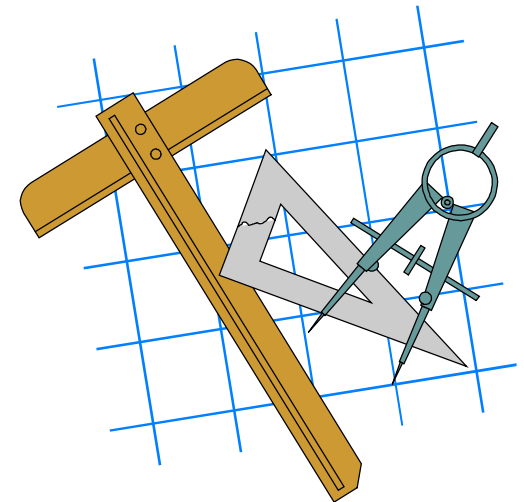


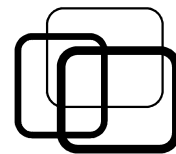


■ Bài tập 6.3:

Viết chương trình C phân loại tam giác như sau:
(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào 3 số thực $a, b, c \geq 0$.
- Kiểm tra xem a, b, c có là 3 cạnh của một tam giác.
- Nếu abc là tam giác, hãy cho biết đó là tam giác gì.





■ Bài tập 6.4:

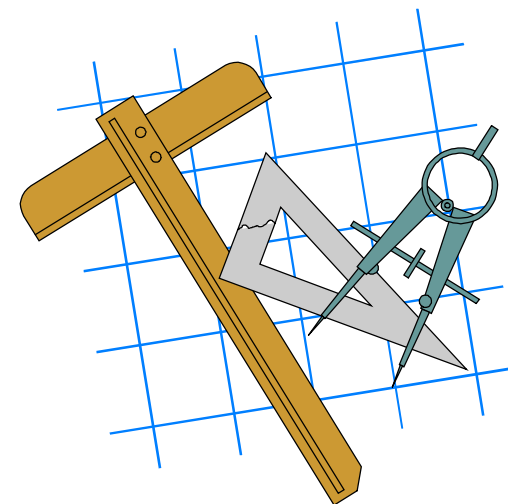
Viết chương trình C tính tiền điện như sau:

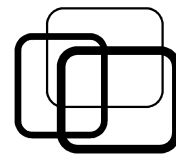
(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào chỉ số điện cũ và chỉ số điện mới (theo kWh) ≥ 0 .
- Tính tiền điện và xuất kết quả.

Biết rằng:

- 100 kWh đầu tiên có đơn giá 1242đ.
- Các kWh từ 101 đến 150 có đơn giá 1304đ.
- Các kWh từ 151 đến 200 có đơn giá 1651đ.
- Các kWh từ 201 đến 300 có đơn giá 1788đ.
- Các kWh từ 301 đến 400 có đơn giá 1912đ.
- Các kWh từ 401 trở lên có đơn giá 1962đ.





■ Bài tập 6.5:

Viết chương trình C mô phỏng menu lựa chọn như sau:
(tổ chức theo dạng hàm và chia làm nhiều file):

- Xuất ra menu như sau:

1. <i>Bai tap 6.1.</i> 2. <i>Bai tap 6.2.</i> 3. <i>Bai tap 6.3.</i> 4. <i>Bai tap 6.4.</i> 5. <i>Thoat.</i>
<i>Lua chon cua ban (1-5):</i>

- Nhập vào lựa chọn là số nguyên từ 1 đến 5.
- Lựa chọn từ 1 đến 4:
 - + Thực hiện bài tập tương ứng.
 - + Trở lại menu để lựa chọn tiếp.
- Lựa chọn là 5: kết thúc chương trình.

