Hàm &
Kỹ thuật tổ chức chương trình
Phần c: Biến toàn cục – Biến cục bộ

Nhập môn lập trình

Trình bày: Nguyễn Sơn Hoàng Quốc
Email: nshquoc@fit.hcmus.edu.vn

cuu duong than cong . com
cuu duong than cong . com

Khái niệm tầm vực của biến

- Là phạm vi hiệu quả của biến khi được khai báo trong chương trình
- Biến cục bộ (local variable)
  - Được khai báo bên trong hàm.
  - Chỉ có tác dụng trong hàm đó.
  - Được khởi tạo bởi một hằng số hoặc một biểu thức tương ứng với kiểu của biến.
  - Biến cục bộ sẽ bị xóa khỏi bộ nhớ ngay khi kết thúc hàm.

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Khái niệm tầm vực của biến

- Biến toàn cục (global variable)
  - Được khai báo bên ngoài tất cả các hàm (kể cả hàm main()).
  - Có tác dụng trên toàn bộ chương trình(!).
  - Được khởi tạo một lần duy nhất bởi một hằng số tương ứng với kiểu của nó trước khi được sử dụng bên trong các hàm (tự động được gán giá trị 0 nếu không khởi gán tường minh).
  - Chỉ được giải phóng khi kết thúc chương trình.

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Ví dụ biến toàn cục, cục bộ

Biến toàn cục

```
int x = 999;
void f();
void main() {
        printf("%d\n", x);
        f();
}
void f() {
        printf("%d\n", x);
}
```

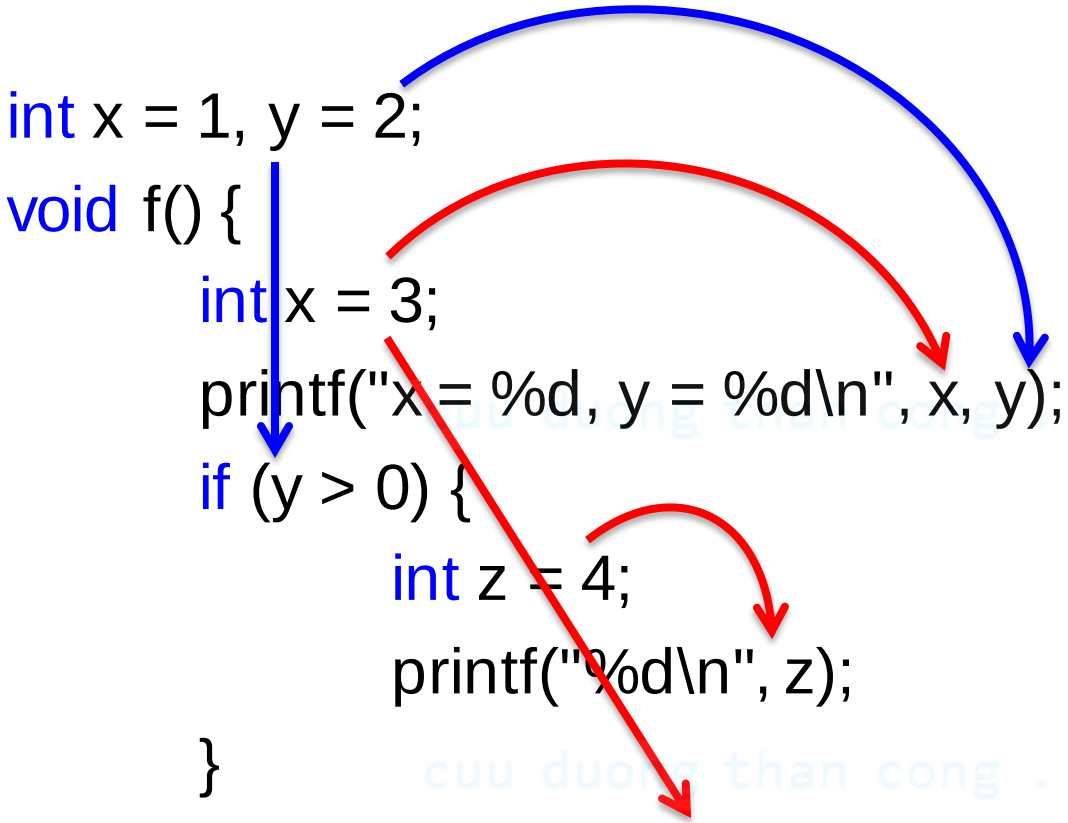
Biến cục bộ

```
void f();
void main() {
        int x = 999;
        printf("%d\n", x);
        f();
}
void f() {
        printf("%d\n", x);
}
```

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Ví dụ biến toàn cục, cục bộ



```
int x = 1, y = 2;
void f() {
        int x = 3;
        printf("x = %d, y = %d\n", x, y);
        if (y > 0) {
                int z = 4;
                printf("%d\n", z);
        }
        printf("x = %d\n", x);
        printf("z = %d\n", z); // error
}
```

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Nói thêm về biến toàn cục

- Biến toàn cục (global variable) là cách gọi khác của biến ngoài (external variable).
- Nói đúng ra, tầm vực của biến ngoài (hay biến toàn cục) là trong toàn bộ mã nguồn của tập tin chứa khai báo biến đó.
- Các chương trình C có kích thước không lớn chỉ được chứa trong một tập tin mã nguồn nên tầm vực là toàn bộ chương trình.
- Biến ngoài được khai báo tường minh bằng từ khóa extern.

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Ví dụ khai báo biến ngoài

Source1.cpp

```
1. int x = 999; // external/global variable
2. void f()
3. {
4.     extern int x;
5.     printf("%d\n", x);
6. }
```

main.cpp

```
1.extern int x;
2.void main()
3.{
4.          printf("%d\n", x);
5.}
```

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Sử dụng biến cục bộ

- Hạn chế sử dụng biến ngoài/toàn cục vì điều này phá vỡ tính độc lập đơn thể (modular independence), nguyên lý trung tâm của lập tình cấu trúc.
- Độc lập đơn thể là ý tưởng mỗi hàm hay đơn thể trong một chương trình chứa tất cả mã nguồn và dữ liệu cần thiết để thực hiện công việc của nó.
- Đối với các chương trình nhỏ thì việc sử dụng chung biến ngoài/toàn cục không quan trọng nhưng khi làm việc với các chương trình lớn hơn và phức tạp hơn thì sự quá ràng buộc vào biến ngoài sẽ nảy sinh nhiều vấn đề rắc rối.

CuuDuongThanCong.com https://fb.com/tailieudientucntt

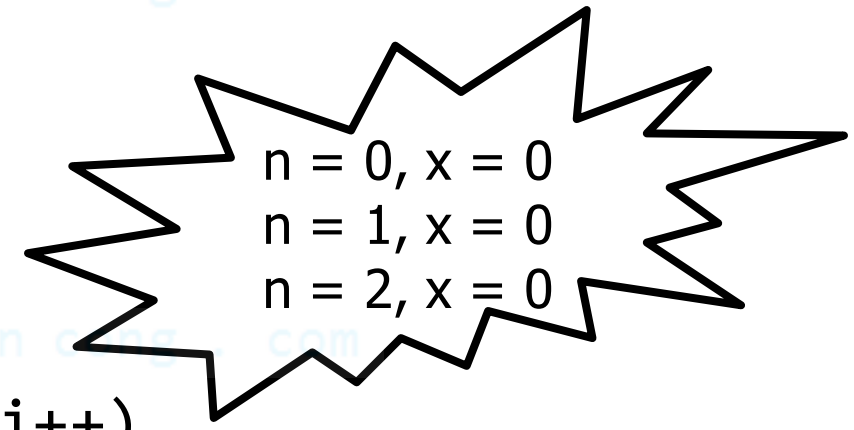
Khái niệm biến cục bộ tĩnh

- Mỗi khi chương trình thực thi lời khai báo biến cục bộ, một bản sao riêng biệt của biến cục bộ đó được tạo ra.
- Nếu biến cục bộ được khai báo là tĩnh (static) thì biến này sẽ được tạo ra một lần duy nhất ở lần đầu tiên khi chương trình thực thi lời khai báo của nó.
- Không như biến toàn cục, biến cục bộ tĩnh không bị truy cập và thay đổi bởi các hàm khác.

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Ví dụ biến cục bộ tĩnh

```
1. void f() {
2.     static int n = 0;      // initialized once
3.     int x = 0;             // initialized n times
4.     printf("n = %d, x = %d\n", n++, x++);
5. }
6.
7. void main() {
8.     int i;
9.     for (i = 0; i < 3; i++)
10.         f();
11.}
```



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Dữ liệu nhập, xuất, trung gian

- Có 3 loại dữ liệu sau khi thực hiện yêu cầu gọi hàm:
  - Dữ liệu nhập: dữ liệu có sẵn, cần thiết để thực hiện hàm, thường được truyền ở dạng tham trị hoặc tham biến.
  - Dữ liệu xuất: dữ liệu hàm tính toán được, thường được trả về bằng lệnh return hoặc ở dạng tham biến.
  - Dữ liệu trung gian: dữ liệu do hàm tạo ra trong quá trình thực hiện công việc, thường phục vụ cho việc tính toán dữ liệu xuất.

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Ví dụ các loại dữ liệu

```
1.  // returns f(x, y) = ax + by and reverses the signs of
    a, b if f < 0
2.  int Calculate(float &a, float &b, float x, float y) {
3.      int temp1, temp2, f;
4.      temp1 = a * x;
5.      temp2 = b * y;
6.      f = temp1 + temp2;
7.      if (f < 0) {
8.              a = -a;
9.              b = -b;
10.     }
11.     return f;
12. }
```

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt