1. **Tìm hiểu về các cơ chế lưu chữ biến trong C**

Trong ngôn ngữ lập trình C, các biến được lưu trữ dưới dạng các vùng nhớ tại thời điểm biên dịch hoặc khi chương trình chạy. Có một số cơ chế lưu trữ ính trong Cch

**-Biến cục bộ (Local variables):** Được khai báo trong phạm vi của một hàm hoặc khối mã. Chúng chỉ tồn tại và có thể truy cập trong phạm vi mà chúng được khai báo.

**-Biến toàn cục (Global variables):** Được khai báo ngoài các hàm, nghĩa là chúng có thể được truy cập từ bất kỳ nơi nào trong chương trình. Biến toàn cục tồn tại trong toàn bộ vòng đời của chương trình.

**-Tham số (Parameters):** Là các biến được truyền vào một hàm và được sử dụng trong phạm vi hàm đó. Chúng tồn tại trong quá trình thực hiện hàm và biến mất khi hàm kết thúc.

**-Vùng nhớ động (Dynamic memory):** Cấp phát và giải phóng bộ nhớ theo yêu cầu trong quá trình thực thi chương trình. malloc(), calloc(), realloc(), và free() là các hàm thường được sử dụng để quản lý vùng nhớ động

**-Vùng nhớ tĩnh (Static memory):** Được sử dụng để lưu trữ biến trong suốt vòng đời của chương trình. Biến static có thể được khai báo trong một hàm để giữ giá trị của biến sau khi hàm kết thúc.

Các cơ chế lưu trữ này định nghĩa vùng nhớ và phạm vi sử dụng của biến, từ đó giúp quản lý dữ liệu trong chương trình C.

1. **tìm hiểu về kiểu dữ liệu boolean trong C**

Trong ngôn ngữ lập trình C, kiểu dữ liệu boolean không được hỗ trợ mặc định như trong một số ngôn ngữ khác như C++, Java hoặc Python. Thay vào đó, trong C, kiểu dữ liệu boolean thường được đại diện bằng các giá trị số nguyên.

Thông thường, các giá trị sau đây thường được sử dụng để biểu thị true và false:

0: Đại diện cho giá trị false.

Bất kỳ giá trị số nguyên khác 0: Đại diện cho giá trị true.

Để cải thiện độ rõ ràng và dễ đọc của mã, người lập trình có thể định nghĩa các hằng số hoặc sử dụng các tiền xử lý để tạo ra các định nghĩa về boolean:

Hoặc:

Sử dụng enum có thể tạo ra một kiểu dữ liệu mới có thể chứa các giá trị TRUE và FALSE, cung cấp tính nhất quán và dễ đọc hơn.

Khi lập trình trong C, các phép toán logic có thể trả về giá trị số nguyên 0 hoặc 1 tương ứng với false và true. Điều này cho phép bạn sử dụng các toán tử logic (ví dụ như && cho AND, || cho OR) để thực hiện kiểm tra điều kiện.

**3.Tìm hiểu về toán tử 3 ngôi trong lập trình C**

* **Toán tử ba ngôi** (ternary operator) là một trong những khái niệm quan trọng trong lập trình C. Toán tử ba ngôi cho phép thực hiện đồng thời dựa trên một điều kiện và cung cấp một giá trị trả về tùy thuộc vào kết quả **đúng/sai**của điều kiện đó. Toán tử ba ngôi giúp tạo mã ngắn gọn hơn và dễ đọc hơn trong một số trường hợp, ngoài ra nó cũng có thể sử dụng thay thế cho câu lệnh**if else** trong một số trường hợp đơn giản. Trong bài viết này, chúng ta sẽ tìm hiểu cách sử dụng toán tử ba ngôi trong C/C++.
* Trong đó:
* condition: Là biểu thức điều kiện(Ví dụ: a > 5, b > 5, b<5, a >b….).
* value\_if\_true: Gía trị trả về nếu biểu thức điều kiện là đúng
* value\_if\_false: Giá trị trả về nếu biểu thức điều kiện là sai.

=> *value\_if\_true, value\_if\_false* cũng có thể là 1 toán tử ba ngôi cấp 2.

Nói tóm lại biểu thức ba ngôi sẽ trả về hay thực hiện tham số thứ nhất(sau dấu ?) nếu đk là đúng, tham số thứ 2(sau dấu :) nếu đk là sai.

**4.Tại sao khi đang nhập số mà chuyển sang nhập chuỗi lại bị bỏ qua và lệnh fflush**

* Thông thường, khi nhập một chuỗi trong màn hình console, ta phải có thao tác xóa bộ nhớ đệm bàn phím. Nếu không có thể thấy rằng kết quả nhập chuỗi bị sai hoặc trôi đi mất.
* Trong quá trình chạy chương trình ta sẽ phải nhập bằng bàn phím, mọi ký tự bạn gõ vào bàn phím (kể cả ký tự Enter \n) đều được đẩy vào bộ nhớ đệm trước khi được gán vào biến. Nếu trước đó bạn có nhập số bằng scanf hoặc cin, chúng chỉ nhận số chứ không nhận được ký tự Enter, và ký tự Enter vẫn còn trong bộ nhớ đệm. Đến khi nhập chuỗi, các hàm nhập chuỗi nhận được ký tự Enter thì dừng nhập luôn và chương trình vẫn chạy tiếp. Điều này khiến kết quả bị sai.
* Bạn có thể sử dụng các hàm sau để thực hiện xóa bộ nhớ đệm”fflush(stdin)”

+ Hàm fflush() trong thư viện stdio.h cũng có tác dụng tương tự flushall(). Tuy nhiên nó cho phép lựa chọn xóa bộ nhớ đệm cho stream nào. Ở đây ta truyền vào stdin để xóa bộ đệm cho dòng nhập chuẩn, tức là bàn phím.