Bài tự tìm hiểu

Câu 1 Cách lưu trữ biến trong bộ nhớ C

Trong C, các biến được lưu trữ trong bộ nhớ của máy tính. Khi một biến được khai báo, một vùng nhớ được cấp phát cho nó trong bộ nhớ. Các biến có thể được lưu trữ trong các vùng nhớ khác nhau, bao gồm:

1. Stack: Đây là nơi lưu trữ các biến cục bộ và các giá trị trả về của các hàm. Khi một hàm được gọi, các biến cục bộ của nó được lưu trữ trên stack. Khi hàm kết thúc, các biến này sẽ bị xóa khỏi stack.

2. Heap: Đây là nơi lưu trữ các biến được cấp phát động bằng cách sử dụng các hàm như malloc() và calloc(). Các biến này không được lưu trữ trên stack và sẽ tồn tại cho đến khi được giải phóng bằng cách sử dụng hàm free().

3. Data segment: Đây là nơi lưu trữ các biến toàn cục và các biến tĩnh. Các biến này được khai báo bên ngoài các hàm và sẽ tồn tại trong suốt quá trình chạy chương trình.

4. Code segment: Đây là nơi lưu trữ mã máy của chương trình. Các biến không được lưu trữ trong code segment.

Khi sử dụng các biến trong chương trình C, bạn cần phải biết vị trí lưu trữ của chúng để tránh các lỗi như tràn bộ nhớ hoặc truy cập vào vùng nhớ không hợp lệ.

Câu 2 Tìm hiểu bool và cách dùng bool trong C

Kiểu dữ liệu bool trong C được sử dụng để đại diện cho giá trị logic true hoặc false. Kiểu dữ liệu bool được định nghĩa trong thư viện stdbool.h và có hai giá trị có thể được gán cho nó là true và false.

Để sử dụng kiểu dữ liệu bool trong C, bạn cần bao gồm thư viện stdbool.h bằng cách sử dụng lệnh #include. Sau đó, bạn có thể khai báo một biến bool bằng cách sử dụng từ khóa bool và gán giá trị true hoặc false cho nó.

Ví dụ:

#include <stdbool.h>

#include <stdio.h>

int main() {

bool isTrue = true;

bool isFalse = false;

printf("isTrue = %d\n", isTrue);

printf("isFalse = %d\n", isFalse);

return 0;

}

Kết quả khi chạy chương trình sẽ là:

isTrue = 1

isFalse = 0

Trong đó, giá trị 1 tương đương với true và giá trị 0 tương đương với false.

Bạn có thể sử dụng kiểu dữ liệu bool trong các biểu thức logic và các câu lệnh điều kiện như if và while. Ví dụ:

bool isRainy = true;

if (isRainy) {

printf("It's raining today.\n");

} else {

printf("It's not raining today.\n");

}

Kết quả khi chạy chương trình sẽ là:

It's raining today.

Trong đó, biểu thức isRainy được đánh giá là true, vì nó được gán giá trị true trước đó.

Câu 3 Tìm hiểu toán tử 3 ngôi trong C

Toán tử 3 ngôi trong C là một toán tử điều kiện, được sử dụng để thay thế cho câu lệnh if-else trong một số trường hợp đơn giản. Toán tử 3 ngôi có cú pháp như sau:

(condition) ? expression1 : expression2;

Trong đó, condition là một biểu thức điều kiện, expression1 và expression2 là hai biểu thức có thể có giá trị khác nhau. Nếu condition đúng, toán tử sẽ trả về giá trị của expression1, ngược lại nếu condition sai, toán tử sẽ trả về giá trị của expression2.

Ví dụ#include <stdio.h>

int main() {

int a = 10, b = 20, max;

max = (a > b) ? a : b;

printf("Max value is %d\n", max);

return 0;

}

Kết quả khi chạy chương trình sẽ là:

Max value is 20

Trong đó, biểu thức (a > b) được đánh giá là false, vì vậy toán tử sẽ trả về giá trị của biến b, tức là 20.

Toán tử 3 ngôi cũng có thể được sử dụng để gán giá trị cho một biến dựa trên một điều kiện. Ví dụ:

#include <stdio.h>

int main() {

int a = 10, b = 20, max;

max = (a > b) ? a : b;

printf("Max value is %d\n", max);

max = (a < b) ? a : b;

printf("Min value is %d\n", max);

return 0;

}

Kết quả khi chạy chương trình sẽ là:

Max value is 20

Min value is 10

Trong đó, biểu thức (a < b) được đánh giá là true, vì vậy toán tử sẽ trả về giá trị của biến a, tức là 10.

Câu 4 Tại sao nhập đang nhập số mà chuyển sang nhập chữ thì không được

Và dùng lệnh fflush

Khi nhập số, dữ liệu được lưu trữ dưới dạng kiểu số (integer hoặc float), trong khi khi nhập chuỗi, dữ liệu được lưu trữ dưới dạng kiểu ký tự (character array). Các kiểu dữ liệu này có cách lưu trữ và xử lý khác nhau, do đó khi bạn nhập số và chuyển sang nhập chuỗi, chương trình sẽ không thể xử lý dữ liệu đó theo cách bạn mong muốn.

Ngoài ra, khi nhập số, bạn cần phải kiểm tra tính hợp lệ của dữ liệu đầu vào, ví dụ như kiểm tra xem người dùng đã nhập số hay chưa, hoặc kiểm tra giá trị nhập vào có nằm trong phạm vi cho phép hay không. Nếu bạn nhập chuỗi thay vì số, các kiểm tra này sẽ không thể thực hiện được, dẫn đến lỗi trong quá trình xử lý dữ liệu.

Vì vậy, khi nhập số, bạn nên sử dụng các hàm nhập số như scanf() hoặc atoi() để đảm bảo tính hợp lệ của dữ liệu đầu vào.

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.