

```

#include <stdio.h>
int foo(int*a,int*b)
{
    int sum=*a+*b;
    *b=*a;
    return *a=sum-*b;
}
int main()
{
    int i=0,j=1,k=2,l;
    l=i++ || foo(&j,&k);
    printf("%d\t%d\t%d\t%d",i,j,k,l);
    return 0;
}

```

Giải thích chi tiết:

1. Hàm `foo(int *a, int *b)`:

- Khi hàm `foo` được gọi với `a` trỏ đến `j` và `b` trỏ đến `k`, hàm thực hiện các bước sau:
 - `sum = *a + *b` tương đương với `sum = j + k = 1 + 2 = 3`.
 - `*b = *a`, gán giá trị `*a` (giá trị của `j`) cho `*b` (giá trị của `k`), tức là `k = 1`.
 - `*a = sum - *b`, gán `j = sum - k = 3 - 1 = 2`.
- Sau khi thực hiện, `foo` trả về `2`.

Kết quả của hàm `foo(&j, &k)` là:

- `j = 2`
- `k = 1`
- Giá trị trả về là `2`.

2. Trong `main()`:

- Khởi tạo các biến: `i = 0`, `j = 1`, `k = 2`.
- Dòng `l = i++ || foo(&j, &k);` thực hiện:
 - `i++` là một phép toán hậu tố, tức là `i` có giá trị 0 khi được sử dụng trong biểu thức, sau đó `i` tăng lên thành 1.
 - `i++` có giá trị 0 (tức là sai), do đó, hàm `foo(&j, &k)` được gọi để đánh giá phần còn lại của biểu thức.
 - `foo(&j, &k)` trả về `2`, là giá trị khác 0 (tức là đúng), vì vậy giá trị của `l` là `1`.

3. Kết quả cuối cùng:

- Sau khi biểu thức `l = i++ || foo(&j, &k);` được thực hiện:
 - `i = 1`
 - `j = 2`
 - `k = 1`
 - `l = 1`