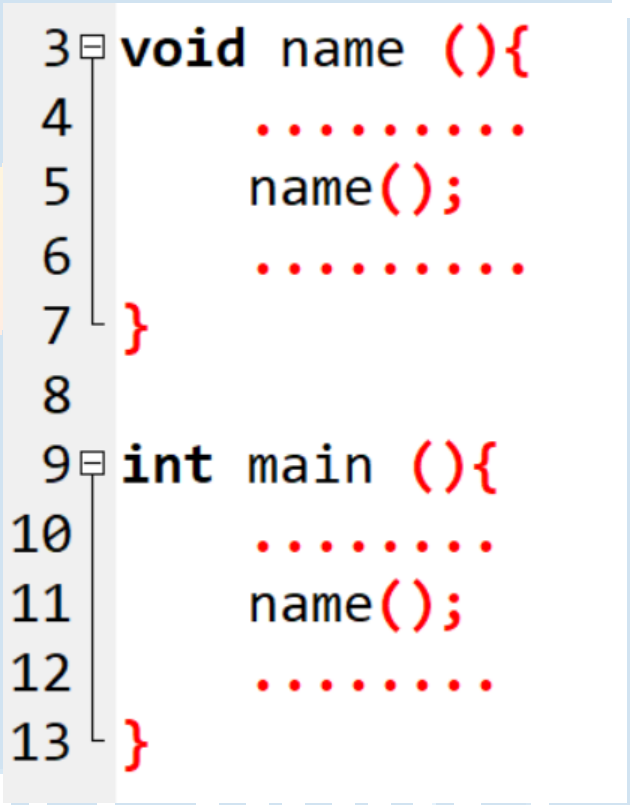
BUỔI 10 – FULL HOUSE

I. Đệ quy

a. Định nghĩa

Hàm đệ quy trong C là các hàm mà **bản thân nó** có khả năng **gọi lại chính nó**.



b. Tính dừng

Để tránh việc **lặp vô tận**, hàm đệ quy cần có tính dừng: Nếu gặp 1 điều kiện nào đó, nó cần phải **dừng lại việc tự gọi lại chính mình**. Và tính dừng là điều **bắt buộc** phải có trong 1 hàm đệ quy trong C cùng như mọi ngôn ngữ khác.

c. Bài tập vd:

1. Tính tổng từ 1 tới n *(Bài 10.1)*

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

int tong(int n){
    if(n==1) return 1;
    return n + tong(n-1);
}

int main(){
    int n;
    scanf("%d",&n);
    printf("%d",tong(n));
}
```

2. Tính n! bằng đệ quy. *(Bài 10.5)*

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

int tong(int n){
    if(n==1) return 1;
    return n * tong(n-1);
}

int main(){
    int n;
    scanf("%d",&n);
    printf("%d",tong(n));
}
```

3. Đếm số lượng chữ số của n *(Bài 10.4)*

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

int dem(int n){
    if(n<10) return 1;
    return 1 + dem(n/10);
}

int main(){
    int n;
    scanf("%d",&n);
    printf("%d",dem(n));
}
```

4. Tìm ước chung lớn nhất của 2 số nguyên dương. *(Bài 10.7)*

C1:

```c
int UCLN(int n, int m){
    if(!(n%m)) return m;
    if(!(m%n)) return n;
    if(m>n) UCLN (n,m-n);
    else UCLN (n-m,m);
}
```

C2:

```
int UCLN(int n, int m){
    if(!m) return n;
    return UCLN (m,n%m);
}
```

5. In ra số fibonacci thứ n *(Bài 10.3)*

```
91  #include <stdio.h>
92
93  int fibo(int n){
94      if(n==1||n==2) return 1;
95      return fibo(n-1) + fibo(n-2);
96  }
97
98  int main(){
99      int n;
100     scanf("%d",&n);
101     printf("%d",fibo(n));
102 }
```

II. Thao tác với bit

a. Một số toán tử

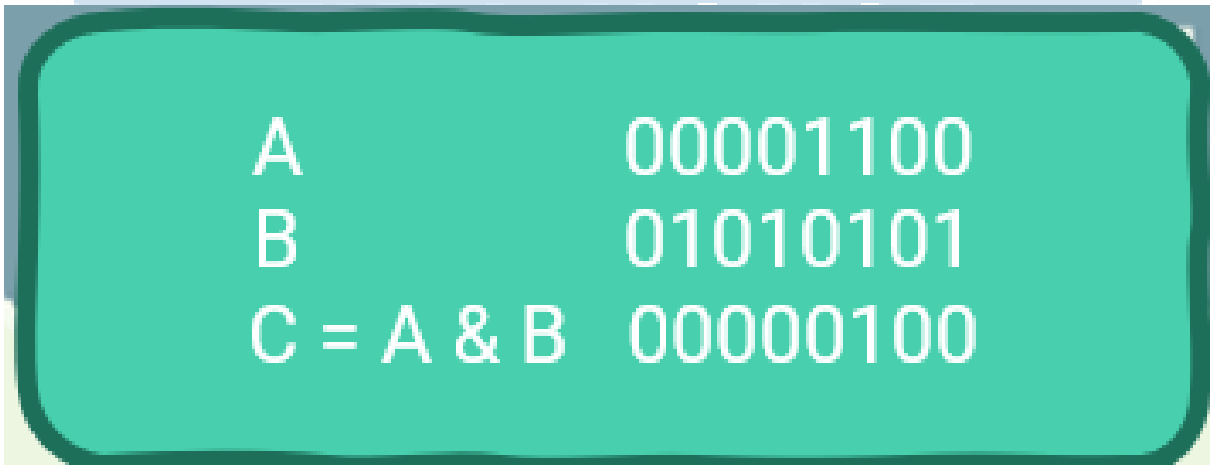
Phép toán thao tác trên bit	Kí hiệu
Phép AND	&
Phép OR	|
Phép phủ đinh NOT	~
Phép XOR	^
Phép dịch trái	<<
Phép dịch phải	>>

i. Phép AND

- Kí hiệu: &

A	B	A & B
0	0	0
0	1	0
1	0	0
1	1	1

- Ví dụ:

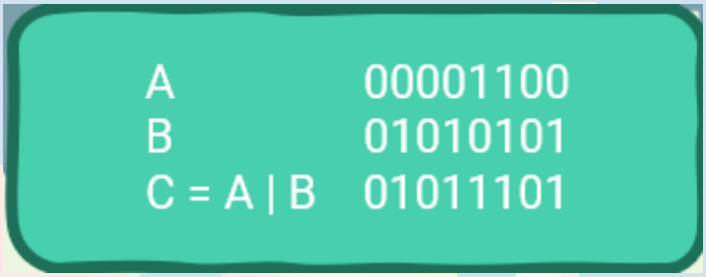


ii. Phép or

- Kí hiệu |

A	B	A | B
0	0	0
0	1	1
1	0	1
1	1	1

- Ví dụ:

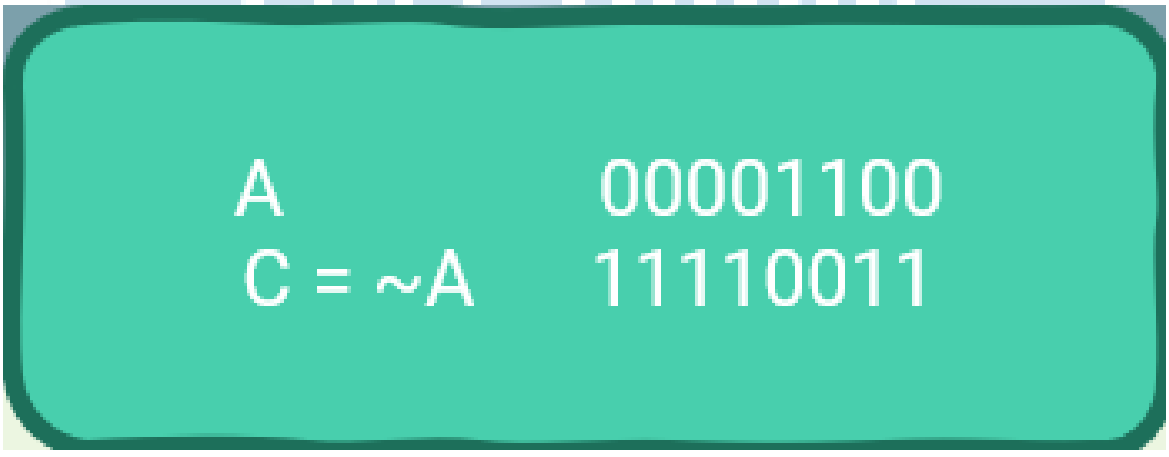


iii. Phép phủ định NOT

- Kí hiệu: ~

A	~A
0	1
1	0

- Ví dụ:

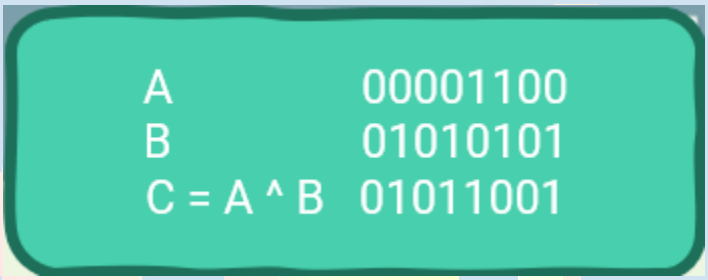


iv. Phép XOR

- Kí hiệu: ^

A	B	A ^ B
0	0	0
0	1	1
1	0	1
1	1	0

- Ví dụ:

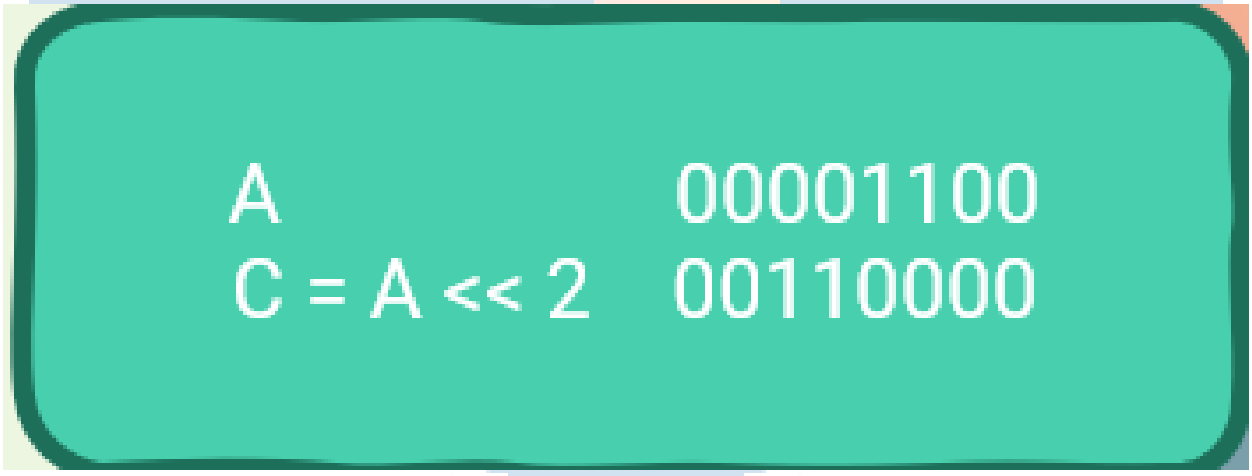


v. Phép dịch trái <<

- Kí hiệu: <<

#Phép dịch trái n bit tương đương với phép nhân cho 2^n.

- Ví dụ:

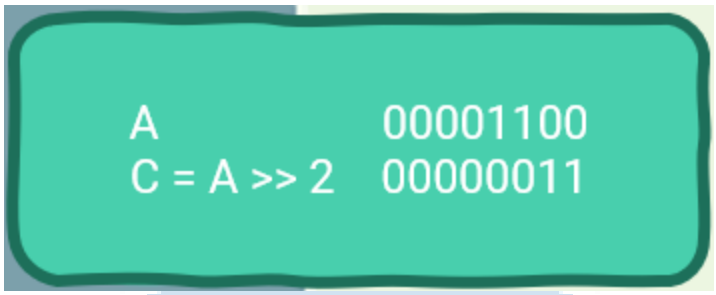


vi. Phép dịch phải >>

- Kí hiệu: >>

#Phép dịch phải n bit tương đương với phép chia cho 2^n.

- Ví dụ:



III. Chuyển đổi hệ thập phân sang nhị phân.

```
2  #include <stdio.h>
3
4  void Nhi_phan (int n){
5      if(!n) return;
6      Nhi_phan (n/2);
7      printf("%d",n%2);
8  }
9
10 int main(){
11     int n;
12     scanf("%d",&n);
13     Nhi_phan(n);
14 }
```

IV. Chuyển đổi hệ nhị phân sang thập phân

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

int main(){
    int n;
    scanf("%d",&n);
    int d=0;
    int res =0;
    while(n){
        res += n%10 * pow(2,d++);
        n /= 10;
    }
    printf("%d",res);
}
```