



# HUST

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.





**ĐẠI HỌC  
BÁCH KHOA HÀ NỘI**  
HANOI UNIVERSITY  
OF SCIENCE AND TECHNOLOGY

# C BASIC

ĐỆ QUY

ONE LOVE. ONE FUTURE.

# NỘI DUNG

- Đề quy
- Bài toán ước số chung lớn nhất (P.02.04.01)
- Bài toán đổi số nguyên sang chuỗi bit nhị phân (P.02.04.02)
- Bài toán tháp Hà Nội (P.02.04.03)
- Đề quy có nhớ
- Bài toán tính dãy số Fibonacci (P.02.04.04)
- Bài toán tính hằng số tổ hợp (P.02.04.05)



# ĐỆ QUY

- Đối tượng có cấu trúc đệ quy là đối tượng được định nghĩa/xây dựng qua chính nó với quy mô nhỏ hơn
- Hàm đệ quy là hàm đưa ra lời gọi đến chính nó với quy mô tham số nhỏ hơn
- Thuật toán đệ quy (thường được thể hiện bởi 1 hàm đệ quy) phù hợp để thực hiện các xử lý, tính toán trên các đối tượng có cấu trúc đệ quy

$$F(n) = \begin{cases} F(n-1) + n, & n \geq 2 \\ 1, & \text{khi } n = 1 \end{cases}$$

$$F(n) = \begin{cases} F(n-1) + F(n-2), & n \geq 2 \\ n, & \text{khi } n = 0, 1 \end{cases}$$

$$F(a,b) = \begin{cases} a, & \text{nếu } a = b \\ F(a-b, b), & \text{nếu } a > b \\ F(a, b-a), & \text{nếu } a < b \end{cases}$$

$$C(k, n) = \begin{cases} 1, & \text{khi } k = 0 \text{ hoặc } k = n \\ C(k, n-1) + C(k-1, n-1), & \text{ngược lại} \end{cases}$$

## BÀI TOÁN ƯỚC SỐ CHUNG LỚN NHẤT (P.02.04.01)

- Cho hai số nguyên dương a và b. Hãy viết chương trình tìm ước số chung lớn nhất của a và b.
- Dữ liệu
  - Dòng 1: ghi 2 số nguyên dương a và b ( $1 \leq a, b \leq 100000$ )
- Kết quả
  - Ghi ra ước số chung lớn nhất của a và b

stdin	stdout
16 24	8



# BÀI TOÁN ƯỚC SỐ CHUNG LỚN NHẤT – MÃ GIẢ

- Nếu  $a = b$  thì  $\text{USCLN}(a, b) = a$
- Nếu  $a > b$  thì  $\text{USCLN}(a, b) = \text{USCLN}(a-b, b)$
- Nếu  $a < b$  thì  $\text{USCLN}(a, b) = \text{USCLN}(a, b-a)$

```
F(a, b){  
    if a = b then return a;  
    if a > b then return F(a-b, b);  
    else return F(a, b-a);  
}
```



# BÀI TOÁN ƯỚC SỐ CHUNG LỚN NHẤT – CODE HOÀN CHỈNH

```
#include <stdio.h>

int F(int a, int b){
    if(a == b) return a;
    if(a > b) return F(a-b, b);
    else return F(a, b-a);
}

int main(){
    int a,b; scanf("%d%d",&a,&b);
    printf("%d",F(a,b));
    return 0;
}
```





## BÀI TOÁN ĐỔI SỐ NGUYÊN SANG DÃY BÍT NHỊ PHÂN (P.02.04.02)

- Cho số nguyên dương  $N$ , hãy viết chương trình đổi  $N$  sang chuỗi bit nhị phân (bỏ qua các bit 0 ngoài cùng bên trái)
- Dữ liệu
  - Dòng duy nhất chứa số nguyên dương  $N$  ( $1 \leq N \leq 2 \times 10^7$ )
- Kết quả
  - Dòng duy nhất ghi chuỗi bit kết quả

stdin	stdout
20	10100



# BÀI TOÁN ĐỔI SỐ NGUYÊN SANG DÃY BÍT NHỊ PHÂN – MÃ GIẢ

- Gọi đệ quy để đổi  $N/2$  sang chuỗi bit nhị phân, sau đó ghép kết quả với bit phải nhất (bít phải nhất có giá trị bằng  $N \bmod 2$ )

```
Convert(N){  
    if N = 0 then return;  
    Convert(N/2);  
    b = N mod 2;  
    print(b);  
}
```



# BÀI TOÁN ĐỔI SỐ NGUYÊN SANG DÃY BÍT NHỊ PHÂN – CODE HOÀN CHỈNH

```
#include <stdio.h>

void convert(int N){
    if(N == 0) return;
    int b = N%2;
    convert(N/2);
    printf("%d",b);
}

int main(){
    int N; scanf("%d",&N);
    convert(N);
    return 0;
}
```



## BÀI TOÁN THÁP HÀ NỘI (P.02.04.03)

- Cho  $n$  đĩa có bán kính khác nhau và 3 cọc A, B, C. Ban đầu  $n$  đĩa nằm ở cọc A theo thứ tự đĩa nhỏ ở trên và đĩa lớn ở dưới. Hãy tìm cách chuyển  $n$  đĩa từ cọc A sang cọc B (dùng cọc C làm trung gian) theo nguyên tắc
  - Mỗi bước chỉ được chuyển 1 đĩa trên cùng từ 1 cọc sang 1 cọc khác (đặt trên cùng)
  - Không được phép để xảy ra trường hợp đĩa lớn nằm bên trên đĩa bé ở 1 cọc nào đó
- Dữ liệu
  - Dòng duy nhất chứa 4 số nguyên dương  $n, A, B, C$  ( $1 \leq n \leq 20, 1 \leq A, B, C \leq 100$ )
- Kết quả
  - Dòng 1 ghi số nguyên dương  $m$  (số bước thực hiện)
  - Dòng  $i + 1$  ( $i = 1, 2, \dots, m$ ) chứa 2 số nguyên dương  $X$  và  $Y$ : tại bước  $i$ , ta chuyển 1 đĩa từ cọc  $X$  sang cọc  $Y$

stdin	stdout
2 11 22 33	3
	11 33
	11 22
	33 22



# BÀI TOÁN THÁP HÀ NỘI – MÃ GIẢ

- Thuật toán
  - Chuyển  $n-1$  đĩa từ cọc A sang cọc C, lấy B làm cọc trung gian
  - Chuyển 1 đĩa từ cọc A sang cọc B
  - Chuyển  $n-1$  đĩa từ cọc C sang cọc B, lấy A làm cọc trung gian
- Số bước cần thực hiện là  $2^n - 1$

```
move(n, A, B, C){  
    if n = 1 then print(A, B);  
    else {  
        move(n-1, A, C, B);  
        move(1, A, B, C);  
        move(n-1, C, B, A);  
    }  
}
```



# BÀI TOÁN THÁP HÀ NỘI – CODE HOÀN CHỈNH

```
#include <stdio.h>

int n;
int A, B, C;

void move(int n, int A, int B, int C){
    if(n==1){
        printf("%d %d\n",A,B); return;
    }
    move(n-1,A,C,B);
    move(1,A,B,C);
    move(n-1,C,B,A);
}
```

```
int main(){
    scanf("%d%d%d", &n, &A, &B, &C);
    int step = 1;
    for(int i = 1; i <= n; i++) step = step*2;
    step = step - 1;
    printf("%d\n", step);
    move(n,A,B,C);
    return 0;
}
```



# BÀI TOÁN TÍNH DÃY SỐ FIBONACCI (P.02.04.04)

- Cho số nguyên dương  $n$ , hãy tính số Fibonacci thứ  $n$
- Dữ liệu
  - Dòng 1: chứa số nguyên dương  $n$  ( $2 \leq n \leq 100000$ )
- Kết quả
  - Ghi giá trị  $F(n) \bmod 10^9+7$

$$F(n) = \begin{cases} F(n-1) + F(n-2), & n \geq 2 \\ n, & \text{khi } n = 0, 1 \end{cases}$$

stdin	stdout
10	55



# BÀI TOÁN TÍNH DÃY SỐ FIBONACCI – MÃ GIẢI

- Cho số nguyên dương  $n$ , hãy tính số Fibonacci thứ  $n$
- Dữ liệu
  - Dòng 1: chứa số nguyên dương  $n$  ( $2 \leq n \leq 100000$ )
- Kết quả
  - Ghi giá trị  $F(n) \bmod 10^9+7$

```
F(n){  
    if n <= 1 then return n;  
    return (F(n-1) + F(n-2)) mod 109+7;  
}
```





# BÀI TOÁN TÍNH DÃY SỐ FIBONACCI – CODE HOÀN CHỈNH

- Cho số nguyên dương  $n$ , hãy tính số Fibonacci thứ  $n$
- Dữ liệu
  - Dòng 1: chứa số nguyên dương  $n$  ( $2 \leq n \leq 100000$ )
- Kết quả
  - Ghi giá trị  $F(n) \bmod 10^9+7$

```
#include <stdio.h>
#define P 1000000007

int F(int n){
    if(n <= 1) return n;
    return (F(n-1) + F(n-2))%P;
}

int main(){
    int n; scanf("%d",&n);
    printf("%d",F(n));
    return 0;
}
```



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP (P.02.04.05)

- Cho số nguyên không âm  $k$  và  $n$ , hãy tính hằng số tổ hợp  $C(k, n)$
- Dữ liệu
  - Dòng duy nhất chứa 2 số nguyên  $k$  và  $n$  ( $0 \leq k, n \leq 999$ )
- Kết quả
  - Ghi ra giá trị  $C(k, n) \bmod 10^9+7$

$$C(k, n) = \begin{cases} 1, & \text{khi } k = 0 \text{ hoặc } k = n \\ C(k, n-1) + C(k-1, n-1), & \text{ngược lại} \end{cases}$$

stdin	stdout
3 5	10



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP – MÃ GIẢ

- Cho số nguyên không âm  $k$  và  $n$ , hãy tính hằng số tổ hợp  $C(k, n)$
- Dữ liệu
  - Dòng duy nhất chứa 2 số nguyên  $k$  và  $n$  ( $0 \leq k, n \leq 999$ )
- Kết quả
  - Ghi ra giá trị  $C(k, n) \bmod 10^9+7$

```
if k = 0 or k = n then return 1;  
return (C(k-1, n-1) + C(k, n-1)) mod 109+7;  
}
```



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP – CODE HOÀN CHỈNH

- Cho số nguyên không âm  $k$  và  $n$ , hãy tính hằng số tổ hợp  $C(k,n)$
- Dữ liệu
  - Dòng duy nhất chứa 2 số nguyên  $k$  và  $n$  ( $0 \leq k, n \leq 999$ )
- Kết quả
  - Ghi ra giá trị  $C(k,n) \bmod 10^9+7$

```
#include <stdio.h>
#define P 1000000007

int C(int k, int n){
    if(k==0 || k == n) return 1;
    return (C(k-1,n-1) + C(k,n-1))%P;
}

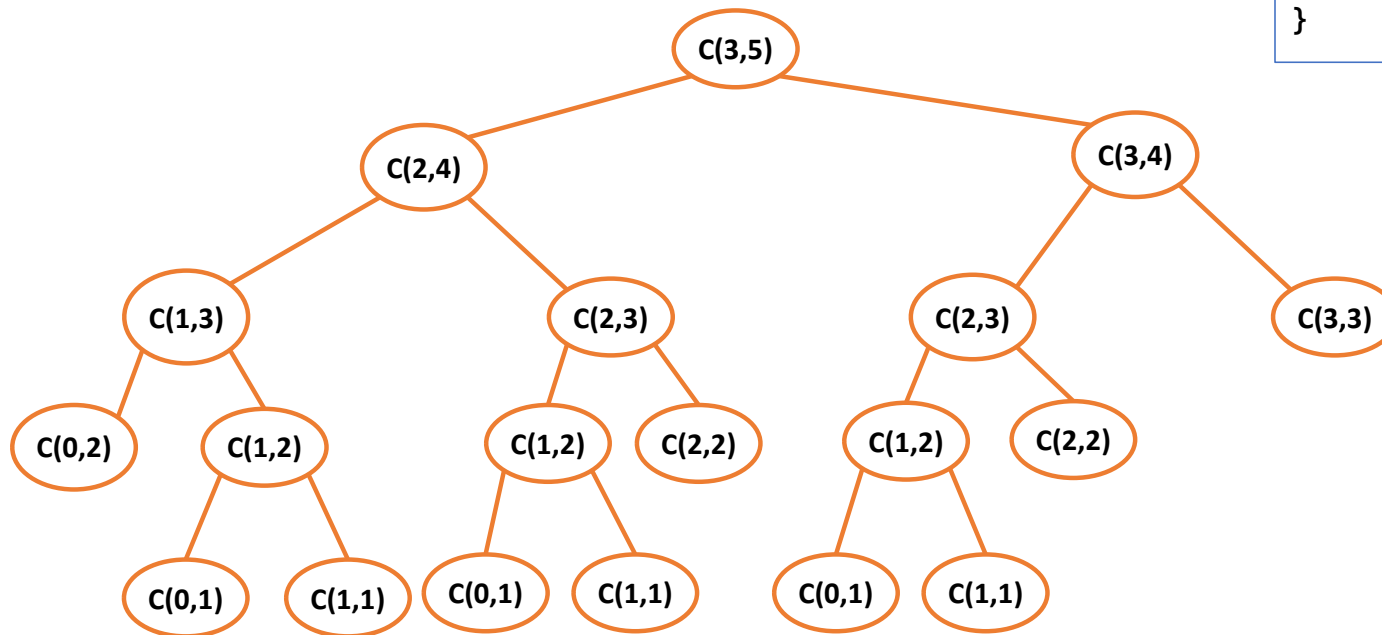
int main() {
    int k,n; scanf("%d%d",&k,&n);
    printf("%d",C(k,n));
    return 0;
}
```



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP – ĐỆ QUY CÓ NHỚ

- Thuật toán đệ quy để tính  $C(k, n)$

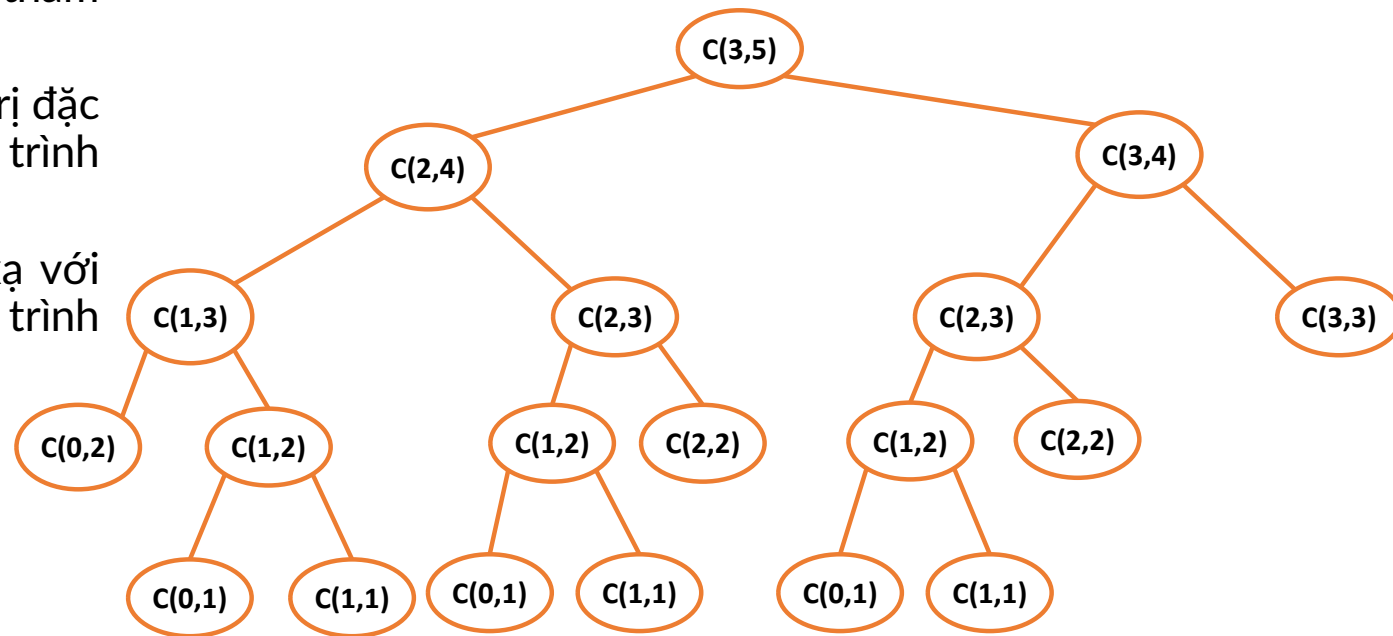
```
int C(int k, int n){  
    if (k == 0 || k == n) return 1;  
    return C(k-1, n-1) + C(k, n-1);  
}
```



# BÀI TOÁN TÍNH HẲNG SỐ TỔ HỢP – ĐỆ QUY CÓ NHỚ

- Khắc phục tình trạng một chương trình con với tham số xác định được gọi đệ quy nhiều lần
- Sử dụng bộ nhớ để lưu trữ kết quả của một chương trình con với tham số xác định
- Bộ nhớ được khởi tạo với giá trị đặc biệt để ghi nhận mỗi chương trình con chưa được gọi lần nào
- Địa chỉ bộ nhớ sẽ được ánh xạ với các giá trị tham số của chương trình con

```
int C(int k, int n){
    if (k == 0 || k == n) return 1;
    return C(k-1, n-1) + C(k, n-1);
}
```



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP – ĐỆ QUY CÓ NHỚ (MÃ GIẢ)

- Khắc phục tình trạng một chương trình con với tham số xác định được gọi đệ quy nhiều lần
- Sử dụng bộ nhớ để lưu trữ kết quả của một chương trình con với tham số xác định
- Bộ nhớ được khởi tạo với giá trị đặc biệt (ví dụ giá trị 0) để ghi nhận mỗi chương trình con chưa được gọi lần nào
- Địa chỉ bộ nhớ sẽ được ánh xạ với các giá trị tham số của chương trình con

```
M[N,N] = {0}; // Initialize 0-array as a memory
              // M[k,n] stores the value C(k,n)

C(k, n){
    if (k == 0 || k == n) M[k,n] = 1;
    else {
        if M[k,n] = 0 then {
            M[k,n] = C(k-1,n-1) + C(k,n-1);
        }
    }
    return M[k,n];
}
```



# BÀI TOÁN TÍNH HẰNG SỐ TỔ HỢP – ĐỆ QUY CÓ NHỚ (CODE HOÀN CHỈNH)

- Khắc phục tình trạng một chương trình con với tham số xác định được gọi đệ quy nhiều lần
- Sử dụng bộ nhớ để lưu trữ kết quả của một chương trình con với tham số xác định
- Bộ nhớ được khởi tạo với giá trị đặc biệt (ví dụ giá trị 0) để ghi nhận mỗi chương trình con chưa được gọi lần nào
- Địa chỉ bộ nhớ sẽ được ánh xạ với các giá trị tham số của chương trình con

```
#include <stdio.h>
#define P 1000000007
#define N 1000
int M[N][N] = {0};
int C(int k, int n){
    if(k == 0 || k == n) M[k][n] = 1;
    else{
        if(M[k][n] == 0){
            M[k][n] = (C(k-1,n-1) + C(k,n-1))%P;
        }
    }
    return M[k][n];
}
int main(){
    int k,n; scanf("%d %d",&k,&n); printf("%d",C(k,n));
    return 0;
}
```





The logo graphic for HUST (Hanoi University of Science and Technology) is a large, stylized circular emblem. It is composed of numerous small red dots arranged in concentric, slightly irregular rings, creating a textured, halftone-like effect. The word "HUST" is centered within this circular pattern in a bold, white, sans-serif font.

**HUST**

 [hust.edu.vn](http://hust.edu.vn)  [fb.com/dhbkhn](https://fb.com/dhbkhn)

**THANK YOU !**