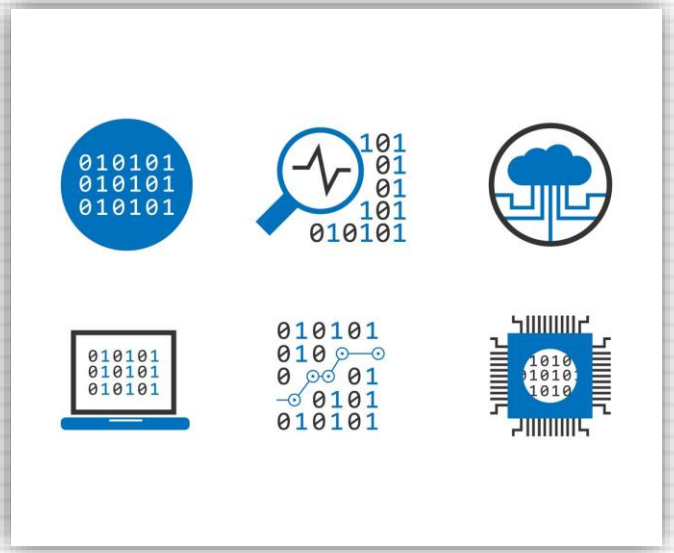
Đại Học Quốc Gia Tp. Hồ Chí Minh
Trường ĐH Khoa Học Tự Nhiên
Khoa Công Nghệ Thông Tin

KỸ THUẬT LẬP TRÌNH



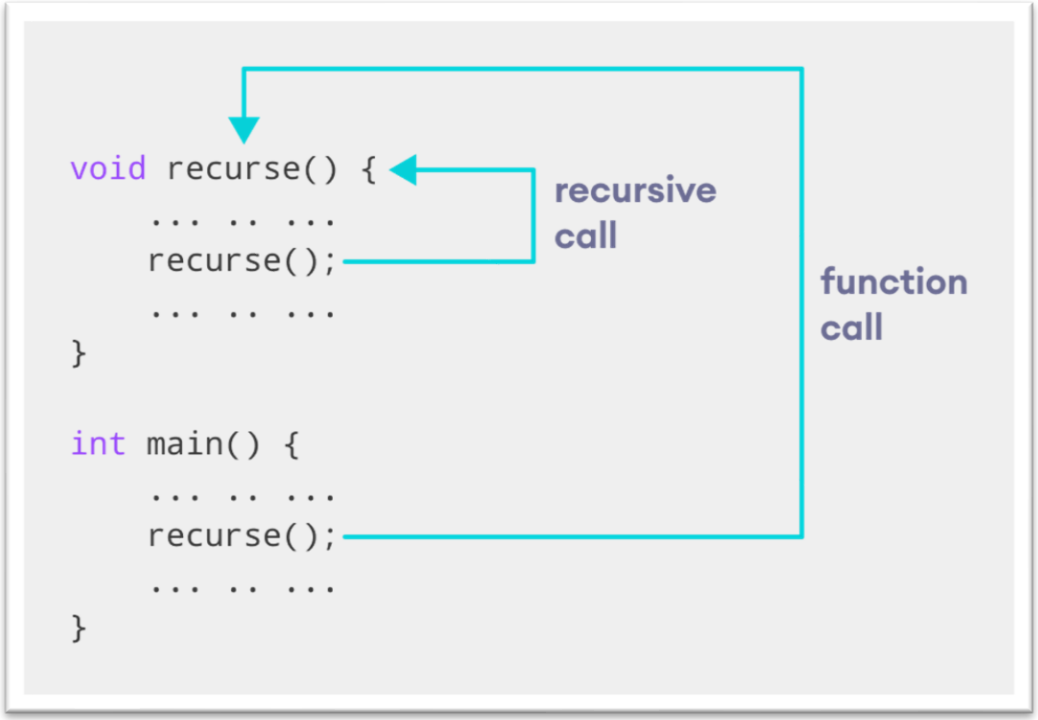
CHƯƠNG 3: ĐỆ QUY (RECURSION)

GV: Phạm Nguyễn Sơn Tùng

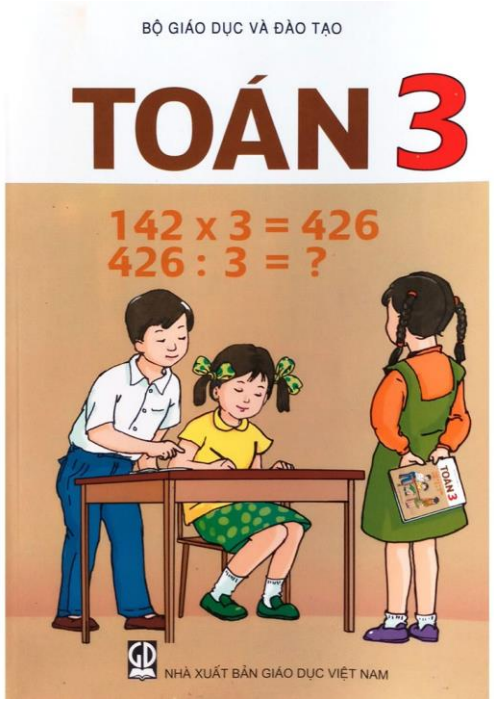
Email: pnstung@fit.hcmus.edu.vn

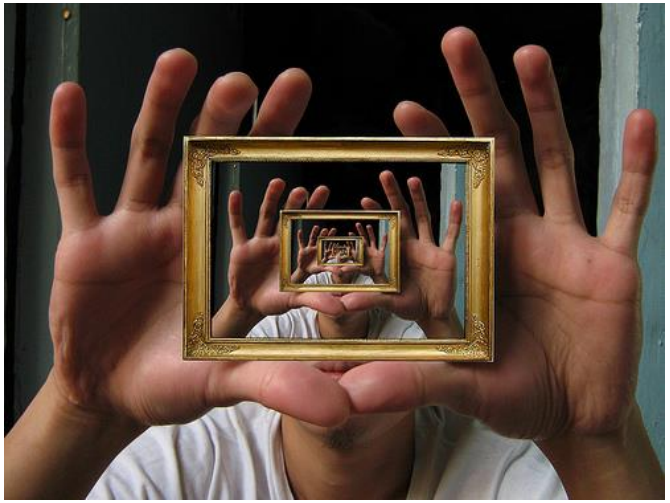
GIỚI THIỆU VỀ ĐỆ QUY

Định nghĩa: khi bạn viết một hàm, và bạn sử dụng lại hàm đó bên trong thân hàm, đó gọi là đệ quy.



GIỚI THIỆU VỀ ĐỆ QUY





GIỚI THIỆU VỀ ĐỆ QUY

Một hàm đệ quy đơn giản.

```
#include <iostream>
using namespace std;
void function(int n)
{
    if (n > 0)
    {
        cout << n << " ";
        function(n - 1);
    }
}
int main()
{
    int n = 5;
    function(n);
    return 0;
}
```

GIỚI THIỆU VỀ ĐỆ QUY

Một hàm đệ quy khác. Theo bạn kết quả sẽ như thế nào?

```
#include <iostream>
using namespace std;
void function(int n)
{
    if (n > 0)
    {
        function(n - 1);
        cout << n << " ";
    }
}
int main()
{
    int n = 5;
    function(n);
    return 0;
}
```

CÁC LOẠI ĐỆ QUY

1. Đệ quy tuyến tính: là loại đệ quy có duy nhất 1 lời mời gọi hàm.

```
int Tong(int n)
{
    if (n == 0)
        return 0;
    return Tong(n - 1) + n;
}

int main()
{
    int n = 5;
    int ketqua = Tong(n);
    cout << ketqua;
    return 0;
}
```

CÁC LOẠI ĐỆ QUY

1. Đệ quy tuyến tính: là loại đệ quy có duy nhất 1 lời mời gọi hàm.

```
int Tong(int n)
{
    if (n == 0)
        return 0;
    return Tong(n - 1) + n;
}

int main()
{
    int n = 5;
    int ketqua = Tong(n);
    cout << ketqua;
    return 0;
}
```

CÁC LOẠI ĐỆ QUY

2. Đệ quy nhị phân: là loại đệ quy có 2 lời gọi làm lại chính nó.

```
long Fibo(int n)
{
    if (n == 0 || n == 1)
        return 1;
    return Fibo(n-1) + Fibo(n-2);
}
int main()
{
    cout << Fibo(5);
    return 0;
}
```

CÁC LOẠI ĐỆ QUY

3. Đệ quy hỗ tương: là loại đệ quy thân hàm này gọi tới làm kia và thân làm kia gọi tới hàm này.

```
void funA(int n)
{
    if (n > 0)
    {
        cout << n << " ";
        funB(n - 1);
    }
}
void funB(int n)
{
    if (n > 1)
    {
        cout << n << " ";
        funA(n / 2);
    }
}
```

CÁC LOẠI ĐỆ QUY

3. Đệ quy hỗ tương: là loại đệ quy thân hàm này gọi tới làm kia và thân làm kia gọi tới hàm này.

```
void funA(int n)
{
    if (n > 0)
    {
        cout << n << " ";
        funB(n - 1);
    }
}

void funB(int n)
{
    if (n > 1)
    {
        cout << n << " ";
        funA(n / 2);
    }
}
```

CÁC LOẠI ĐỆ QUY

4. Đệ quy phi tuyến: là loại đệ quy lời gọi hàm nằm bên trong vòng lặp.

Xn = n*n*Xo + (n-1)*(n-1)*X1 +...+ 1*1*X(n-1). (Xo = 1)

```
#include <iostream>
using namespace std;
int Tong(int n)
{
    if (n == 0)
        return 1;
    long s = 0;
    for (int i = 1; i <= n; i++)
        s += i * i * Tong(n - i);
    return s;
}
int main()
{
    int ketqua = Tong(2);
    cout << ketqua;
}
```

BÀI TẬP TRẮC NGHIỆM

Bài tập 1

```
void Tinh(int n)
{
    if (n == 0)
        return;
    cout << (n % 2);
    Tinh(n / 2);
}
int main()
{
    Tinh(30);
    return 0;
}
```

BÀI TẬP TRẮC NGHIỆM

Bài tập 2

```
int Tinh(int n)
{
    if (n == 3)
        return n;
    else
        return 5 * Tinh(n + 1);
}

int main()
{
    int ketqua = Tinh(1);
    cout << ketqua;
    return 0;
}
```

BÀI TẬP TRẮC NGHIỆM

Bài tập 3

```
int Tinh(int n)
{
    if (n == 0 || n == 1)
        return n;
    if (n % 3 != 0)
        return 1;
    return Tinh(n / 3);
}

int main()
{
    cout<<Tinh(30);
    return 0;
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

5. Đệ quy đuôi: chạy một hàm GCD (**Greatest Common Divisor**).

```
int GCD(int a, int b)
{
    while (a * b != 0)
    {
        if (a > b)
        {
            a %= b;
        }
        else
        {
            b %= a;
        }
    }
    return a + b;
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

5. Đệ quy đuôi: chạy một hàm GCD đệ quy.

```
int GCD(int a, int b)
{
    if (b == 0)
        return a;
    return GCD(b, a % b);
}

void main()
{
    int a = 174;
    int b = 44;
    cout << GCD(a, b);
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

Phân tích hàm số mũ $(x)^n$.

```
int pow(int x, int n)
{
    if (n == 0)
        return 1;
    return pow(x, n - 1)*x;
}

void main()
{
    int m = 2;
    int n = 5;
    cout << pow(m, n);
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

Cũng là hàm số mũ $(x)^n$. Nhưng được viết theo một cách khác.

```
int pow(int x, int n)
{
    if (n == 0)
        return 1;
    if (n % 2 == 0)
        return pow(x * x, n / 2);
    else
        return x * pow(x * x, (n - 1) / 2);
}

void main()
{
    int x = 2;
    int n = 5;
    cout << pow(x, n);
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

6. Đệ quy lồng nhau: ít khi sử dụng vì thời gian chạy rất lâu.

```
int fun(int n)
{
    if (n > 100)
        return n - 10;
    return fun(fun(n + 11));
}

void main()
{
    cout << fun(95);
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

6. Đệ quy lồng nhau: Hàm Ackerman trong toán logic.

$$A(m,n) = \begin{cases} n+1 \text{ if } m = 0 \\ A(m-1,1) \text{ if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m,n-1)) \text{ if } m > 0 \text{ and } n > 0 \end{cases}$$

```
int ack(int m, int n)
{
    if (m == 0)
        return n + 1;
    else if ((m > 0) && (n == 0))
        return ack(m - 1, 1);
    else if ((m > 0) && (n > 0))
        return ack(m - 1, ack(m, n - 1));
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

7. Đệ quy trên mảng một chiều.

```
int TongMang(int* a, int n)
{
    if (n == 1)
        return a[0];
    return TongMang(a, n - 1) + a[n - 1];
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

7. Đệ quy trên mảng một chiều.

```
int TongMang(int* a, int n)
{
    if (n == 1)
        return a[0];
    return TongMang(a, n - 1) + a[n - 1];
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

7. Đệ quy trên mảng một chiều.

```
int TimLonNhat(int* a, int n)
{
    if (n == 1)
        return a[0];
    int linhcanh = TimLonNhat(a, n - 1);
    if (linhcanh > a[n - 1])
        return linhcanh;
    else
        return a[n - 1];
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

7. Đệ quy trên mảng một chiều.

```
int DemDuong(int* a, int n)
{
    if (n == 1 && a[0] >= 0)
        return 1;
    if (n == 1 && a[0] < 0)
        return 0;
    if (a[n - 1] >= 0)
        return (1 + DemDuong(a, n - 1));
    return (DemDuong(a, n - 1));
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

7. Đệ quy trên mảng một chiều.

```
bool KiemTraToanDuong(int* a, int n)
{
    if (n == 0)
        return false;
    if (n == 1)
        if (a[0] > 0)
            return true;
        else
            return false;
    if (a[n - 1] > 0)
        return KiemTraToanDuong(a, n - 1);
    else
        return 0;
}
```

CÁC LOẠI ĐỆ QUY ĐẶC BIỆT

Đệ quy còn được áp dụng vào rất nhiều phương pháp lập trình khác:

- **Đệ quy trên danh sách.**
- Backtracking (quay lui)
- Divide & Conquer (chia để trị)
- Dynamic Programming (Quy hoạch động)

CÁC BÀI TOÁN ĐỆ QUY KHÁC

Bài toán: Tổ hợp là chọn những phần tử từ một nhóm lớn mà không phân biệt thứ tự.

$$C(n,k) = \frac{n!}{(n-k)!k!}$$

Ví dụ 1: xác suất lấy 5 lá bài trong bộ bài 52 lá.

Ví dụ 2:

$$C_k^n = \frac{n!}{k! * (n-k)!} \qquad C_6^{45} = \frac{45!}{6! * (45! - 6!)} = 8145060$$

https://tienichnho.com/to-hop-chap

CÁC BÀI TOÁN ĐỆ QUY KHÁC

Cách 1: Giải bài tổ hợp chập k của n phần tử. Sử dụng 2 hàm khác nhau.

```
int GiaiThua(int n)
{
    if (n == 1)
        return 1;
    return n * GiaiThua(n - 1);
}
int C(int n, int k)
{
    int a = GiaiThua(n);
    int b = GiaiThua(n - k);
    int c = GiaiThua(k);
    return a/(b*c);
}
```

CÁC BÀI TOÁN ĐỆ QUY KHÁC

Cách 2: Giải với công thức quy nạp.

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

BÀI TẬP TRẮC NGHIỆM

Bài tập 1

```
int Tinh(int x, int y)
{
    if (x == 0)
        return y;
    return Tinh(x - 1, x + y);
}
```

BÀI TẬP TRẮC NGHIỆM

Bài tập 2

```
int Tinh(int n)
{
    if (n <= 1)
        return 1;
    if (n % 2 == 0)
        return Tinh(n / 2);
    return Tinh(n / 2) + Tinh(n / 2 + 1);
}
```