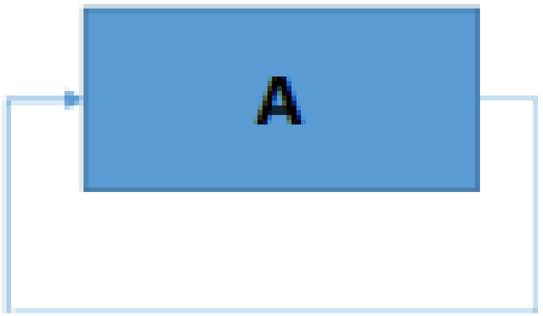
- Gọi chương trình con đệ quy



Tính thời gian thực hiện của A?

- Độ phức tạp chương trình con dạng đệ quy

  - Cách giải quyết:
    1. Thành lập phương trình đệ quy
    2. Giải phương trình đệ quy

    Nghiệm của lời giải ở bước 2 là thời gian thực hiện chương trình

- Độ phức tạp chương trình con dạng đệ quy
    - Phương trình đệ quy: Biểu diễn mối liên hệ giữa T(n) với T(k), k<n. Trong đó T(n) thời gian thực hiện chương trình và T(k) thời gian thực hiện với kích thước bộ dữ liệu là k, và k<n.
    - Để lập phương trình: Căn cứ vào chương trình đệ quy.

- Độ phức tạp chương trình con dạng đệ quy
  - Dạng tổng quát:

$$T(n) = \begin{cases} C(n_0), & \text{với } n=n_0 \\ T(k) \quad + \quad d^* & \text{với } n>k>n_0 \end{cases}$$

    - $C(n_0)$: Thời gian thực hiện khi $n=n_0$
    - $T(k)$: thời gian thực hiện khi $n>k>n_0$
    - d^*: Thời gian phân chia và tổng hợp kết quả

- Độ phức tạp chương trình con dạng đệ quy
  - Ví dụ: xét hàm tính giai thừa

```
Function gt(n)
begin
    if n=0 then gt=1
    else gt=n*gt(n-1)
end
```

```python
def giaithua(n):
    gt = 0;
    if (n == 0): gt = 1;
    else: gt = n*giaithua(n-1);
```

Gọi T(n) là thời gian tính n!, thì T(n-1) là thời gian tính (n-1)!

Khi n=0, ta có C(0)=1 (phép gán)

Khi n>0, hàm gọi đệ quy gt(n-1), tốn T(n-1)

Tổng hợp kết quả ở đây cần 1 phép gán, d*=1

$$T(n) = \begin{cases} 1, & \text{với } n=0 \\ T(n-1) + 1 & \text{với } n>k>0 \end{cases}$$

- Giải phương trình đệ quy – Phương pháp truy hồi
  1. Với $n>k>n_0$: dùng phương trình đệ quy lần lượt thay thế $T(k)$ vào vế phải
  2. Dừng khi $k=n_0$
  3. Thế $T(n_0)$ để tìm $T(n)$

1. Ví dụ: Giải

$$T(n) = \begin{cases} 1, & \text{với } n=0 \\ T(n-1) + 1 & \text{với } n>k>0 \end{cases}$$

$T(n) = T(n-1) + 1$

$= T(n-2) + 1 + 1$

....

$= T(n-i) + i$

Dừng khi $n-i = 0$, hay $i=n$, khi đó $T(n) = 1 + n = O(n)$

Ví dụ 2:

$$T(n) = \begin{cases} 0 & \text{với } n=1 \\ T(n/2) \quad + \quad 1 & \text{với } n>1 \end{cases}$$

Ví dụ 2:

$$T(n) = \begin{cases} 0 & \text{với } n=1 \\ T(n/2) \quad + \quad 1 & \text{với } n>1 \end{cases}$$

Giải:

$$T(n) = T(n/2) + 1$$
$$= T(n/2^2) + 1 + 1$$
....
$$= T(n/2^i) + i$$

Dừng: $n/2^i = 1$ (n_0), hay $i=\log_2 n$, khi đó $T(n) = 0 + \log_2 n$

Bài tập vận dụng:
Hãy tính độ phức tạp của hàm sau:

```
def test(n):
    if (n>0):
        for i in range(n):
            print(i)
        test(n-1)
```

Giải:

```python
def test(n):
    if (n>0):
        for i in range(n):
            print(i)
        test(n-1)
```

1 -----------→

n+1 -----------→

n -----------→

T(n-1) -----------→

Vậy: $T(n) = T(n-1) + 2n + 2$

Hay: $T(n) = T(n-1) + n$

$$T(n) = \begin{cases} 1 & \text{khi } n=0 \\ T(n-1) + n & \text{khi } n>k>0 \end{cases}$$

$T(n) = T(n-1) + n$

$= T(n-2) + (n -1) + n$

$= T(n-3) + (n-2) + (n -1) + n$

.

.

$= T(n-k) + n(n-1)/2$

Hàm dừng khi $k = n$

Khi đó $T(n) = T(0) + n(n-1)/2$

Vậy => $O(n^2)$