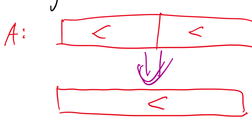
Divide and Conquer (分治法)

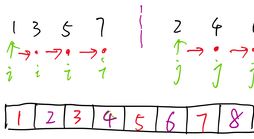
1. Divide the problem into subproblems
2. Conquer the subproblems by solving them recursively
3. Combines subproblem solutions.

Ex. Merge Sort



$T(n) = 2T(\frac{n}{2}) + cn$

Ex.



$T(n) = 2T(\frac{n}{2}) + cn$

$a=2,\ b=2,\ n^{\log_b a} = n^{\log_2 2} = n$

$f(n) = cn = \Theta(n(\lg^0 n)) = \Theta(n \lg n)$

Binary Search

```
binary_search(A, p, q, k)                    T(n)
if p > q                                      c
    return -1
m = ⌊(p+q)/2⌋
if A[m] == k
    return m
else if A[m] > k
    return binary_search(A, p, m-1, k)  → T(n/2 - 1)
else
    return binary_search(A, m+1, q, k)  → T(n/2 - 1)
```

binary-Search(A, 1, n, k)

$E[T(n)] = c + \frac{1}{2}T(\frac{n}{2}-1) + \frac{1}{2}T(\frac{n}{2}-1)$

$= T(\frac{n}{2}-1) + c$

$\leq T(\frac{n}{2}) + c$

$T(n) = T(\frac{n}{2}) + c$

$a=1,\ b=2,\ n^{\log_b a} = n^{\log_2 1} = n^0 = 1$

$f(n) = c = \Theta(n^0 \lg^0 n)$

$T(n) = \Theta(n^0 \lg^{0+1} n) = \Theta(\lg n)$

a^n

```
for (i=1; i<n; ++i)
```

$T(n) = \Theta(n)$

$$a^n = \begin{cases} a^{\frac{n-1}{2}} \cdot a^{\frac{n-1}{2}} \cdot a, & n \text{ is odd} \\ a^{\frac{n}{2}} \cdot a^{\frac{n}{2}}, & n \text{ is even} \end{cases}$$

```
pow(a, n)                                    T(n)
if n == 1                                     c1
    return a
if n%2 == 1
    A = pow(a, (n-1)/2)                       T(n/2)
    return A·A·a                              c2
else
    A = pow(a, n/2)                           T(n/2)
    return A·A                                c3
```

$T(n) = T(\frac{n}{2}) + c$

$a=1,\ b=2$

$T(n) = \Theta(\lg n)$