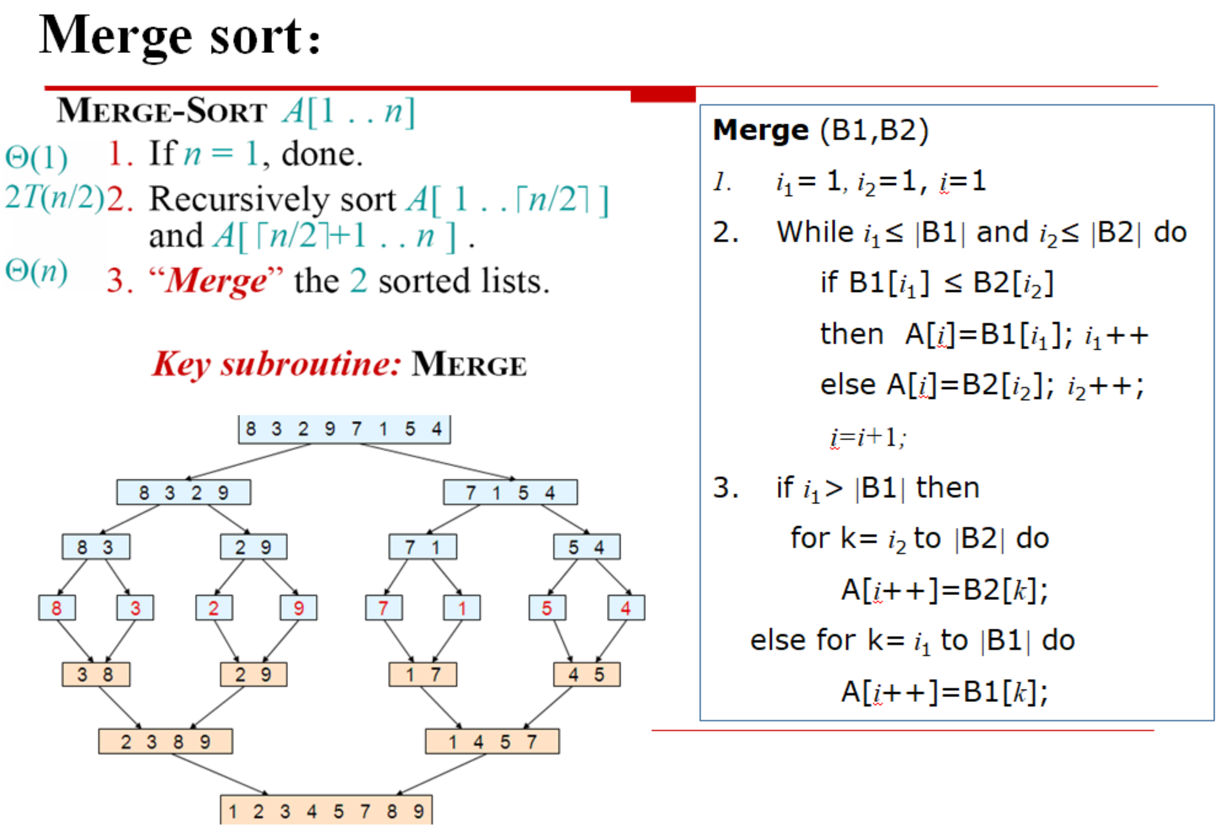
# Algorithm 0x05

49.234.77.58/index.php/2019/12/25/algorithm-0x05

XZLang 2019年12月25日

## 分治法

分治法的基本思想是把一个规模为n的问题分解成k个规模较小的子问题，这些子问题相互独立且与原问题相同，只是规模更小。递归地求解这些子问题，最后将各个子问题的解合并得到原问题的解。



# Integer Multiplication

multiply two $n$-digit numbers $x$ and $y$

Recursive-Multiply(x,y):
  Write $x = x_1 \cdot 2^{n/2} + x_0$
     $y = y_1 \cdot 2^{n/2} + y_0$
  Compute $x_1 + x_0$ and $y_1 + y_0$
  $p$ = Recursive-Multiply($x_1 + x_0$, $y_1 + y_0$)
  $x_1 y_1$ = Recursive-Multiply($x_1, y_1$)
  $x_0 y_0$ = Recursive-Multiply($x_0, y_0$)
  Return $x_1 y_1 \cdot 2^n + (p - x_1 y_1 - x_0 y_0) \cdot 2^{n/2} + x_0 y_0$

$T(n) \leq 3T(n/2) + cn$ $O(n^{\log_2 3}) = O(n^{1.59})$

2019/12/25

# Strassen's Matrix Multiplication

$$\begin{pmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{pmatrix} = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix} * \begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix}$$

$$= \begin{pmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 + M_3 - M_2 + M_6 \end{pmatrix}$$

- $M_1 = (A_{00} + A_{11}) * (B_{00} + \mathbf{B_{11}})$
- $M_2 = (A_{10} + A_{11}) * B_{00}$
- $M_3 = A_{00} * (B_{01} - \mathbf{B_{11}})$
- $M_4 = A_{11} * (B_{10} - \mathbf{B_{00}})$
- $M_5 = (A_{00} + A_{01}) * \mathbf{B_{11}}$
- $M_6 = (A_{10} - A_{00}) * (B_{00} + \mathbf{B_{01}})$
- $M_7 = (A_{01} - A_{11}) * (B_{10} + \mathbf{B_{11}})$

# Strassen's Matrix Multiplication

## Strassen's algorithm

1. ***Divide:*** Partition $A$ and $B$ into $(n/2)\times(n/2)$ submatrices. Form terms to be multiplied using $+$ and $-$ .
2. ***Conquer:*** Perform $7$ multiplications of $(n/2)\times(n/2)$ submatrices recursively.
3. ***Combine:*** Form $C$ using $+$ and $-$ on $(n/2)\times(n/2)$ submatrices.

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

$T(n)=O(n^{\log 7}) = O(n^{2.81})$

Best to date: $O(n^{2.376})$

## Find the k-th smallest element

主要利用快速排序的思想查找第K小的数，核心的思想就是快排的分治思想，具体思路：

1. 利用快排的Parition()函数将数组分成两部分，返回基准值value，小于value的都在左边，大于的在右边.
2. 如果index刚好等于k,则说明index位置的数就是我们要找的数，如果值小于它，就肯定在左边，大于就在右边.
3. 递归在index的左边或者右边进行查找

```
//找第k小的数
#include <iostream>
using namespace std;

int partition(int a[], int left, int right)
{//将数组a的第left到right个元素进行划分
 int x = a[left];

 while (left < right)
 {//采用快排策略
  while (left < right && a[right] >= x)
   right--;
  a[left] = a[right];

  while (left < right && a[left] <= x)
   left++;
  a[right] = a[left];
 }

 a[left] = x;

 return left;
}

int find(int a[], int left, int right, int k)
{//在数组a的第left到right中寻找第k小的数
 int pos = partition(a, left, right);

 if (k - 1 == pos)
  cout << a[k - 1];
 else if (k - 1 < pos)//判断下一次划分在哪一区间进行
  find(a, left, pos - 1, k);
 else
  find(a, pos + 1, right, k);

 return 0;

}

int main()
{
 int n, k;
 cin >> n >> k;

 int a[1000];
 for (int i = 0; i < n; i++)
  cin >> a[i];

 find(a, 0, n - 1, k);

 return 0;

}
```