第8章　算法设计初体验

第84课　分治法

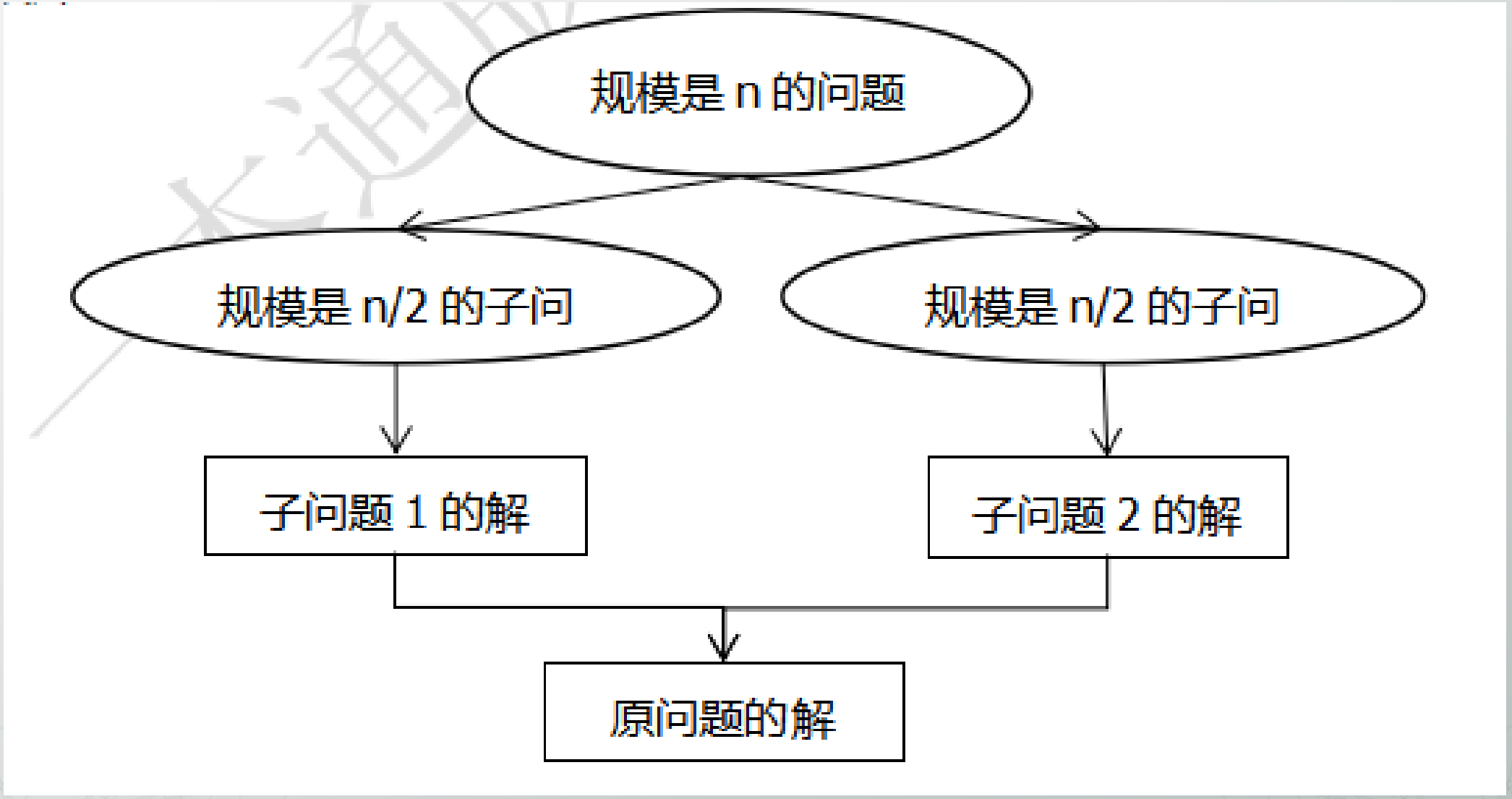
《信息学奥赛一本通·编程启蒙 C++版》

分治法是按照以下方案工作的。

(1)将一个问题划分为同一类型的若干子问题,子问题最好规模相同。

(2)对这些子问题求解(一般使用递归方法，但在问题规模足够小时，有时会利用另一个算法)。

(3)有必要的话，合并这些子问题的解，以得到原始问题的答案。


规模是 n 的问题
规模是 n/2 的子问
规模是 n/2 的子问
子问题 1 的解
子问题 2 的解
原问题的解

【例 84.1】 取余运算

【题目描述】

输入 b, P，k 的值，求 $b^P \bmod k$ 的值。其中 b, p, k×k 为长整型数。

【输入格式】

输入 b, P, k 的值。

【输出格式】

求 $b^P \bmod k$ 的值。

【样例输入】

```
2 10 9
```

【样例输出】

```
2^10 mod 9=7
```

【代码实现】

```
#include<iostream>
using namespace std;
long long a,b,c;
long long pow(long long b,long long p,long long k) {
    if(p==0) return 1;
    else if(p%2==1) return pow(b,p-1,k)*b%k;
    long long temp = pow(b,p/2,k);
    return temp*temp%k;
}
int main(){
   cin>>a>>b>>c;
   cout<<a<<'^'<<b<<" mod "<<c<<"="<<pow(a,b,c);
   return 0;
}
```

【例 84.2】分香蕉

【题目描述】

又要了丰收的季节，花果山的 n 个香蕉成熟了，每个香蕉的质量为 a_i。蒜头君还养着 m 只猴子，每只猴子的体重为 b_i。猴子们吃香蕉有一定的顺序，按照体重从大到小的顺序一个个拿香蕉。当一轮拿完时，如果还有多的香蕉就会继续一个个拿，直到香蕉被取完。每个猴子都很聪明，每次会选质量最大的那个香蕉。

现在问题来了，最后每个猴子能获得多少质量的香蕉？

【输入格式】

第一行两个整数 n，m（$1 \leqslant n, m \leqslant 10^5$）。

第二行 n 个整数 a_i（$1 < a_i \leqslant 10^4$），表示每个香蕉的质量。

第三行 m 个整数 b_i（$1 < b_i \leqslant 10^9$），表示每只猴子的体重，保证每个体重互不相同。

【输出格式】

一行，m 个用空格分隔的整数，表示每个猴子获得的香蕉质量之和。

【样例输入】

```
5 3
1 2 3 4 5
3 2 1
```

【样例输出】

```
7 5 3
```

```cpp
#include<bits/stdc++.h>
using namespace std;
struct mo{
    long long t,sum,num;
};
mo a[100005];
int b[100005],n,m,q;
bool cmp1(mo x,mo y){
    return x.t>y.t;
}
bool cmp2(mo x,mo y){
    return x.num<y.num;
}
//快速排序
void quick_sort(int a[], int l, int r){
    if (l < r){
        int i,j,x;
        i = l;
        j = r;
        x = a[i];
        while (i < j){
            while(i < j && a[j] > x)
                j--; // 从右向左找第一个小于x的数
            if(i < j)
                a[i++] = a[j];
            while(i < j && a[i] < x)
                i++; // 从左向右找第一个大于x的数
            if(i < j)
                a[j--] = a[i];
        }
        a[i] = x;
        quick_sort(a, l, i-1); // 分治
        quick_sort(a, i+1, r); // 分治
    }
}
//将一个数组中的两个相邻有序区间合并成一个
void merge(mo a[], int start, int mid, int end,bool (*cmp)(mo,mo)){
    // tmp是汇总2个有序区的临时区域，也可以用mo tmp[end-start+1]
    mo *tmp = (mo *)malloc((end-start+1)*sizeof(mo));
    // i:第1个有序区的索引 j:第2个有序区的索引 k:临时区域的索引
    int i = start,j = mid + 1,k = 0;
    while(i <= mid && j <= end)
        //if (a[i].t >= a[j].t) //根据体重从大到小排列
        if((*cmp)(a[i],a[j]))
            tmp[k++] = a[i++];
        else
            tmp[k++] = a[j++];
```

```
48.     while(i <= mid) tmp[k++] = a[i++];
49.     while(j <= end) tmp[k++] = a[j++];
50.     //将排序后的元素，全部都整合到数组a 中。
51.     for (int i=0;i<k;i++) a[start+i]=tmp[i];
52.     free(tmp);
53.}
54.//归并排序(从上往下)
55.void merge_sort_up_to_down(mo a[], int start, int end,
   bool (*cmp)(mo,mo))
56.{
57.     if(a==NULL || start>=end) return ;
58.     int mid=(end+start)/2;
59.     merge_sort_up_to_down(a, start, mid,cmp); // 分治
60.     merge_sort_up_to_down(a, mid+1, end,cmp); // 分治
61.     // a[start...mid] 和 a[mid...end]是两个有序空间,
62.     // 将它们排序成一个有序空间a[start...end]
63.     merge(a, start, mid, end,cmp);
64.}
65.//归并排序(从上往下)
66.void merge_sort_down_to_up(mo a[], int start, int end,
   bool (*cmp)(mo,mo))
67.{
68.  int length=(end-start)+1;
69.     for (int sz = 1;sz<length;sz *= 2)
70.      for (int lo = 0; lo <length-sz; lo += 2 * sz)
71.       merge(a, lo, lo+sz-1, min(length-1,lo+2*sz-
   1),cmp);
72.}
```

```
73.int main()
74.{
75.    cin>>n>>m;
76.    for(int i=0;i<n;i++) cin>>b[i];
77.    for(int i=0;i<m;i++){
78.       cin>>a[i].t;
79.       a[i].num=i;
80.    }
81.    quick_sort(b,0,n-1);
82.    merge_sort_up_to_down(a,0,m-1,cmp1);
83.    while(n--){
84.       a[q].sum+=b[n];
85.       q++;
86.       q%=m;
87.    }
88.    merge_sort_down_to_up(a,0,m-1,cmp2);
89.    for(int i=0;i<m;i++) cout<<a[i].sum<<" ";
90.    return 0;
91.}
```

练 84.1 2011^n 的后四位

【题目描述】

已知长度最大为 200 位的正整数 n，请求出 2011^n 的后四位。

【输入格式】

第一行为一个正整数 k，代表有 k 组数据（k≤200），接下来的 k 行，每行都有一个正整数 n，n 的位数≤200。

【输出格式】

每一个 n 的结果为一个整数占一行，若不足 4 位，去除高位多余的 0

【样例输入】

3
5
28
792

【样例输出】

1051
81
5521

【代码实现】

```cpp
#include<bits/stdc++.h>
using namespace std;
long long fast_pow_mod(long long b,long long p,long long k){
    if(p==0) return 1;
    else if(p%2==1) return fast_pow_mod(b,p-1,k)*b%k;
    long long temp = fast_pow_mod(b,p/2,k);
    return temp*temp%k;
}
void solve() {
    char s[205];
    cin >> s;
    int len = strlen(s);
    char *p = s + (len > 3 ? len - 3 : 0);
    int n;
    sscanf(p, "%d", &n);
    n %= 500;
    cout << fast_pow_mod(2011, n, 10000) << endl;
}
int main(){
   int n;
   cin >> n;
   for (int i = 0; i < n; i++)
      solve();
   return 0;
}
```

练 84.2 光荣的梦想

【题目描述】

Prince 对他在这片大陆上维护的秩序感到满意，于是决定启程离开艾泽拉斯。在他动身之前，Prince 决定赋予 King_Bette 最强大的能量以守护世界、保卫这里的平衡与和谐。在那个时代，平衡是个梦想。因为有很多奇异的物种拥有各种不稳定的能量，平衡瞬间即被打破。KB 决定求助于你，帮助他完成这个梦想。

一串数列即表示一个世界的状态。

平衡是指这串数列以升序排列。而从一串无序数列到有序数列需要通过交换数列中的元素来实现。KB 的能量只能交换相邻两个数字。他想知道他最少需要交换几次就能使数列有序。

【输入格式】

第一行为数列中数的个数 n,第二行为 n <= 10000 个数。表示当前数列的状态。

【输出格式】

输出一个整数，表示最少需要交换几次能达到平衡状态。

【样例输入】

4
2 1 4 3

【样例输出】

2

【代码实现】

```
#include<bits/stdc++.h>
using namespace std;
int n,a[10005],b[10005],ans;
int main(){
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
        scanf("%d",&a[i]);
    for(int i=1;i<=n;i++)
        for(int j=i+1;j<=n;j++)
            if(a[i]>a[j]) ans++;
    printf("%d",ans);
    return 0;
}
```

谢谢！