算法设计与分析

作业（四）

姓　　名	熊恪峥
学　　号	22920202204622
日　　期	2022年3月14日
学　　院	信息学院
课程名称	算法设计与分析

作业（四）

目录

1 题5.2 1

2 题5.3 1

3 题5.8 2

4 题5.10 2

5 题5.18 3

6 XOJ 1004 5

7 XOJ1057 5

1 题5.2

实现非递归的归并排序进行一个自下而上的从2个元素的数组开始的合并。算法如算法1，其中$merge$就是普通的二路归并实现。

算法 1 非递归归并排序

procedure MergeSort(A,n)
  for $size \leftarrow 1$ to n step $size$ **do**
    for $left \leftarrow 0$ to $n-1$ step $2size$ **do**
      $mid \leftarrow min(left + size - 1, n - 1)$
      $right \leftarrow min(left + 2size - 1, n - 1)$
      $merge(A, left, mid, right)$

通过画出区间划分的树状图，可以发现它与递归方法等效，并且有相同的时间复杂度，它的时间复杂度仍是$O(n \log n)$，这是基于比较的排序算法的时间复杂度下界，因此它是最有效的。

2 题5.3

如算法2，通过二分搜索的方法寻找x，在找到时返回下标，否则返回-1。

算法 2 二分搜索

procedure Search(A,x,l,r)
  $mid \leftarrow \frac{l+r}{2}$
  if $A[mid] == x$ **then return** mid
  if $l >= r$ **then return** -1
  else if $x > A[mid]$ **then return** $Search(A, x, mid + 1, r)$
  else if $x < A[mid]$ **then return** $Search(A, x, l, mid - 1)$
   return -1

复杂度分析：该算法有递推方程

$$T(n) = T(\lfloor \frac{n}{2} \rfloor) + 1$$

可以证明

$$T(n) = O(\log n)$$

证：$T(n) < c \log n$

当$n = 2$时$T(n) = 1 \leq c \log 2 = c$

当

$$c \geq 1$$

若 $T(\frac{n}{2}) \leq c \log \frac{n}{2}$ 成立，则

$$\begin{aligned} T(n) &= T(\frac{n}{2}) + 1 \\ &\leq c \log \frac{n}{2} + 1 \\ &= c \log n - c \log 2 + 1 \leq c \log n \end{aligned}$$

当$c \geq \frac{1}{\log 2}$成立

3 题5.8

4路归并排序如算法3，把当前区间分成四份并对每一份分别进行归并排序，之后对左边的两个区间、右边的两个区间分别进行二路归并，此时得到了左、右两个有序子区间。再对这两个子区间进行再一次归并，得到整个有序序列。

算法 3 二分搜索

```
procedure MergeSort(A,l,r)
    mid ← (l+r)/2
    lmid ← (l+mid)/2
    rmid ← (mid+r)/2
    MergeSort(A,l,lmid)
    MergeSort(A,lmid,mid)
    MergeSort(A,mid,rmid)
    MergeSort(A,rmid,r)
    Merge(A,l,lmid,mid)
    Merge(A,mid,rmid,r)
    Merge(A,l,mid,r)
```

复杂度分析：该算法满足递推方程

$$T(n) = 4T(\frac{n}{4}) + 2n$$

因为每一个子问题的规模是$\frac{n}{4}$，合并子区间时先合并左右两边长为$\frac{n}{4} + \frac{n}{4} = \frac{n}{2}$的子区间，最后进行长度为$n$的子区间，由$Merge$操作的时间复杂度w为$O(n)$可知合并的代价是$2n$。则有

$$T(n) = O(n \log n)$$

证： $T(n) < cn \log n$

当$n = 4$有$T(n) = 8 < c \cdot 4 \log 4$，当$c \geq 1$

若$T(\frac{n}{4}) \leq c\frac{n}{4} \log \frac{n}{4}$，则

$$\begin{aligned} T(n) &= 4T(\frac{n}{4}) + 2n \\ &\leq cn \log \frac{n}{4} + 2n \\ &= cn \log n + (2 - 2c \log 2)n \\ &\leq cn \log n \end{aligned}$$

当$c \geq \frac{1}{\log 2}$成立

4 题5.10

快速排序的$partition$操作能够做到使得小于枢轴($pivot$)的元素都处于枢轴左侧，大于枢轴的元素都处在枢轴右侧，根据这一性质，那么当一次$partition$完成时

- 如果枢轴处在k处，那么它恰好是第k大
- 如果枢轴在k右侧，k比第k大更大，那么第k大的元素在枢轴左侧的子区间中，那么可以递归地继续寻找第k大
- 否则，第k大的元素在枢轴右侧的子区间中，递归地寻找第k大

根据以上思路，实现算法4，其中$partition$就是快速排序算法中$partition$的实现，它的时间复杂度是$O(n)$

算法 4 选择第k大元素

```
procedure SELECT(A,k,l,r)
    pivot ← PARTITION(A,l,r)
    if pivot == k then
        return A[pivot]
    else if pivot > k then
        return SELECT(A,k,l,pivot-1)
    else
        return SELECT(A,k,pivot+1,r)
    return −1
```

平均情况下，可以认为元素的大小随机分布，那么对于每一个元素大致有$\frac{n}{2}$个元素比它大或者比它小。考虑到$partition$的开销，那么有如下递推式

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

证：$T(n) \leq cn$

当$n = 1$，有$T(1) = 1 \leq cn$，当$c \geq 1$

若$T(\frac{n}{2}) < c\frac{n}{2}$成立，则

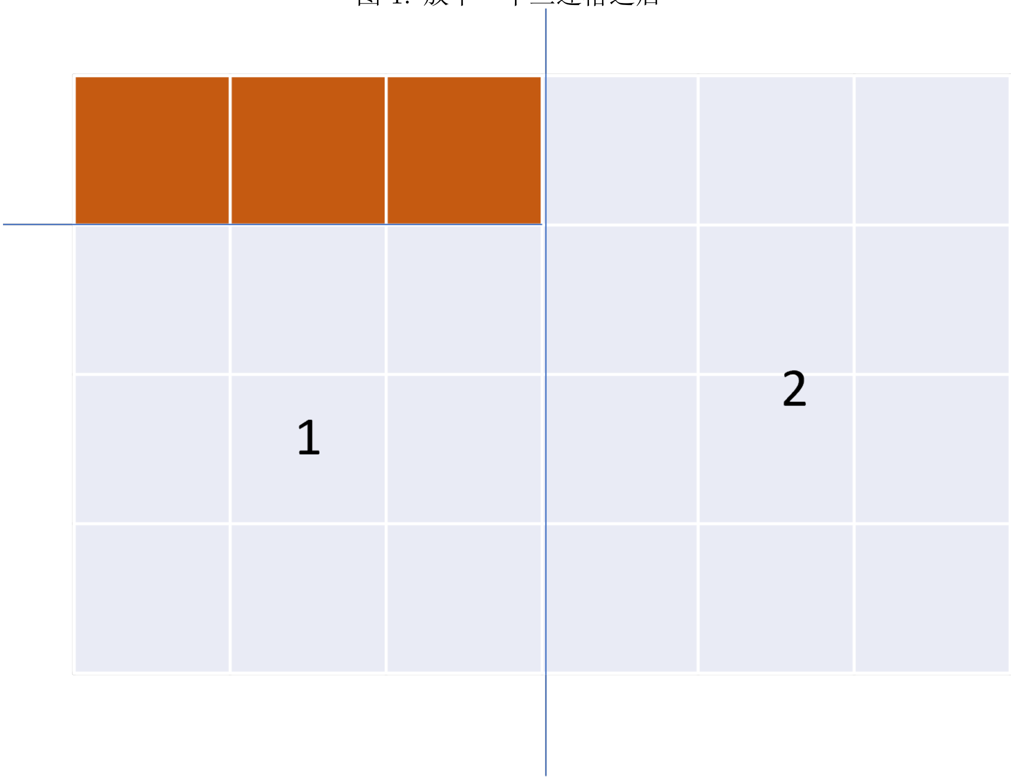
$$\begin{aligned} T(n) &= T(\frac{n}{2}) + n \\ &\leq c\frac{n}{2} + n = (1 + \frac{n}{2})n \\ &\leq cn \end{aligned}$$

当$n \geq 2$时成立

5 题5.18

如图5所示，在左上角放下一个三连格之后，可以将完整的棋盘分为两个完整的子棋盘，然后可以递归地继续解决这两个完整的子棋盘的铺设问题。

图 1: 放下一个三连格之后



6 XOJ 1004

本题要求最大公约数和最小公倍数，由公式

$$gcd(x,y) = gcd(b, a \mod b) \quad a > b \text{且} a \mod b \neq 0$$

和gcd与lcm的关系

$$lcm(x,y) = x \cdot y \cdot gcd(x,y)$$

可以简单地给出递归实现代码1，运行结果如图2

代码 1: XOJ1004

```python
def gcd(a, b):
        return b if a % b == 0 else gcd(b, a % b)


a, b =  map( int,  input().split())
g = gcd(a, b)

print(g, a * b // g, sep='\n')
```

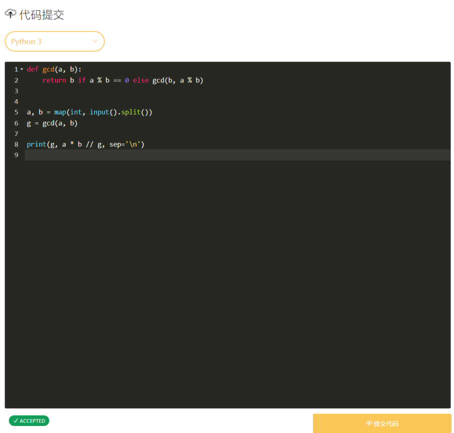


图 2: XOJ1004

图 3: XOJ1057

7 XOJ1057

按题意在区间$[1,n]$内计算有几个数字被n整除即可，运行结果如图3

代码 2: XOJ1057

```python
n =  int( input())

print( sum(1 for i in  filter(lambda x: n % x == 0,  range(1, n + 1))))
```