一、（16 分）设 $f(n)$和 $g(n)$为自然数集合上的函数，c 和 k 为某个大于 0 的常数，在下面的空格内填上“Y”(表示是) 或者“N”(表示否).

$f(n)$	$g(n)$	$f(n)=O(g(n))$	$f(n)=o(g(n))$	$f(n)=\Omega(g(n))$	$f(n)=\Theta(g(n))$
$\log^k n$	n	Y	Y	N	N
$\sqrt{n}$	$n^{\sin n}$	N	N	N	N
$n^{\log c}$	$c^{\log n}$	Y	N	Y	Y
$\log(n!)$	$\log(n^n)$	Y	N	Y	Y

二、（10 分）下述 Find-Second-Min 算法是找第二小算法. 输入是 n 个不等的数构成的数组 S，输出是第二小的数 *SecondMin*. 在最坏情况下，该算法做多少次比较？

```
Find-Second-Min(S,n)
1.  if   S[1] < S[2]
2.  then   min ← S[1], SecondMin ← S[2]
3.  else   min ← S[2], SecondMin ← S[1]
4.  for    i ← 3   to   n   do
5.             if     S[i] < SecondMin
6.             then   if   S[i] < min
7.                              then     SecondMin ← min,   min ← S[i]
8.                              else     SecondMin ← S[i]
```

解：行 4 的 for 循环执行 n-2 次，每次至多 2 次比较，比较次数至多为

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

三、（14 分）设原问题的规模是 n，从下述三个算法中选择一个最坏情况下时间复杂度最低的算法，简要说明你的理由.

算法 A：将原问题划分规模减半的 5 个子问题，递归求解每个子问题，然后在线性时间将子问题的解合并得到原问题的解.

算法 B：先递归求解 2 个规模为 n-1 的子问题，然后在常量时间内将子问题的解合并.

算法 C：将原问题划分规模为 n/3 的 9 个子问题，递归求解每个子问题，然后在 $O(n^3)$时间将子问题的解合并得到原问题的解.

解：选择算法 A。

算法 A：$T_A(n)=5T_A(n/2)+O(n)$， $T_A(n)=\Theta(n^{\log 5})$

算法 B：$T_B(n)=2T_B(n-1)+O(1)$， $T_B(n)=\Theta(2^n)$

算法 C：$T_C(n)=9T_C(n/3)+O(n^3)$， $T_C(n)=\Omega(n^3)$

四、（20 分）设 $A=\{a_1,a_2,\ldots,a_n\}$, $B=\{b_1,b_2,\ldots,b_m\}$是整数集合， 其中 $m=O(\log n)$。设计算法计算集合 $C=(A-B)\cup(B-A)$，说明算法的主要步骤，并以比较作基本运算分析算法最坏情况下的时间复杂度.

解：
算法
1. 对 B 排序；
2. 对每个 A 元素在 B 中二分检索，如果存在，则删除 B 中相等的元素；
3. 将 A 中元素放到输出集合 C 中；
4. 将 B 中剩余元素加到 C 集合中.

时间：$O(n\log m)=O(n\log\log n)$

五、（20 分）设 $S=\{1,2,\ldots,n\}$是 n 项广告的集合，广告 i（$i=1,2,\ldots,n$）有发布开始时间 $s(i)$和截至时间 $d(i)$，发布效益是 $v(i)$，其中 $s(i)$是非负整数, $d(i)$ 和 $v(i)$是正整数。问如何在 S 中选择一组广告 A，使得 A 中任两个广告都相容（时间段不重叠）且总效益最大？

解：用动态规划算法.

算法一、对广告按照截止时间排序，使得 $d(1)\le d(2)\le\ldots\le d(n)$.

设 $F[k]$表示考虑前 k 个广告时的最大效益. 定义 $p(k)$是与 k 相容且标号小于 k 的活动中的最大标号. 那么 $F[k]$满足如下关系：

$$F[k]=\max\{F[k-1],F[p(k)]+v(k)\}$$
$$F[1]=v(1)$$

设立标记函数 $i[k]=k$ 若 $F[p(k)]+v(k)\ge F[k-1]$

$i[k]=i[k-1]$ 否则

预处理计算所有的 $p(k)$，$k=1,2,\ldots,n$. 对于给定的 k，用二分检索找到 $p(k)$，若不存在这样的活动，则令 $p(k)=0$ 且 $F[0]=0$. 对于给定的 k，二分捡索用 $O(\log k)$时间，预处理总计用 $O(n\log n)$时间. 对所有 $F[k]$的计算需要 $O(n)$的时间，因此算法的时间复杂度是 $T(n)=O(n\log n)$.

算法二、动态规划算法，$F_k(y)$是考虑前 k 个广告，截止时间为 y 的最大效益

$F_k(y)=\max\{F_{k-1}(y), F_{k-1}(s(k))+v(k)\}$

$$F_1(y)=\begin{cases} v(1) & 若\ y \ge d(1) \\ 0 & 否则 \end{cases}$$

标记函数

$$i_k(y)=\begin{cases} k & 若\ F_{k-1}(s(k))+v(k) \ge F_{k-1}(y) \\ i_{k-1}(y) & 否则 \end{cases}$$

$i_1(y)=1 \quad \text{if} \quad y \ge d(1)$

时间为 $O(n^2)$.
算法一是更好的算法。

六、（20 分）有 n 个文件存在磁带上，每个文件占用连续的空间。已知第 i 个文件需要的存储空间为 s_i，被检索的概率是 f_i，$i=1,2,\ldots,n$，且 $f_1+f_2+..+f_n=1$. 检索每个文件需要从磁带的开始位置进行操作，比如文件 i 需要空间 $s_i=310$，存储在磁带的 121-430 单元，那么检索该文件需要的时间为 430. 问如何排列 n 个文件而使得平均检索时间最少？设计算法求解这个问题，说明算法的设计思想，证明算法的正确性，给出算法最坏情况下的时间复杂度.

解：采用贪心法.
算法：
1. 排序文件使得

$$\frac{f_1}{s_1} \ge \frac{f_2}{s_2} \ge \ldots \ge \frac{f_n}{s_n}$$

2. 把文件按照标号从小到大的顺序存入磁带.
最坏情况下时间复杂度是 $O(n\log n)$.
设最优解 OPT 的磁盘文件顺序是 $i_1,i_2,\ldots,i_n$，则平均检索时间是

$$t(\text{OPT}) = f_{i_1}s_{i_1} + f_{i_2}(s_{i_1}+s_{i_2}) + f_{i_3}(s_{i_1}+s_{i_2}+s_{i_3}) + \ldots + f_{i_n}(s_{i_1}+\ldots+s_{i_n})$$

假设在 OPT 中存在逆序，一定存在相邻的逆序，比如第 i 个文件和第 j 个文件构成相邻的逆序. 交换文件 i 和 j 得到解 P，那么

$$t(\text{OPT}) - t(P) = [f_i s_i + f_j(s_i + s_j)] - [f_j s_j + f_i(s_j + s_i)]$$

$$= f_j s_i - f_i s_j \ge 0 \qquad (因为\frac{f_i}{s_i} \le \frac{f_j}{s_j})$$

从而证明了从 OPT 出发，交换具有逆序的相邻元素后仍旧得到最优解. 每做 1 次交换，消除 1 个逆序. 至多经过 $n(n-1)/2$ 次交换，就消除了所有的逆序，从而得到算法的解，因此算法的解也是最优解.