中国科学技术大学

2019—2020 学年 第 1 学期考试试卷

考试科目: 算法基础　　得分: _____

学生所在系: _____　姓名: _____　学号: _____

装订线 答题时不要超过此线

一、基础题: （共 30 分，每小题 5 分）

1. 已知，$f(n) = 2n^2 \log^2 n$，$g(n) = n^2$，请证明 $f(n) = \Omega(g(n))$。

2. 请问下述四个关键字中哪些可以取代 **XX** 从而使关键字序列: 103，78，95，83，XX，89，93 成为某棵二分检索树的一个合理检索序列？

a) 76　b) 79　c) 85　d) 91　　答案: C

3. 已知数组 A[1..605]中存放了 605 个数据（非整数），现在需要同时找出数组 A 中的最小值和最大值，请问最少需进行多少次元素间的比较？

a) 905　b) 906　c) 907　d) 908　e) 其它　　答案: b

4. 求解递归方程 $T(n) = 4T(n/3) + 5n$。（给出推导过程）

5. 已知 $T(0) = 0$，$T(1) = 2$，对所有 $n>1$，$T(n) = T(n/2) + T(n/5) + 2n$，请求解该递归方程。（给出推导过程）

6. 已知某棵红黑树 T 共有 25 个内部结点，请问 T 的黑高度的最大值可以是多少？其黑高度的最小值可以是多少？【注：从某个结点 x 出发（不含该结点）到达叶结点的任意一条简单路径上的黑色结点个数（含叶结点）称为该结点的黑高度，记为 bh(x)】

二、计算题: （共 40 分，每题 10 分）

1. 现在要求出 6 个矩阵的链乘 $A_1 \times A_2 \times A_3 \times A_4 \times A_5 \times A_6$，其中 A_i 是一个 $P_{i-1} \times P_i$ 的矩阵，$P[0..6] = (4, 20, 4, 30, 15, 10, 4)$ 请计算其最优的乘法次序，给出计算过程。

2. 已知数组 S[1..m]，对字符集 Σ 中的任一字符 x 定义如下的函数 GS(x)，

$$GS(x) = \begin{cases} \min\{\, i : 1 \le i < m \text{ 且 } S[m-i] = x \,\} & \text{如果 x 在 S[1..m-1]中出现} \\ m & \text{如果 x 属于Σ，但 } x \text{ 不在S[1..m-1]中出现} \end{cases}$$

设 S[1..15]=(A, C, B, A, B, F, C, B, A, D, E, A, C, A, F)；字符集 Σ=（A，B，C，D，E，F，G，H），求Σ中字符对应的 **GS(x)** 函数值。

3. 已知有如图 1 所示的 Fibonacci 堆，请给出在其上执行 EXTRACT-MIN 操作的过程和结果。（请画图表示）

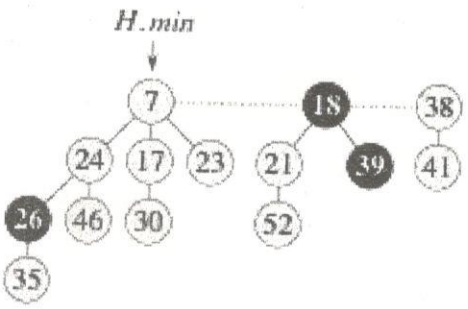


图 1: Fibonacci 堆

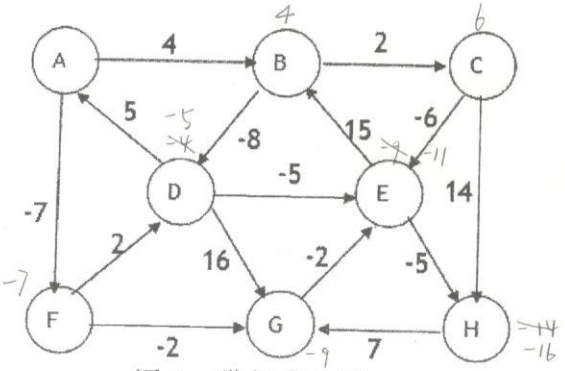


图 2: 带权有向图

4. 已知，有如图 2 所示的带权有向图，请计算并给出由结点 A 至其余各个结点的最短距离和对应的最短路径。

① Bellman-Ford

② Dijkstra

三、问答题：（共 10 分，每小题 5 分）

1. 请问是否存在两个函数 $f(n)$ 和 $g(n)$，使得 $f(n) \neq O(g(n))$ 且 $f(n) \neq \Omega(g(n))$？如果存在，请给出函数，如果不存在，请加以证明。

2. 已知有 n 个数据组成的序列（数据非整数），这些数据可以相互比较大小，现在要求出其中的最长单调增子序列，请问该怎样做才能使所用的数据元素间比较次数最少？分析你的方法所需的比较次数（只需说明方法和简单的分析，无需写算法）

法一：排序后的序列求最长公共子序列。 $O(n\log n + n^2) = O(n^2)$

法二：

四、算法设计：（共 20 分，每题 10 分）。

1. 设 G = (V, E) 是一个有向图，请设计算法判断图 G 中是否存在回路？要求算法的最坏情况时间复杂度为 $O(|V|+|E|)$。

2. 已知王先生共有 M 万元人民币的资金可用于投资。现在有 n 个投资项目可供其选择（假设所有投资项目的投资风险相同，投资时间长度也相同），这 n 个项目分别记作 $A_1, A_2, A_3, \ldots, A_n$，其中项目 A_i 可以接纳的最大资金量是 V_i 万元，可获最大收益是 P_i 万元，如果对项目 A_i 的投资额小于 V_i 万元，其投次收益按实际投资比率计算。请设计算法，计算王先生投资这 n 个项目所能获得的最大投资收益，并分析算法的最坏情况时间复杂度。

分数背包.

— 完. —

算法基础 2019-2020 学年第一学期期末考试参考答案

一、 基础题

1. (5 分) 证明：当$n \geq 2$时，$f(n) = 2n^2 \log^2 n \geq 2n^2 = 2g(n) \geq 0$，即存在$c = 2,\ n_0 = 2$，使得对所有$n \geq n_0$，有$f(n) \geq cg(n) \geq 0$成立，故$f(n) = \Omega(g(n))$.

2. (5 分) 答案：c
   说明：由二叉树的性质可知，在查找x的检索序列中，比x大的序列是逆序，比x小的序列是顺序，即 103>95>XX>93 或 78<83<XX<89<93，故选 c.

3. (5 分) 答案：b
   说明：同时找出最小值和最大值，可以对元素两两成对处理，每次比较一对元素，将较大的和当前最大值比较，较小的和当前最小值比较，因此每 2 个元素需要 3 次比较。当元素个数 n 为奇数，先把一个元素设为最大值和最小值的初值，剩下的元素成对处理，共需$(n-1)/2 \times 3 = 906$次比较。

4. (5 分) 解：（主方法）

   递归式$T(n) = 4\left(\frac{n}{3}\right) + 5n$中，$a = 4, b = 3, f(n) = 5n$，$\log_3 4 > 1$，故存在

   某个常数$\varepsilon > 0$，有$f(n) = O(n^{\log_3 4 - \varepsilon})$，对应主定理的第 1 种情况，故

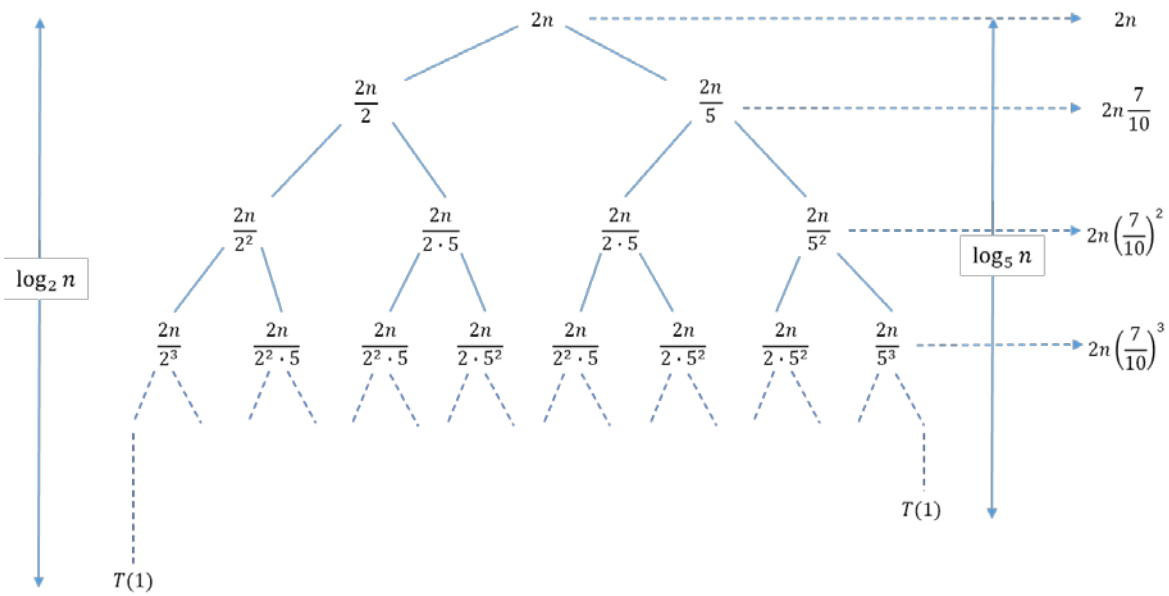
   $T(n) = \Theta(n^{\log_3 4})$

5. (5 分) 解：

   （方法一）代入法证明$T(n) = O(n)$
   假设正常数c，使得对所有$m < n$，有$T(m) \leq cm$成立。
   当$n = 0$时，$T(0) = 0 \leq c \cdot 0$成立.
   当$n = 1$时，要使$T(1) = 2 \leq c$成立，需要$c \geq 2$.
   则$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{5}\right) + 2n \leq \frac{c}{2}n + \frac{c}{5}n + 2n = (\frac{7}{10}c + 1)n.$

   当$\frac{7}{10}c + 1 \leq c$即$c \geq 4$时，$T(n) \leq cn$成立，故$T(n) = O(n)$.

（方法二）构造递归树如下



树高的上界为$\log_2 n$，下界为$\log_5 n$，
根据每一层的代价，求整棵树的代价的上下界.

上界：$T(n) \le 2n\left(1+\frac{7}{10}+\left(\frac{7}{10}\right)^2+\cdots+\left(\frac{7}{10}\right)^{\log_2 n-1}\right)+T(1)=$

$2n\frac{1-\left(\frac{7}{10}\right)^{\log_2 n}}{\frac{3}{10}}+2=\frac{20n}{3}\left[1-\left(\frac{7}{10}\right)^{\log_2 n}\right]+2$

下界：$T(n) \ge 2n\left(1+\frac{7}{10}+\left(\frac{7}{10}\right)^2+\cdots+\left(\frac{7}{10}\right)^{\log_5 n}\right)=2n\frac{1-\left(\frac{7}{10}\right)^{\log_5 n+1}}{\frac{3}{10}}=$

$\frac{20n}{3}\left[1-\left(\frac{7}{10}\right)^{\log_5 n+1}\right]$

（代入法 O(n)???）

6. (5 分) 解： $3 \le bh(x) \le 4$

上界：记黑高为$bh(x)$，则红黑树的内部结点至少有$2^{bh(x)}-1$个（所有结点都是黑色且是满二叉树的情况），故$2^{bh(x)}-1 \le 25$，可得$bh(x) \le 4$.

下界：设树高为h，则树的内部结点至多有2^h-1个，故$2^h-1 \ge 25$，$h \ge 5$，由于红黑树中，红结点的孩子必须是黑结点，故$bh(x) \ge \mathrm{h}/2 = 5/2$，故$bh(x) \ge 3$.

二、 计算题

1. (10 分)

$$m[i,j] = \begin{cases} 0 & if\ i = j \\ \min\limits_{i \le k < j}\{m[i,k] + m[k+1,j] + p_{i-1}p_k p_j & if\ i < j \end{cases}$$

因为P[0..6] = (4,20,4,30,15,10,4)
故$A1$:4*20, $A2$:20*4, $A3$:4*30, $A4$:30*15, $A5$:15*10, $A6$:10*4
$P0$=4, $P1$=20, $P2$=4, $P3$=30, $P4$=15, $P5$=10, $P6$=4

m[1,1]=0, m[2,2]=0, m[3,3]=0, m[4,4]=0,m[5,5]=0, m[6,6]=0

m[1,2]=m[1,1]+m[2,2]+P0P1P2=4*20*4=320 k=1
m[2,3]=m[2,2]+m[3,3]+P1P2P3=20*4*30=2400 k=2
m[3,4]=m[3,3]+m[4,4]+P2P3P4=4*30*15=1800 k=3
m[4,5]=m[4,4]+m[5,5]+P3P4P5 = 30*15*10=4500 k=4
m[5,6]=m[5,5]+m[6,6]+P4P5P6 = 15*10*4=600 k=5

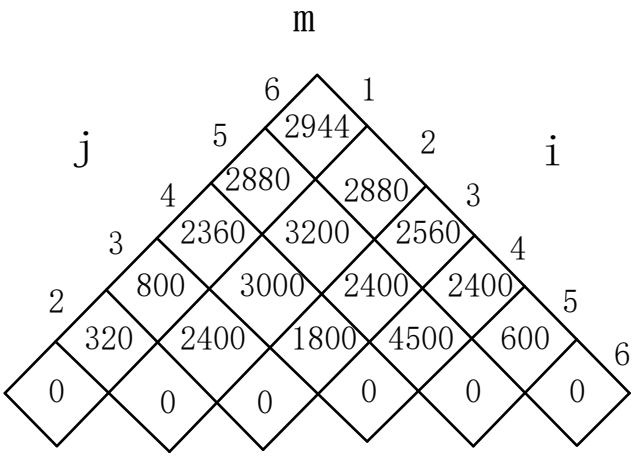
m[1,3] = min(
m[1,1]+m[2,3]+$P0P1P3$ = 2400+4*20*30=4800
m[1,2]+m[3,3]+$P0P2P3$ = 320+4*4*30=800 k=2
)
m[2,4] = min(
m[2,2]+m[3,4]+$P1P2P4$ = 1800+20*4*15=3000 k=2
m[2,3]+m[4,4]+$P1P3P4$ =2400+20*30*15=11400
)
m[3,5] = min(
m[3,3]+m[4,5]+$P2P3P5$ = 4500+4*30*10=5700
m[3,4]+m[5,5]+$P2P4P5$ =1800+4*15*10=2400 k=4
)
m[4,6] = min(
m[4,4]+m[5,6]+$P3P4P6$ = 600+30*15*4=2400 k=4
m[4,5]+m[6,6]+$P3P5P6$ =4500+30*10*4=5700
)

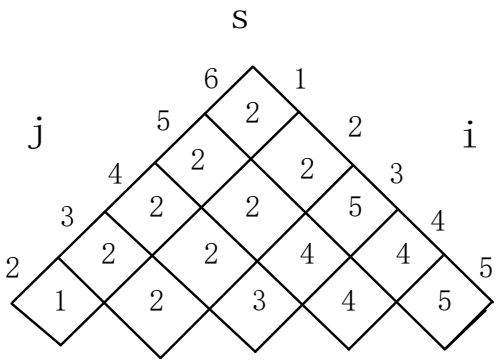
m[1,4] = min(
m[1,1]+m[2,4]+$P0P1P4$ = 3000+4*20*15=4200
m[1,2]+m[3,4]+$P0P2P4$ =320+1800+4*4*15=2360 k=2
m[1,3]+m[4,4]+$P0P3P4$ = 800+4*30*15=2600
)
m[2,5] = min(
m[2,2]+m[3,5]+$P1P2P5$ = 2400+20*4*10=3200 k=2
m[2,3]+m[4,5]+$P1P3P5$ =2400+4500+20*30*10=12900

m[2,4]+m[5,5]+$P_1P_4P_5$ = 3000+20*15*10=6000
)
m[3,6] = min(
m[3,3]+m[4,6]+$P_2P_3P_6$ = 2400+4*30*4=2880
m[3,4]+m[5,6]+$P_2P_4P_6$ =1800+600+4*15*4=2640
m[3,5]+m[6,6]+$P_2P_5P_6$ = 2400+4*10*4=2560 k=5
)

m[1,5]=min(
m[1,1]+m[2,5]+$P_0P_1P_5$=3200+4*20*10=4000
m[1,2]+m[3,5]+$P_0P_2P_5$=320+2400+4*4*10=2880 k=2
m[1,3]+m[4,5]+$P_0P_3P_5$=800+4500+4*30*10=6500
m[1,4]+m[5,5]+$P_0P_4P_5$=2360+4*15*10=2960
)
m[2,6]=min(
m[2,2]+m[3,6]+$P_1P_2P_6$=2560+20*4*4=2880 k=2
m[2,3]+m[4,6]+$P_1P_3P_6$=2400+2400+20*30*4=7200
m[2,4]+m[5,6]+$P_1P_4P_6$=3000+600+20*15*4=4800
m[2,5]+m[6,6]+$P_1P_5P_6$=3200+20*10*4=4000
)

m[1,6]=min(
m[1,1]+m[2,6]+$P_0P_1P_6$=2880+4*20*4=3200
m[1,2]+m[3,6]+$P_0P_2P_6$=320+2560+4*4*4=2944 k=2
m[1,3]+m[4,6]+$P_0P_3P_6$=800+2400+4*30*4=3680
m[1,4]+m[5,6]+$P_0P_4P_6$=2360+600+4*15*4=3200
m[1,5]+m[6,6]+$P_0P_5P_6$=2880+4*10*4=3040
)





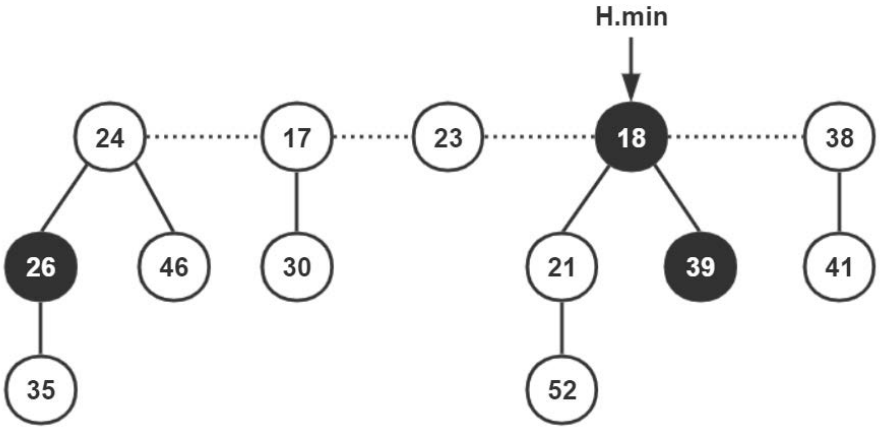
所以最优乘法次序应该为(*A*1*A*2)(((*A*3*A*4)*A*5) *A*6)。

2. (10 分) 若 x 在 S[1..m-1]中，GS(x)表示自 S[m-1]起从右往左数第 GS(x)个数是第一个等于 x 的数；若 x 不在 S[1..m-1]中，GS(x)=m。字符集对应的 GS(x)函数值如下：

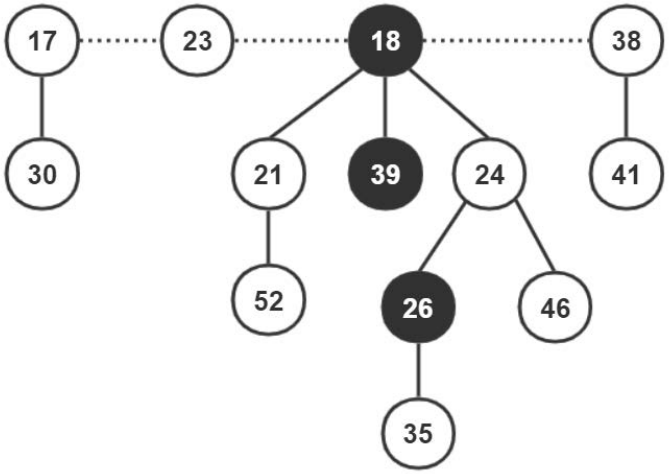
x	A	B	C	D	E	F	G	H
GS(x)	1	7	2	5	4	9	15	15

3. (10 分)

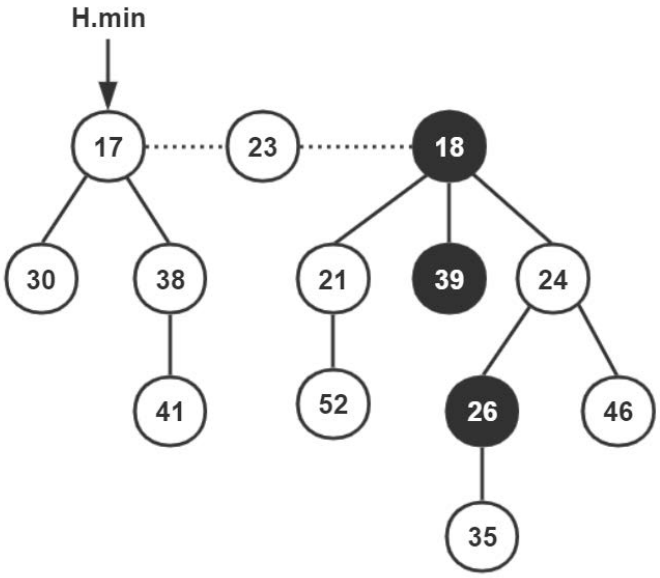
Step 1：将最小结点的孩子变为根节点



Step 2：合并根为 24 和根为 18 的结点

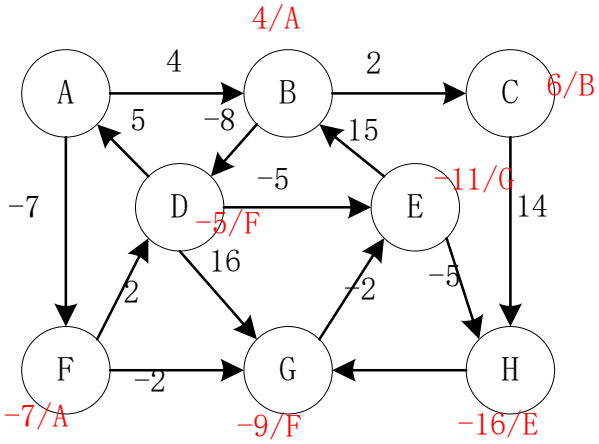


Step 3：合并根为 17 和根为 38 的结点



4. (10 分)

方法一： 使用 Bellman-Ford 算法

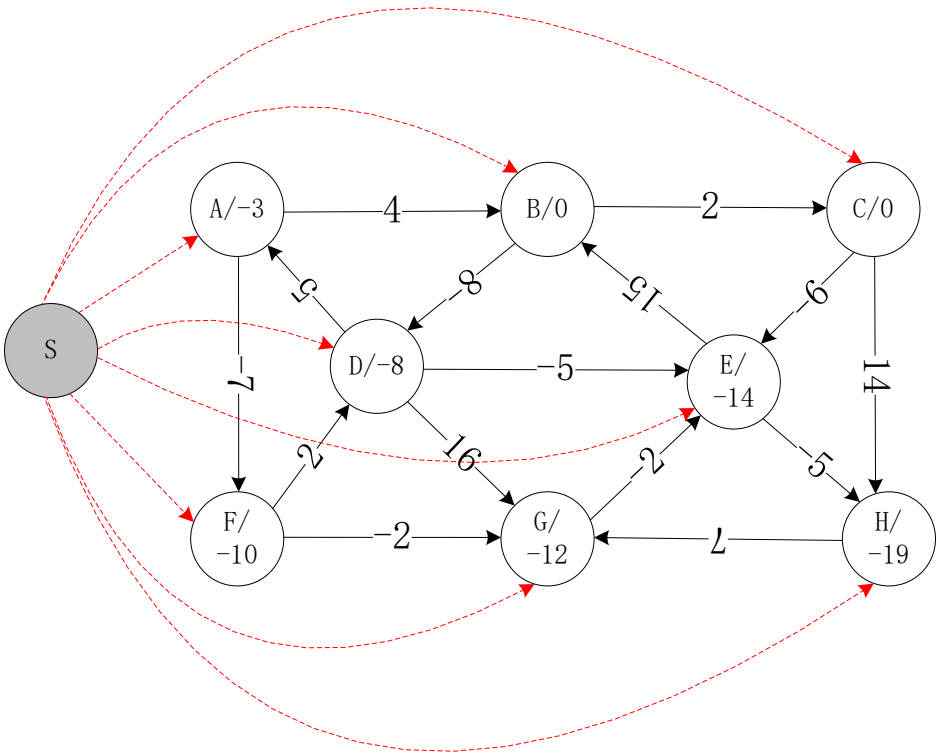


使用 Bellman-Ford 算法，利用对边的松弛操作来降低最短路径

B: 4 AB / AFGEB
C: 6 ABC
D: -5 AFD
E: -11 AFGE
F: -7 AF
G: -9 AFG
H: -16 AFGEH

方法二：利用 Johnson 算法求解

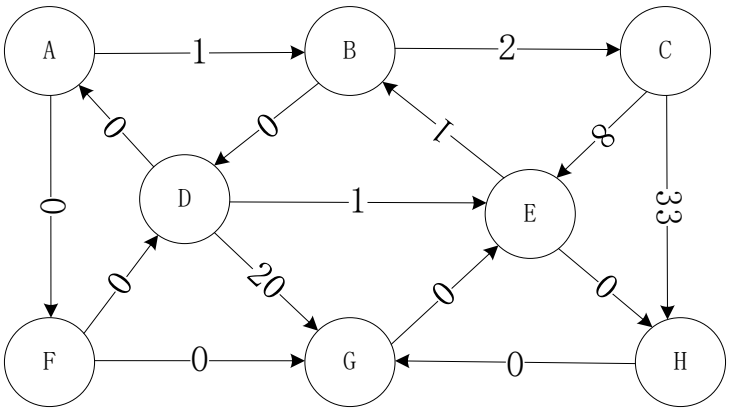
构造一个新结点 S，且令 S 到图中所有点的距离都为 0，然后使用 Bellman-Ford 算法求出 S 到图中所有点的距离。



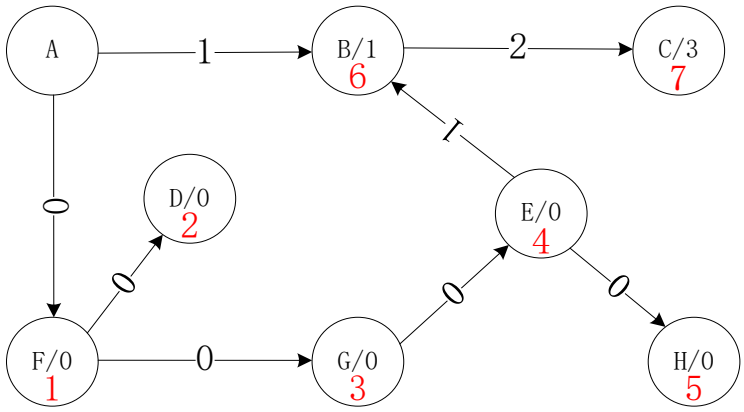
如上图，每个结点内部标记的数据即是$h(v) = \delta(s, v)$的值。

利用 h(v)函数计算图中点对间新的路径长度$\hat{w}(u, v) = w(u, v) + h(u) - h(v)$.

得到新路径图如下所示：



利用 Dijkstra 算法寻找上图中 A 到其余结点的最短路径。
过程如下：



其中结点内部黑色的值表示（新图中）该点到点 A 的最短距离，红色值表示结点被访问的顺序。
由上图可知 A 到各点的最短路径，结合原图可知 A 到各点最短路径长度：

A——>B：最短路径为 A-B 或者 A-F-G-E-B，最短路径长度为 4
A——>C：最短路径为 A-B-C，最短路径长度为 6
A——>D：最短路径为 A-F-D，最短路径长度为-5
A——>E：最短路径为 A-F-G-E，最短路径长度为-11
A——>F：最短路径为 A-F，最短路径长度为-7
A——>G：最短路径为 A-F-G，最短路径长度-9
A——>H：最短路径为 A-F-G-E-H，最短路径长度为-16

三、问答题

1.（5 分）
存在，如$f(n)=\sin n, g(n)=\cos n.$

2.（5 分）
方法一：转化为 LCS 问题求解

设序列为 L=<a1,a2,…,an>，先将该序列按递增序列排序得到新的序列为 X=<b1,b2,…,bn>，显然 X 与 L 的最长公共子序列即为 L 的最长递增子序列。
时间复杂度分析：利用快速排序或归并排序对序列 L 排序，时间复杂度为$O(nlgn)$，对两个长度都为 n 的序列求最长公共子序列，时间复杂度为$O(n^2)$，所以总的时间复杂度为$O(n^2)$。

方法二：动态规划法求解

设序列为 L=<a1,a2,…,an>，令 dp[i] 表示以 ai 结束的所有递增子序列的最长长度，有如下递推式：
dp[1] = 1
dp[i] = max{1, dp[j]+1} | aj<ai && j<i}

具体算法如下：
pre[1-n] 表示 ai 所在的最长递增子序列中的 ai 前面那个数的下标。

```
for ( i=2; i<=n; i++ )
    for( j=i-1; j>=1; j-- ){
        dp[i] = 1
        if( aj < ai && dp[j]+1 > dp[i] ){
            dp[i] = dp[j] + 1;
            pre[i] = j;
        }
    }
```

最后只需再遍历一次 dp 数组找到最大值的下标并利用 pre 找出最长递增序列即可。

时间复杂度分析：该算法有两层循环，复杂度为$O(n^2)$。

四、算法设计

1.（10 分）
方法一：深度优先遍历

对图进行深度优先遍历，如果在遍历的过程中发现一条深度遍历路线中有结点第二次被访问到，那么有环。可以用一个变量标记某结点的访问状态：未访问过，正在访问（其后结点还没有全部访问），访问过（其后结点都已经访问过），然后判断每一个结点的深度遍历路线即可。
采用邻接表存储，只需存储 e 条边和 n 个结点，每个结点和每个边都只会访问一次，故最坏时间复杂度为$O(n+e)$.

方法二：拓扑排序

方法是重复删除一个入度为 0 的顶点，将该顶点从图中删除并将该结点及其所有的出边从图中删除（即该结点指向的结点的入度减 1），最终若图中所有点都被删除则没有环路，否则存在环。
同样采用邻接表存储，遍历所有边求各个顶点入度的时间复杂度为$O(e)$，即边的个数。找出入度为 0 的结点的时间复杂度是$O(n)$。遍历所有边删除入度为 0 的结点的出边的复杂度为$O(e)$。故总的时间复杂度为$O(n+e)$。

具体算法如下：
设原图为 G
初始： 求每个顶点的入度 degree
　　并将入度为 0 的顶点加入到队列 queue 中
while (queue 不为空)
　tmp_u = queue.top()
　for tmp_u 到达的每个顶点 v
　　删除边(tmp_u, v)，v.degree--
　　if (v.degree==0) queue.push(v)
　queue.pop()
　将顶点 tmp_u 从图 G 中删除
if (G 为空) 不存在环
else　　存在环

(注：如果采用邻接矩阵存储则时间复杂度为$O(n^2)$)

2. (10 分)

该问题的实质为分数背包:n 件物品，背包承重：M ，物品价值：P_i ，物品重量为 V_i
算法思路为：先按 P_i /V_i（单位万元收益）对项目进行排序，然后采用贪心算法，优先选择收益较高的项目；

```
GREEDY-PROJECT(n,M,A) //A 为结构体，里面有 P，V，w 属性
for i=1 to n
    A[i].w=A[i].P/A[i].V
sort(A) //按 A[i].w 的大小对 A 数组进行降序排序
ans=0
for j=1 to n
    if(M>A [i].V)
        ans+= A[i].P
        M-= A [i].V
    else
        ans+= A[i].P * M/ A [i].V
        break
return ans
```

上述算法的时间复杂度为排序的时间+O(n)
排序算法可使用快速排序算法，其最坏情况的时间复杂度为 $O(n^2)$
使用归并排序算法和堆排序算法，其最坏情况的时间复杂度为 O(nlgn)