中南大学考试试卷

学年 2 学期　　时间 100分钟

算法分析与设计　课程 48 学时 3 学分 考试形式： 闭 卷

专业年级：　总分 100分，占总评成绩 60 %

注：此页不作答题纸，请将答案写在答题纸上

一、判断正误题，正确的后面标 T，错误的后面标 F（本题 10 分，每空 1 分）

1. 好的算法在很大程度上取决于问题中数据所采用的数据结构。
2. 深度优先搜索 (DFS) 是一种线性时间算法。
3. 用贪婪算法解决零钱兑换问题时，总能找到问题的最优解。
4. 给定 n 个整数 a1, . . . , an,其中第三小的数可以在 O(n) 时间内计算出来。
5. 背包问题的最佳解决方案始终包含具有最大单位价值比 vi/ ci 的对象 i。
6. 考虑加权有向图 G =（V; E; w），令 X 为一条最短 s-t 路径，s, t ϵV。 如果将图中每条边的权重加倍，对于每条边 e, w'（e）= 2w（e），则 X 仍将是（V; E; w'）中的最短 s-t 路径。
7. .具有至少三个顶点的简单、无向、连通、加权图，图中最重的边不在最小生成树中。
8. 假设数据结构上的每个操作都分摊在 O(1)时间内运行。则在初始为空的数据结构中执行 n 个操作序列的运行时间为 O(n)。
9. 每个有向无环图都只有一个拓扑顺序.
10. 给定一个由 n 个整数组成的数组，每个整数都属于{-1; 0; 1}，在最坏的情况下，可以按 O（n）时间对数组进行排序。

二、 简答题（本题 12 分，每小题 6 分）

1. **试比较分支限界法与回溯法的异同。**
2. **（1）如果我们可以在多项式时间内解决一个 NPC 问题，那么 NP 中的所有其他问题都一定可以在多项式时间内解决吗？ 试证明你的答案是正确的。（2）如果我们可以在 $O(n^{2021})$ 时间内解决一个 NPC 问题，那么 NP 中的所有其他问题是否一定可以在 $O(n^{2021})$ 时间内解决？试证明你的答案是正确的。**

三、计算与算法应用题（本题 48 分，每小题 12 分）

1. **假设您必须在下列三种算法中选择一个来解决某问题：**
   1) **算法 A 通过递归求解大小为 n/2 的8八个实例，然后在 $O(n^3)$ 时间内组合它们的解来求解大小为 n 的实例。**
   2) **算法 B 通过递归求解大小为 n/3 的 20 个实例，然后在 $O(n^2)$ 时间内组合它们的解来求解大小为 n 的实例。**
   3) **算法 C 通过递归求解两个大小为 2n 的实例，然后在 O(n) 时间内组合它们的解来求解大小为 n 的实例。**

   其中，哪一个算法更可取，为什么？

2. 给定一个由 **n** 个不同整数组成的未排序数组以及将会实施m次查询。 每次查询都是在数组中搜索一个整数，然后报告“找到”或“未找到”。假设 $m = \lfloor\sqrt{n}\rfloor$，您会选择通过对未排序数组做顺序查找执行每次查询还是先对数组进行预排序以加快查询速度？ 如果 $m = \lfloor\sqrt{\log n}\rfloor$呢？ 试证明你的答案是合理的。

3. 一个小偷进入房子抢劫。 他携带的包里最多可以携带 **10** 公斤的物品。 房子里有 **5** 件物品，其重量和价值如下表。 如果每件物品要么完全拿走要么不拿，小偷应该拿走些什么物品使得总价值最大？

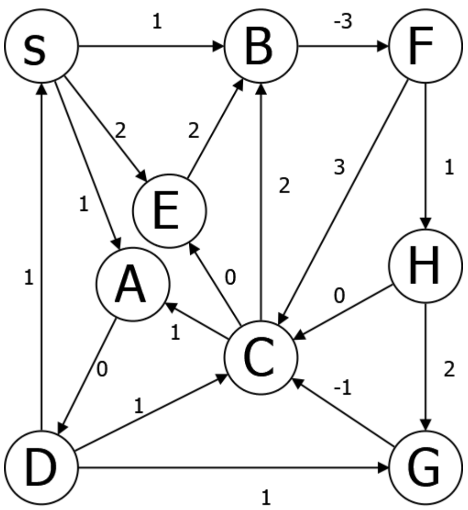
物品	重量 **(kg)**	价值 **($)**
镜子	**2**	**1**
银块	**3**	**2**
绘画	**3**	**5**
花瓶	**4**	**9**
雕刻	**6**	**4**

（**1**）写出递归方程；

（**2**）请按执行过程填写下面表格并标注出最优解。

（**3**）从空间性能角度给出您的优化方案。

4. 在下图中求从 **s** 到其他顶点的最短路径长度。



四、算法设计题（本题 30 分，每小题 15 分，第 1, 2 题选做 1 题, 第 3 题必做）

1. 在基于DFS的求有向图中的极大连通子图的算法中有个步骤是图的转置。假设给定图G以邻接链表形式表示，给出具有线性时间的算法，该算法计算出该图的转置图 G^T 。

2. 有 n 个程序和长度为 L 的磁带，程序 i 的长度为 a_i，已知 $\sum_{i=1}^{n} a_i \succ L$，求最优解($x_i$，$x_2$ ，. . . ，x_i，…， x_n)，x_i =0，1， x_i =1，表示程序 i 存入磁带，x_i =0，表示程序 i 不存入磁带，满足 $\sum_{i=1}^{n} x_i a_i \le L$，且存放的程序数目最多。

3. A 是一个由不同整数组成的 m × n 矩阵，这样每一行从左到右排序，每一列从下到上排序。即，对于每个 i ∈{1,...,m} 和每个 j ∈{1,...,n}，A[i,j] < A[i,j + 1] （当 j < n) 和 A[i,j] > A[i + 1,j]（当 i < m 时）。 我们说 A 是有序的。 比如下面的矩阵M被排序，M[1,3] = 9.

$$M = \begin{bmatrix} 7 & 8 & 9 \\ 3 & 4 & 6 \\ 1 & 2 & 5 \end{bmatrix}$$

为下面问题设计算法，给定一个排序的 m×n 矩阵 A 和一个整数 x，确定 x 是否出现在 A 中。

(a) (3分) 假设 A 是一个 1 × n 排序矩阵（即行向量）。给出一个时间复杂度为 O(logn) 的算法来确定 x 是否出现在 A 中。

(b) (6 分) 假设 A 是一个 n × n 排序矩阵。给出一个时间复杂度为 O(nlogn) 的算法来确定 x 是否出现在 A 中。简洁地论证为什么你的算法是正确的，并证明运行时间是 O(nlogn)。

（c）(6 分) 现在将尝试为问题获得 O(n) 时间算法。下面的递归算法 Exists 参数为 A、两个整数 r 和 c 以及 x，并在 A 的子矩阵中搜索 x，该子矩阵由 1 到 r 的行和 1 到 c 的列组成。如果 x 在子矩阵中，Exists(A,r,c,x) 返回 TRUE，否则返回 FALSE。

```
Exists(A,r,c,x):
  if r < 1 or c < 1: return FALSE
  if A[______] = x: return _____________
  else if A[___________] > x: return _______________
      Else: return ________
```

填写算法中的空白。其中一些是与 r 和 c 相关的整数，其中一些是 TRUE 或 FALSE，还有一些是对 Exists 的递归调用。验证Exists(A,n,n,x)在A为n×n时的运行时间复杂为O(n)。