算法导论 期末试题

一、填空题16%

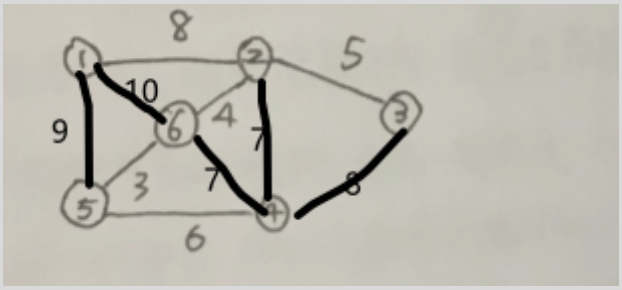
1. $T(n) = 9T(n/3) + n$ 的复杂度 **O(n^2)**
2. 哈夫曼编码之后，叶节点有n个，求整个树有多少个结点 **2n-1**
3. 给定切割cut(S,T)，f(S,T)最大有可能等于c(S,T)吗（判断正误）**有可能**
4. 也是类似上面的一个题目不记得了，也是切割和流的一个判断题
5. a,b,c三个结点分别在三个子树（α，β，γ）中，问如果对X进行左旋操作，那么a,b,c的深度如何变化

```
    |                          Y
    Y                        β  γ伽马
   / \                      X
  X   γ(伽马)               α
 / \
α   β          答：a深度+1，c深度不变，b如果在β左子树不变  如果在β右子树-1
```

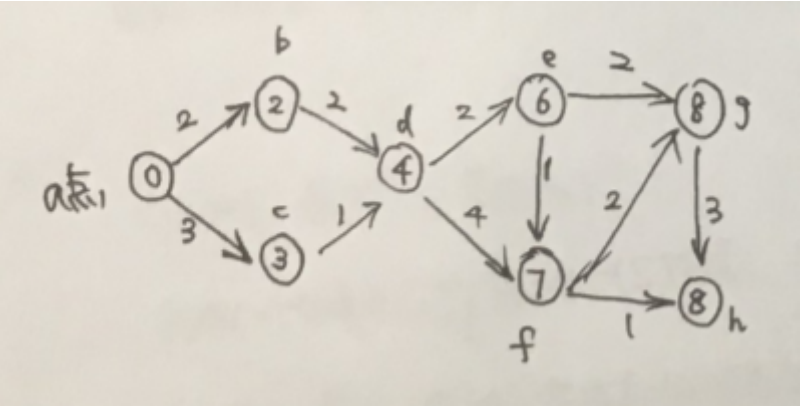
6. edmonds-karp算法和ford-fulkerson方法的区别：
   - **答**：ff采用了深度优先搜索找出增广路径，路径中所有的边容量减去这条增广路的流量，并建立流量为增广路增加流量相反数的反向边，O(F *|E|) F是最大流的值(进行F次深搜)；ek采用的是广度优先找出增广路径，当第一次到达汇点开始增加流量与建立反向边，O(V x E^2) BFS找增广路的时间复杂度是 O(E)，最多需要 O(V·E) 次查询
7. 忘记了好像很简单，也是考学过的算法的一个时间复杂的的题
8. 最小优先队列提取top-k个元素，时间复杂度是多少 **O(klgn)**

二、简答题46%

1. prim和dijkstra对于新节点的选取的方法是怎样的？两个算法的相似之处和不同之处是什么
   - **答**：选取方法：prim每一步在连接集合A与A之外的节点的所有边中，选择一条轻量级边加入A；dijk重复从节点集V-S中选择最短路径估计节点最小的节点u加入集合S；想似之处都利用了贪心的思想选取最小边，不同：Dijkstra的松弛操作考虑多个节点的权值和，而Prim考虑只有相邻两个节点的权值。
2. 证明dijkstra算法为什么不允许图中有负权重的边
   - **答**：因为dijktra算法每一次将一个节点加入已访问集合之中，之后不能在更新，如果遇到负权重可能会使已访问集合内节点之间的距离变小，却无法更新。
3. 画出图中的最小生成树



4. 用dijkstra算法来求出以a为起始的各个其他点的最短路径



5. 用栈来实现队列，并对入队（ENQUEUE）和出队（DEQUEUE）操作设计其摊还代价，使其代价为O(1)
6. 为什么G图中达到最大流为f时，图残存网络Gf中没有增广路径了

三、论述题48%

1. 若BFS和DFS对于一个**无环**图G的生成树一模一样，证明G=T（T就是最小生成树）

   答：当深度优先和广度优先的结果完全一样图有如下特点：1、出度大于1的节点的所有儿子出度为0，否则不能保证BFS与DFS生成树相同，同时图中又无环，所以该图的生成树只有一种，就是最小生成树(仅供参考)

2. 若图G是流网络，则对于每条边都有 c(e)=1，给定参数k，让你执行删除G中的k条边，使得图G中的最大流尽可能地变小，如何删除请设计算法

   答：使得图G中的最大流尽可能地变小，流网络的每条边的容量都为1，类似于二分图最大匹配问题转用最大流算法求解类似问题：

   - 如果是单源或者单汇点图，直接找到单源/单汇点的出/入边进行删除
   - 如果是多源且多汇点图，比较所有源的出度与所有汇点的入度，哪个小从哪边开始删

3. 给你一个整数数组 nums ，你可以对它进行一些操作。

   每次操作中，选择任意一个 nums[i] ，删除它并获得 nums[i] 的点数。之后，你必须删除 所有 等于 nums[i] - 1 和 nums[i] + 1 的元素。

   开始你拥有 0 个点数。返回你能通过这些操作获得的最大点数。请设计算法，写出返回值为分数最大的伪代码，并分析代码执行时间(力扣740)

   答

```cpp
class Solution {
public:
    int deleteAndEarn(vector<int>& nums) {
        int maxn = 0;
        for(int num: nums)
            if(num > maxn) maxn = num;
        vector<int> sum(maxn+1); //类似计数排序，开一个容量为maxnum + 1的数组
        for(int num: nums)
            sum[num] += num;  //把题中提供nums中大小相同的数字num求和并放入sum[num]中
        return fun(sum); //nums 223334对应 sum 00494
    }
    int fun(vector<int> sum){
        int len = sum.size();
        int first = sum[0]; //动态规划，因为相邻的会删除，所以隔一个数采用动态规划
        int second = max(sum[0],sum[1]);
        for(int i = 2; i<len; i++){
            int temp = second;
            second = max(second, first + sum[i]); //这行second始终领先first两位(隔一个)
            first = temp;
        }
        return second; //复杂度 O(len(nums) + maxnum)
    }
};
```

4. 有两个字符串s1,s2，字符串的字符都由"x"和"y"组成。你可以任意地交换s1[i]和s2[j]，交换只能发生在两个不同的字符串之间，绝对不能发生在同一个字符串内部。也就是说，我们可以交换 s1[i] 和 s2[j]，但不能交换 s1[i] 和 s1[j]。请设计算法，使得s1=s2的步数最小，返回步数，如果两者不可能通过交换实现相等，则返回-1。并分析你的时间复杂度。

   答：贪心

   - 对于s1，s2两个字符串对于同样的**i** 一样 + 不一样(YX or XY)
   - 一次遍历找出YX与XY的数量
   - 如果(YX+XY) % 2 不是0，那么说明没法 成功返回 -1
   - 如果是0 返回次数XY/2 + YX/2 + (XY%2) * 2;