**数据结构部分**

1. **判断题 （12\*2’=24’）**
   1. n^(log(log(log(n)))) = O(log(n!))
   2. 如果访问顺序不具有局部性，则伸展树不能保证分摊O(log(n))的性能
   3. 赫夫曼树交换不同深度子树, 编码成本必然增加
   4. 交换序列中的任一一对逆序元素，逆序对数必然减少
   5. 未改进过的next表，KMP算法可以在线性时间完成串匹配
   6. 对于二叉树，知道先序遍历和后序遍历不能确定层序遍历
   7. 基数排序如果采用了不稳定的底层算法，则输出序列未必正确
   8. 对于叶节点数量为2018的树，层序遍历使用的辅助队列规模绝对不超过2018
   9. 有2019个叶节点的真二叉树数目少于2018对括号可以组成的合法表达式数目
   10. 在插入排序过程中，有序前缀每增加一个元素，序列中的循环节点即使不增加，亦不致减少
   11. 程序执行过程中，调用栈中若同时有多帧对应同一函数，则它们必然依次紧邻排列
   12. 在理想随机情况下，完全二叉堆插入操作平均只需O(1)，尽管最坏情况下为O(log(n))
2. **简答题 （8\*4’=32’）**（每题不超过80字）
   1. 相对于常规表达式，逆波兰式在求值计算时为何效率更高？前者转换为后者所需成本已相当于一次常规表达式求值，那么转换的意义何在？
   2. DFS中何时标记前向边？何时标记后向边？
   3. 相对于选择排序，插入排序有什么优点？
   4. Dijkstra用于稠密图时，为何使用多叉堆替换掉完全二叉堆？多叉堆的分叉数怎么确定？
   5. 与开放散列相比，封闭散列有什么优点？试举例说明两点
   6. 相比锦标赛树，败者树的优势是什么？
   7. 举应用场景例说明红黑树有AVL树不具有的优势
   8. KMP算法相对于蛮力算法的优势在什么条件下足够明显？为什么？
3. **第K大节点 (7’+ 4’+ 3’= 14’)**

typedef struct BinNode {

int size; //当前节点和孩子总数

struct BinNode \*lc, \*rc; //左右孩子

}BinNode;

BinNode\* rank(BinNode\* t, unsigned int k) {

// 0 <= k <= t.size

//在此处补充不超过12行的代码

//rank(t,k)实现以t为根的二叉树中按照后序找出返回k个节点，

//若目标节点为x，则计算时间空间复杂度均不可以超过O(depth(x)),即节点深度

//请不要直接模拟后序遍历，复杂度会超过要求，将不得分

}

1) 给出具体实现算法

2）说明原理（200字以内，可附一张插图）

3）证明时间空间复杂度（不超过120字）

**计算机组成部分**

**1. 判断题**

（1）MIPS五级流水中，充分设置功能单元可改善结构冲突

（2）若x是C语言中的int型，如果x>0，则x \* x>0

（3）冯氏体系计算机把程序当数据保存在内存中

（4）对机械硬盘，顺序读取速度快于随机读取

（5）CPI越小，执行程序用时就越短

**2. 填空题**

（1）+1234的32位补码为 （用16进制，小端表示）

（2）单精度浮点数 -27.625在IEEE754下表示为

（3）MIPS处理数据冲突的技术有 、 、 （任写三个即可）

（4）缓存的缺失类型有 、 、 （任写三个即可）·