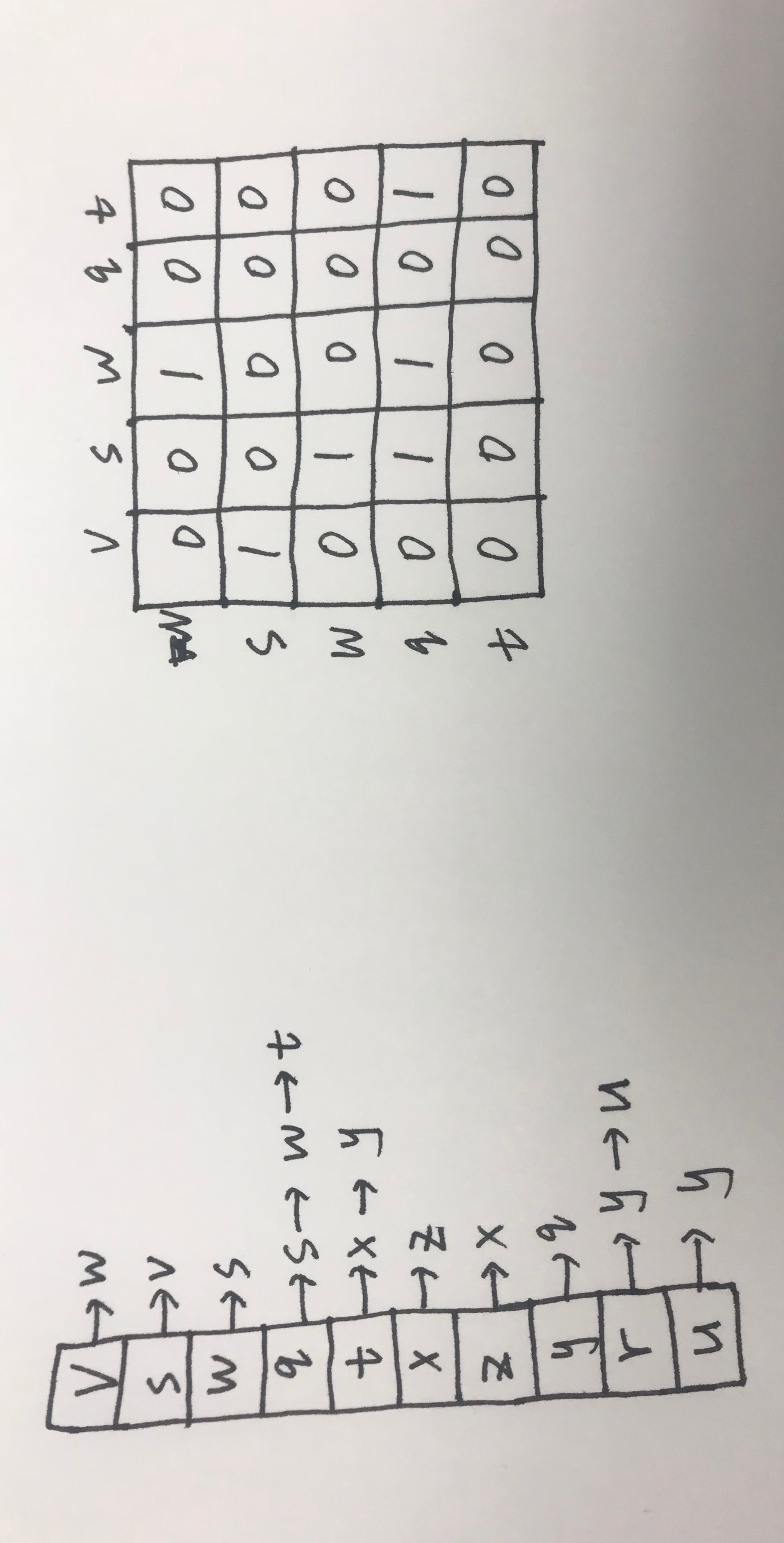
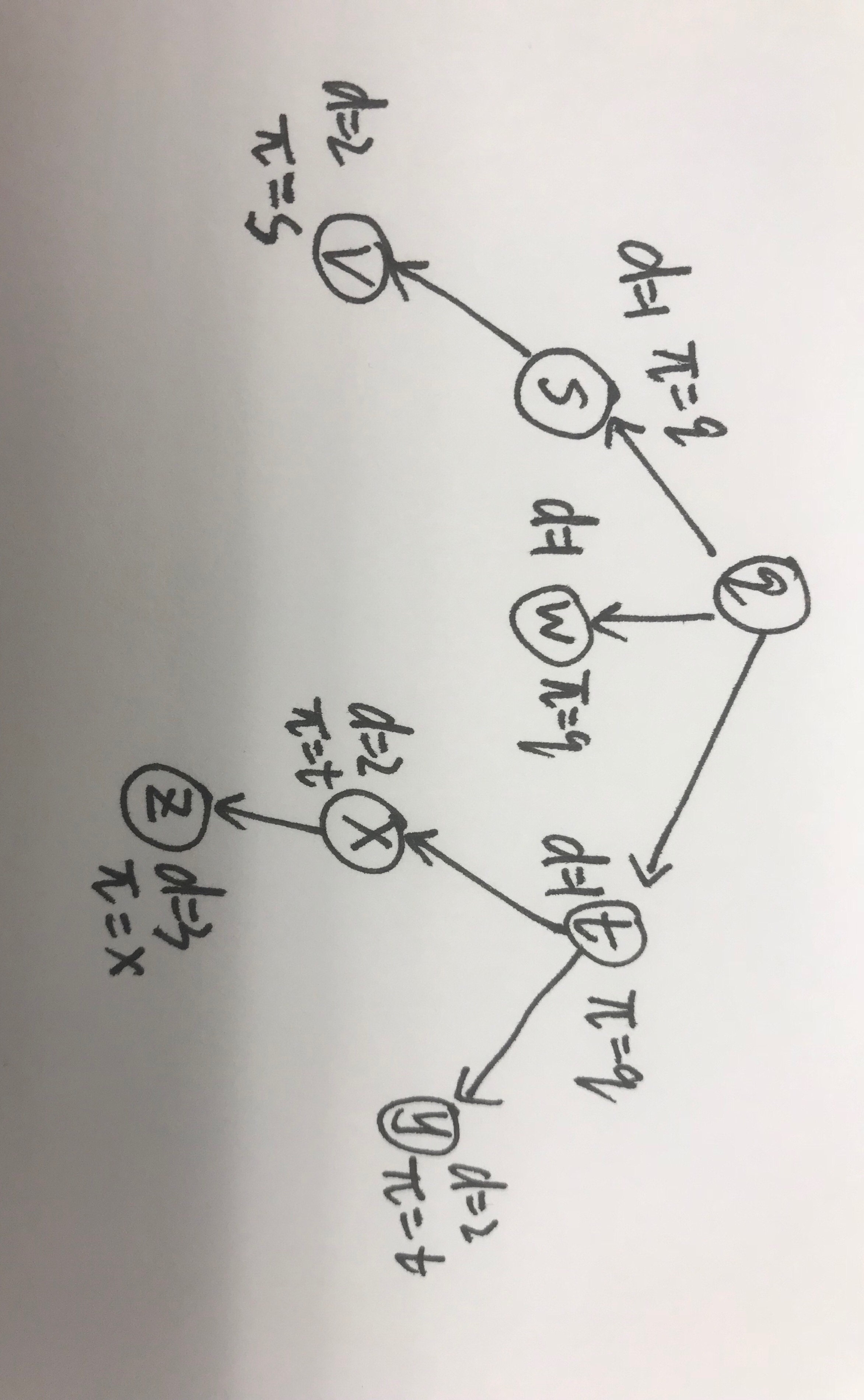
1.



14 entries

2.



3.I think the running time is also O(), because every node need to compare |V| times to test if current node has an edge with the node.

4.

|  |  |  |
| --- | --- | --- |
| Vertexs in traversal order | Discovery time | Finishing time |
| q | 1 | 16 |
| s | 2 | 7 |
| v | 3 | 6 |
| w | 4 | 5 |
| t | 8 | 15 |
| x | 9 | 12 |
| y | 10 | 11 |
| z | 13 | 14 |

5.DFS(G)

for each uG.V

u.colour=white

u.=nil

time =0

stack s

s.push(first vertex in G.V)

while(s is not empty)

v=s.top()

time = time +1

if v.colour=white

v.colour=white

v.colour=gray

v.d=time

for each uG.Adj[v]

if u.colour=white

u.=v

s.push(u)

else if v.colour=gray

s.pop()

v.colour=black

v.f=time

6. For adjacency list, G = (V, E) and consists of an array Adj of |v| lists. For every entry of vertices v in Adj[u], I would put it in a new list of where u in . The running time would be O(V + E).

For the adjacency matrix, I would just flip flop the rows and columns. The time algorithm is .

7. If we divide the input into groups of 7 instead of 5, the number of elements which are smaller than median of input is at least , so the algorithm in step 5, calls itself recursively on a problem of size at most -=, so

), we use substitution method to verify if T(n)

Check: T(n)

so

so we should choose n>56 then s constant c exists such that c>, when we choose n=56\*2, then , since the conditions for choosing c is satisfied, so the running time could be O(n) when we divided input into groups of 7.

Likewise, when we divide the input into groups of 3, the running time for this is not O(n), because according to recursion tree method, at each level of the tree we have a subproblem of size n and we perform O(n) work at each level of the tree, so overall running time cannot be linear.

8.In the algorithm,firstly I calculate the number of 15% of the generals which is m, then I use deterministicselect algorithm to find the mth largest of the generals, this will cost O(n), the I traverse the general list, find all of generals who is greater than the rank 15% general whose running time is O(n), so the total running time is O(n).

9.The recursion stops when , so i=log, the recursion takes O(n log), and then in every subarray of size k, we use insertion sort whose running time is , so the average time for this insertion sort is .

If k is chosen too big, then O(nk) is bigger than , so k must be O(logn),and it must be that)=O(nlogn), if the constant factors in big\_Oh notation is ignored, then O(nk+nlogn-nlogk)=O(nlogn), so k must be such that k<logk, which is impossible, ans the error comes from ignoring constant factors. So let be the constant factor in quicksort, and be the constant factor in insertion sort, so k must be chosen such that which requires . In practice, these constants cannot be ignored, and k should be chosen experimentally.

10. In this algorithm, I will traverse the list of g grudges, every time I put two people who has grudge with each other into separate rooms, if there is a pair of people that has already in the same room, then it is possible to assign all people into two rooms so that no one with a grudge is in the same room.

11.I use quicksort to sort the n music files, if the result is 0, let the computer to calculate again until the result is 1 or -1, in this case, the worst running time is also O(nlogn), because the comparison time is a constant.