September 16, 2018 练手题: 749,556,365,688,669,264,493,573,392

749

556

1. next permutation

```
class Solution {
    public int nextGreaterElement(int n) {
        if (n <= 10) {
            return -1;
        }
        char[] s = String.valueOf(n).toCharArray();
        // 1 2 3 5 1 => 1 2 5 1 3
        int i = s.length - 1;
        for (; i > 0; i--) {
            if (s[i - 1] < s[i]) {
                break;
            }
        }

        if (i == 0) {
            return -1;
        }
        i = i - 1;
        int j = s.length - 1;
        for (;j >= 0; j--) {
            if (s[j] > s[i]) {
                char tmp = s[j];
                s[j] = s[i];
                s[i] = tmp;
                break;
            }
        }
        j = s.length - 1;
        i = i + 1;
        while (i < j) {
            char tmp = s[j];
            s[j] = s[i];
            s[i] = tmp;
            i++;
            j--;
        }

        int res = Long.parseLong(new String(s)) <= Integer.MAX
_VALUE ? Integer.parseInt(new String(s)) : -1;

        return res > n ? res : -1;
    }
}
```

2. 同上：

```
class Solution {
public:
    int nextGreaterElement(int n) {
        string s = to_string(n);
        int l = s.size();
        int j = l-1;
        while(j && s[j]<=s[j-1]) --j;
        if(!j) return -1;
        int k = j;
        while(k<l-1 && s[k+1] > s[j-1]) ++k;
        swap(s[j-1], s[k]);
        reverse(s.begin()+j, s.end());
        long x = stol(s);
        if(x > long(INT_MAX)) return -1;
        return x;
    }
};
```

365

1. gcd:

```
class Solution {
    int gcd(int x, int y){
        if(x < y) swap(x, y);
        if(!y) return x;
        return gcd(y, x%y);
    }
public:
    bool canMeasureWater(int x, int y, int z) {
        if(x < y) swap(x, y);
        if(!y) return x==z || !z;
        if(z > x+y) return false;
        return z%gcd(x, y) == 0;
    }
};
```

688

1. DFS + Memo:

```
class Solution {
    typedef vector<double> vd;
    vector<vector<vd>> dp;
    int di[8] = {1, 1, 2, 2, -1, -1, -2, -2};
    int dj[8] = {2, -2, 1, -1, 2, -2, 1, -1};
    double dfs(int i, int j, int k, const int&N){
        if(i<0 || i>=N || j<0 || j>=N) return 0.;
        if(!k) return 1.;
        if(dp[i][j][k] >= 0) return dp[i][j][k];
        dp[i][j][k] = 0.;
        for(int t=0; t<8; ++t) dp[i][j][k] += 1./8. * dfs(i+di[t], j+dj[t], k-1, N);
        return dp[i][j][k];
    }
public:
    double knightProbability(int N, int K, int r, int c) {
        dp = vector<vector<vd>>(N+1, vector<vd>(N+1, vd(K+1, -1.)));
        return dfs(r, c, K, N);
    }
};
```

669

264

493

573

392