August 22, 2018 题目：324,844,393,289,272,674,787,615,405,744,414,564,567,838,172

324

1. easiest way is to sort the array then add the smaller one when index is even, larger one when index is odd. O(nlogn) time, O(n) space. Follow up is hard.

```
    public void wiggleSort(int[] nums) {
        int[] copy = nums.clone();
        Arrays.sort(copy);
        int i = (nums.length - 1) / 2;
        int j = nums.length - 1;

        int k = 0;
        while (k < nums.length) {
            if (k % 2 == 0) {
                nums[k++] = copy[i--];
            } else {
                nums[k++] = copy[j--];
            }
        }

    }
```

844

1. Scan from right to left, skip the # sign.
2. code :

```
class Solution {
    string trans(string s){
        string t;
        for(char c: s){
            if(c=='#'){
                if(!t.empty()) t.pop_back();
            }
            else t += c;
        }
        return t;
    }
public:
    bool backspaceCompare(string S, string T) {
        return trans(S) == trans(T);
    }
};
```

393

1. Understand the question is harder.
2. 看不懂题放弃
3. 很没意思的一道题，定义了一些规则，外加一些隐藏规则，然后判定这些规则是否都成立：

```
class Solution {
public:
    bool validUtf8(vector<int>& data) {
        if(data.empty()) return true;
        int n = 0;
        for(int x: data){
            if(!n){
                while(n<8 && 1&(x>>(7-n))) ++n;
                if(n==1 || n>4) return false;
                if(n) --n;
            }
            else{
                if(!(1&(x>>7)) || 1&(x>>6)) return false;
                --n;
            }
        }
        return !n;
    }
};
```

289

1. Two pass in place:

```
class Solution {
public:
    void gameOfLife(vector<vector<int>>& B) {
        if(B.empty() || B[0].empty()) return;
        int n = B.size(), m = B[0].size();
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j){
            for(int k=-1;k<=1;++k) for(int l=-1;l<=1;++l) if(l || k){
                int x = i + k, y = j + l;
                if(x>=0 && x<n && y>=0 && y<m && B[x][y]%2) B[i][j] += 2;
            }
        }
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j){
            if(B[i][j]/2 < 2 || B[i][j]/2 > 3) B[i][j] = 0;
            else if(B[i][j]%2==0 && B[i][j]/2==3) B[i][j] = 1;
            else B[i][j] %= 2;
        }
    }
};
```

272

1. 这题好灵性啊，你们都做过么 @Tongtong X @Chong T

```
class Solution {
public:
    vector<int> closestKValues(TreeNode* root, double target, int k) {
        stack<TreeNode *> L, R;
        vector<int> ans;
        for(auto p=root; p; ){
            if(p->val > target) {
                R.push(p);
                p = p->left;
            }
            else{
                L.push(p);
                p = p->right;
            }
        }
        while(k){
            if(L.empty() || (!R.empty() && R.top()->val + L.top()->val < 2*target)){
                ans.push_back(R.top()->val);
                auto p = R.top()->right;
                R.pop();
                while(p){
                    R.push(p);
                    p=p->left;
                }
            }
            else{
                ans.push_back(L.top()->val);
                auto p = L.top()->left;
                L.pop();
                while(p){
                    L.push(p);
                    p = p->right;
                }
            }
            --k;
        }
        return ans;
    }
};
```

674

1.

```
class Solution {
    public int findLengthOfLCIS(int[] nums) {
        if (nums == null || nums.length == 0) return 0;
        int start = nums[0];
        int res = 1;
        int max = 1;
        for (int i = 1; i < nums.length; i++) {
            if (nums[i] > nums[i - 1]) max++;
            else {
                max = 1;
                start = nums[i];
            }
            res = Math.max(max, res);
        }
        return res;
    }
}
```
2. One pass:
```
class Solution {
public:
    int findLengthOfLCIS(vector<int>& nums) {
        if(nums.empty()) return 0;
        int ans = 1;
        for(int i=1, cnt=1; i<nums.size(); ++i){
            if(nums[i] > nums[i-1]) ++cnt;
            else cnt = 1;
            ans = max(ans, cnt);
        }
        return ans;
    }
};
```

787

1. BFS & Dijkstra
```
class Solution {
    class Flight {
        int src;
        int dst;
        int k;
        int price;
        public Flight(int src, int dst, int k, int price) {
            this.src = src;
            this.dst = dst;
            this.k = k;
```

```java
            this.price = price;
        }
    }
    public int findCheapestPrice(int n, int[][] flights, int src, int dst, int K) {
        Map<Integer, List<Flight>> map = new HashMap<>();
        for (int[] f : flights) {
            if (!map.containsKey(f[0])) {
                map.put(f[0], new ArrayList<>());
            }
            map.get(f[0]).add(new Flight(f[0], f[1], -1, f[2]));
        }
        PriorityQueue<Flight> pq = new PriorityQueue<>((a, b) -> a.price - b.price);
        pq.offer(new Flight(0, src, K + 1, 0));
        while (!pq.isEmpty()) {
            Flight f = pq.poll();
            if (f.dst == dst) return f.price;
            if (f.k == 0) continue;

            if (!map.containsKey(f.dst)) return -1;
            List<Flight> list = map.get(f.dst);
            for (Flight l : list) {
                pq.offer(new Flight(l.src, l.dst, f.k - 1, l.price + f.price));
            }
        }
        return -1;
    }
}
```

2. DFS + memo:

```
class Solution {
    vector<unordered_map<int, int>> E;
    int dp[101][101], inf=1E9;
    int dfs(int i, int k, int&tar){
        if(i==tar) return 0;
        if(!k) return inf;
        if(dp[i][k] >= 0) return dp[i][k];
        dp[i][k] = inf;
        for(auto p: E[i]) dp[i][k] = min(dp[i][k], dfs(p.firs
t, k-1, tar) + p.second);
        dp[i][k] = min(dp[i][k], dfs(i, k-1, tar));
        return dp[i][k];
    }
public:
    int findCheapestPrice(int n, vector<vector<int>>& flights,
int src, int dst, int K) {
        E.resize(n);
        memset(dp, -1, sizeof(dp));
        for(auto vec: flights) E[vec[0]][vec[1]] = vec[2];
        int ans = dfs(src, K+1, dst);
        return (ans==inf?-1:ans);
    }
};
```

- [ ] 学楼上 @Zebo L

615

SQL

405

1. 要用Bit 否则不好处理负数：

```
class Solution {
    const string ref = "0123456789abcdef";
public:
    string toHex(int num) {
        string ans;
        for(int i=7; i>=0; --i){
            int idx = 0;
            for(int j=3; j>=0; --j) idx = idx*2 + (1&(num>>(i*
4 + j)));
            if(!ans.empty() || idx) ans += ref[idx];
        }
        return (ans.empty()?"0":ans);
    }
};
```

744

1. 2 ways O(N) and binary search O(logN)

```
class Solution {
    public char nextGreatestLetter(char[] letters, char targe
t) {
        /**char res = target;
        int min = Integer.MAX_VALUE;
        for (char c : letters) {
            if (c > target && (c - target) < min) {
                min = c - target;
                res = c;
            } else if (c < target && (c + 26 - target) < min)
{
                min = c + 26 - target;
                res = c;
            }
        }**/
        int i = 0, j = letters.length;
        while (i < j) {
            int mid = i + (j - i) / 2;
            if (letters[mid] > target) {
                j = mid;
            } else {
                i = mid + 1;
            }
        }
        return letters[i % letters.length];
    }
}
```

2. BI Search:

```
class Solution {
public:
    char nextGreatestLetter(vector<char>& letters, char targe
t) {
        int l = -1, r = letters.size();
        while(l < r - 1){
            int c = (l+r)/2;
            if(letters[c] <= target) l = c;
            else r = c;
        }
        return letters[r%int(letters.size())];
    }
};
```

414

1. 注意corner case

```
class Solution {
    public int thirdMax(int[] nums) {
        long max1 = Long.MIN_VALUE;
        for (int n : nums) {
            if (n > max1) {
                max1 = n;
            }
        }
        long max2 = Long.MIN_VALUE;
        for (int n : nums) {
            if (n > max2 && n < max1) {
                max2 = n;
            }
        }
        long max3 = Long.MIN_VALUE;

        for (int n : nums) {
            if (n > max3 && n < max2) {
                max3 = n;
            }
        }
        System.out.println(max1 + " " + max2 + " " + max3);

        return max3 != Long.MIN_VALUE ? (int) max3 : (int) max
1;
    }
}
```

2. Corner 太多就用 `long` 呗, 方正 `input` 是 `int` ,肯定不可能比 `INT_MIN` 小。

```
class Solution {
public:
    int thirdMax(vector<int>& nums) {
        long a = long(INT_MIN)-1, b = long(INT_MIN)-1, c = lon
g(INT_MIN)-1;
```

```
        for(int n: nums){
            if(n > a){
                c = b;
                b = a;
                a = n;
            }
            else if(n<a && n > b){
                c = b;
                b = n;
            }
            else if(n<b && n>c) c = n;
        }
        return ((c==long(INT_MIN)-1)?a:c);
    }
};
```

564

1. 暴力枚举：

```
class Solution {
public:
    string nearestPalindromic(string n) {
        int l = n.size();
        if(l == 1){
            if(n == "0") return "1";
            else return to_string(stoi(n) - 1);
        }
        set<long> pool{-1L, long(2E18 + 2)};
        pool.insert(stol(string(1, '1') + string(l-1, '0') + s
tring(1, '1')));
        pool.insert(stol(string(l-1, '9')));
        for(int j=1; j<=(l+1)/2; ++j){
            string s = n.substr(0, j);
            long x = stol(s);
            string a = to_string(x+1) , b = to_string(x), c =
to_string(x-1);
            string ra = a, rb = b, rc = c;
            reverse(ra.begin(), ra.end());
            reverse(rb.begin(), rb.end());
            reverse(rc.begin(), rc.end());
            cout<<ra<<' '<<rb<<' '<<rc<<endl;
            if(a.size() * 2 <= l) pool.insert(stol(a + string
(l-2*a.size(), '0') + ra));
            else pool.insert(stol(a + ra.substr(1)));
            if(b.size() * 2 <= l){
                pool.insert(stol(b + string(l-2*b.size(), '9')
+ rb));
                pool.insert(stol(b + string(l-2*b.size(), '0')
+ rb));
            }
            else pool.insert(stol(b + rb.substr(1)));
            if(c.size() * 2 <= l) pool.insert(stol(c + string
(l-2*c.size(), '9') + rc));
            else pool.insert(stol(c + rc.substr(1)));
        }
        long x = stol(n);
        long y = *(--pool.lower_bound(x)), z = *pool.upper_bou
nd(x);
        if(x - y <= z - x) return to_string(y);
        return to_string(z);
    }
};
```

1. brute force

```
class Solution {
    public boolean checkInclusion(String s1, String s2) {
        int k = s1.length();
        int[] ch = new int[26];
        for (char c : s1.toCharArray()) {
            ch[c - 'a']++;
        }

        int i = 0;
        while (i + k <= s2.length()) {
            String t = s2.substring(i, i + k);
            int[] cur = Arrays.copyOf(ch, 26);
            boolean flag = true;
            for (char c : t.toCharArray()) {
                if (--cur[c - 'a'] < 0) {
                    flag = false;
                    break;
                }
            }
            i++;
            if (flag) return true;
        }
        return false;
    }
}
```

2. Sliding window:

```
class Solution {
public:
    bool checkInclusion(string s1, string s2) {
        if(s2.size() < s1.size()) return false;
        unordered_map<char, int> ref, cnt;
        for(char c: s1) ++ref[c];
        for(int i=0; i<s1.size(); ++i) ++cnt[s2[i]];
        if(cnt == ref) return true;
        for(int i=s1.size(); i<s2.size(); ++i){
            ++cnt[s2[i]];
            --cnt[s2[i-s1.size()]];
            if(!cnt[s2[i-s1.size()]]) cnt.erase(s2[i-s1.size
()]);
            if(cnt == ref) return true;
        }
        return false;
    }
};
```

838

1. 这是Two pass，One pass 也不难写：

```
class Solution {
public:
    string pushDominoes(string D) {
        D = "L" + D + "R";
        string ref;
        vector<int> pos;
        for(int i=0; i<D.size(); ++i) if(D[i] != '.'){
            pos.push_back(i);
            ref += D[i];
        }
        for(int i=1; i<pos.size(); ++i){
            if(ref[i]==ref[i-1]) for(int j=pos[i-1]+1; j<pos
[i]; ++j) D[j] = ref[i];
```

```
            else if(ref[i]=='L' && ref[i-1]=='R'){
                for(int l=pos[i-1]+1, r=pos[i]-1; l<r; ++l,--
r){
                    D[l] = ref[i-1];
                    D[r] = ref[i];
                }
            }
        }
        return D.substr(1, D.size()-2);
    }
};
```

172

1. must include 5 x 2 to get 0s. 5 is larger than 2 so just consider how many 5 it will multiply.

```
class Solution {
    public int trailingZeroes(int n) {
        if (n <= 4) return 0;
        return fiveNumber(n);
    }
    private int fiveNumber(int n) {
        if (n == 0) return 0;
        return n / 5 + fiveNumber(n / 5);
    }
}
```

2. Easy 难度：

```
class Solution {
public:
    int trailingZeroes(int n) {
        int ans = 0;
        n /= 5;
        while(n){
            ans += n;
            n /= 5;
        }
        return ans;
    }
};
```