Leetcode
stack/Maximal Rectangle-O
sort-merge/merge k sorted lists
Trie/单词搜索 II
string-re/ Regular Expression matching-O
we mean that the regex can only contain special character: * (star), . (dot), + (plus)
*: 0个或多个pre字符
.: 任何字符
+：1个或多个pre字符
Airbnb
ArrayList／锯齿数组的iterator 删除 - O
给一个2d array，要求写一个顺序访问这个2d array的Iterator，包括hasNext()与next()。注意2d array的每行中元素的个数可能不一样，也可能为空。followup是写一个remove()，注意是remove当前item，不是下一个item。remove是需要同时删除原来数组里的元素，也能在Iterator调用时体现
出来。

```
public static void main(String[] args) {
        test t = new test();
        Array2D<Integer> ad = t.new Array2D<Integer>();
        for(int i = 1 ; i < 3; i++){
                ArrayList<Integer> a = new ArrayList<Integer>();
                for(int j = 2; j < 4; j++){
                        a.add(i*j);
                        System.out.println(i* j);
                }
                ad.addLine(a);
        }
        for(Iterator<Integer> i = ad.iterator(); i.hasNext();){
                Integer num = i.next();
                System.out.print(num + " ");
                if (num == 3)
                        i.remove();
        }
        System.out.println();
        for(Iterator i = ad.iterator(); i.hasNext();){
                System.out.print(i.next() + " ");
        }
}
public class Array2D<T> {
        ArrayList<ArrayList<T>> array;
        public Array2D(){
                array = new ArrayList<ArrayList<T>>();
        }
        public void addLine(ArrayList<T> nums){
                array.add(new ArrayList(nums));
        }
        public T get(int x, int y){
                if(x >= array.size()) return null;
                ArrayList<T> l = array.get(x);
                if(l == null || y >= l.size()) return null;
```

```
        return l.get(y);
}
public Iterator iterator() {
        // TODO Auto-generated method stub
        return new a2Iterator();
}
private class a2Iterator implements Iterator<T>{
        int r;
        int c;
        ArrayList<T> curArray;
        public a2Iterator(){
                r = 0;
                c = 0;
        }
        @Override
        public boolean hasNext() {
                if(curArray == null && array.size() == 0){
                        return false;
                }else if(r >= array.size()) return false;
                return true;
        }
        @Override
        public T next() {
                if(c == 0) curArray = array.get(r);
                T ret = curArray.get(c);
                if(curArray.size()-1 == c){
                        r ++;
                        c = 0;
                }else
                        c ++;
                return ret;
        }
        @Override
        public void remove() {
                ArrayList pre = curArray;
                int x = c;
                int y = r;
                if(x == 0){
                        y--;
                        pre = array.get(y);
                        x = pre.size();
                }
                x--;
                pre.remove(x);
                if(pre.size() == 0){
                        array.remove(y);
                        r--;
                }
                if(c != 0) c--;
        }
```

```
        }
}
```

hashtable/分页-O

给出一个list：每个元素是［host_id, list_id, score, city］ 按score排好序，进行分页。每个页面不能重复有重复的host-id.使用LinkedHashMap，如果有相同id则放在下一页

保存一个最后能用的页面， 每当一个页面满了，更新哈希表

```
 public static void main(String[] args) {
                test t = new test();
                String[] source = new String[]{
                                "1,28,300.1,SanFrancisco",

                                "4,5,209.1,SanFrancisco",

                                "20,7,208.1,SanFrancisco",

                                "23,8,207.1,SanFrancisco",

                                "16,10,206.1,Oakland",

                                "1,16,205.1,SanFrancisco",

                                "6,29,204.1,SanFrancisco",

                                "7,20,203.1,SanFrancisco",

                                "8,21,202.1,SanFrancisco",

                                "2,18,201.1,SanFrancisco",

                                "2,30,200.1,SanFrancisco",

                                "15,27,109.1,Oakland",

                                "10,13,108.1,Oakland",

                                "11,26,107.1,Oakland",

                                "12,9,106.1,Oakland",

                                "13,1,105.1,Oakland",

                                "22,17,104.1,Oakland",

                                "1,2,103.1,Oakland",

                                "28,24,102.1,Oakland",

                                "18,14,11.1,SanJose",

                                "6,25,10.1,Oakland",
```

```
                                "19,15,9.1,SanJose",

                                "3,19,8.1,SanJose",

                                "3,11,7.1,Oakland",

                                "27,12,6.1,Oakland",

                                "1,3,5.1,Oakland",

                                "25,4,4.1,SanJose",

                                "5,6,3.1,SanJose",

                                "29,22,2.1,SanJose",

                };
                ArrayList<ArrayList<String>> ret = t.getPages(source,
12);
                int i = 0;
                for(ArrayList<String> list : ret)
                {
                        System.out.println(i++);
                        for(String str : list)
                                System.out.println(str);
                }
        }
public ArrayList<ArrayList<String>> getPages(String[] source, int k) {
        int firstEmptyPage = 0;
        ArrayList<ArrayList<String>> pages = new
ArrayList<ArrayList<String>>();
        pages.add(new ArrayList<String>());
        HashMap<Integer, Integer> map = new HashMap<Integer,
Integer>();
        for(String s : source){
                int id = getId(s);
                if(map.containsKey(id)){
                        int index = map.get(id) + 1;
                        if(index == pages.size()) pages.add(new
ArrayList());
                        pages.get(index).add(s);
                        map.put(id, index);
                }else{
                        map.put(id, firstEmptyPage);
                        pages.get(firstEmptyPage).add(s);
                        if(pages.get(firstEmptyPage).size() == k)
                                firstEmptyPage = updata(map, pages,
firstEmptyPage);
                }
```

```
        }
        return pages;
}
private int updata(HashMap<Integer,Integer> map,
ArrayList<ArrayList<String>> pages, int i){
        for(String s : pages.get(i)){
                int id = getId(s);
                if(map.get(id) == i) map.remove(id);
        }
        if(i == pages.size()) pages.add(new ArrayList<String>());
        return i+1;
}
private int getId(String s){
        int i = s.indexOf(',');
        return Integer.parseInt(s.substring(0, i));
}
```

DP-str/Edit distance/ Edit distance II-O
题目是给定一个word list 和一个target word，要求输出在word list 里跟target word的edit distance 相差不大于k的所有words。
给一个list，找出所有和target相似的words。
设计一个数据结构，不遍历list，也能找到所有。
需要prun， length > word2.length() + d 或小于 -d

```
public static void main (String[] args) {
        String[] dic = new String[]{"ab","dfs", "a","fsa"};
        ArrayList<String> ret = new test().getWords(dic, 2, "a");
        for(String str : ret)
                System.out.println(str);
        System.out.println("end");
}
class trieNode{
        boolean isEnd;
        trieNode[] nexts;
        public trieNode(){
                nexts = new trieNode[256];
        }
        public void add(String str){
                trieNode cur = this;
                for(char c : str.toCharArray()){
                        if(cur.nexts[c] == null) cur.nexts[c] = new
trieNode();
                        cur = cur.nexts[c];
                }
                cur.isEnd = true;
        }
}
public ArrayList<String> getWords(String[] dic, int d, String target){
        ArrayList<String> ret = new ArrayList<String>();
        trieNode root = new trieNode();
        for(String str : dic)
```

```
                root.add(str);
        int pre[] = new int[target.length() + 1];
        for(int i = 0; i < pre.length; i++)
                pre[i] = i;
        dfs(d,target,"", pre,root,ret);
        return ret;
}
private void dfs(int d, String target, String curStr,
                int[] pre, trieNode root, ArrayList<String> ret) {
        // TODO Auto-generated method stub
        if(root.isEnd){
                if(pre[target.length()] <= d) ret.add(curStr);
                else return;
        }
        for(int i = 0; i < 256; i++){
                if(root.nexts[i] == null) continue;
                int[] next = new int[target.length() + 1];
                next[0] = curStr.length() + 1;
                for(int j = 1; j < pre.length; j++){
                        if(target.charAt(j - 1) == i) next[j] = pre[j
- 1];
                        else next[j] = Math.min(pre[j],
Math.min(pre[j - 1], next[j - 1])) + 1;
                }
                dfs(d, target, curStr + (char)i, next, root.nexts[i],
ret);
        }
}

past
tree n-ary/URL Shortener-O
Given a method decode(testEncStr) which will return the decoded int id
if testEncStr is decodeable or will throw an exception (or return
null) if not, implement a new method decodeFind(String badEncStr)
which takes a string and returns the decoded int id. http://itjob.io/
post/348
greedy-jump game/ maximum subarray/不相邻maximum subarray
一个数组，选出不相邻子序列，要求子序列和最大，
[4,9,6]=10
[4,10,3,1,5]=15
follow up是求得到最优解时的request具体情况，类似于求出最短路径的长度后，follow
up给出最短路径。不过电面感觉不会问。
        public int fun(int[] A){
                int first = A[0];
                int second = Math.max(A[0], A[1]);
                for(int i = 2; i < A.length; i++){
                        int temp = Math.max(second, A[i] + first);
                        first = second;
                        second = temp;
                }
                return second;
```

```
        }
```

string/ CSV parse-O
题目是关于如何Parse csv file：举个例子，给定一个CSV文件，格式是 “some_name|some_address|some_phone|some_job”，要求输出Json format “{name:some_name, address:some_addres,phone:some_phone, job:some_job}”
特殊情为两个引号之间的分号，不可作为分割字符 http://itjob.io/post/349

```
/*
John,                    Smith,          john.smith@gmail.com,    Los Angeles,         1
Jane,                    Roberts, janer@msn.com,          "San Francisco, CA",  0
"Alexandra ""Alex""",    Menendez,       alex.menendez@gmail.com, Miami,               1
"""Alexandra Alex"""
John|                    Smith |         john.smith@gmail.com |   Los Angeles |       1
Jane|                    Roberts| janer@msn.com|          San Francisco, CA|   0
Alexandra "Alex"|        Menendez|        alex.menendez@gmail.com| Miami|               1
"Alexandra Alex"
*/
```

要考虑 quote, escape \得情况

```
class Solution {
          public static void main(String[] args) {
            //#1
            ArrayList<String> output = parseCSV("John,Smith,john.smith@gmail.com,Los Angeles,1");
            String strOutput = printStr(output);
            System.out.println(strOutput);
            //#2
            output = parseCSV("Jane,Roberts,janer@msn.com,\"San Francisco, CA\",0");
            strOutput = printStr(output);
            System.out.println(strOutput);
            output = parseCSV("\"Alexandra \"\"Alex \"\"\",Menendez,alex.menendez@gmail.com,Miami,1");
            strOutput = printStr(output);
            System.out.println(strOutput);
          }

          public static ArrayList<String> parseCSV(String str) {
            ArrayList<String> res = new ArrayList<String>();
            boolean inQuote = false;
            StringBuilder buffer = new StringBuilder();
            for(int i = 0; i < str.length(); i++) {
              if(inQuote) {
                if(str.charAt(i) == '"') {
                  if(i == str.length()-1) {
```

```
                    res.add(buffer.toString());
                    return res;
                  } else if(str.charAt(i+1) == '"') {
                    buffer.append('"');
                    i++;
                  } else {
                    res.add(buffer.toString());
                    buffer.setLength(0);
                    inQuote = false;
                    i++;
                  }
                } else buffer.append(str.charAt(i));
              } else {
                if(str.charAt(i) == '"') {
                  inQuote = true;
                } else if( str.charAt(i) == ',') {
                  res.add(buffer.toString());
                  buffer.setLength(0);
                } else {
                  buffer.append(str.charAt(i));
                }


              }

              }
              if(buffer.length() > 0) res.add(buffer.toString());
              return res;
            }

            public static String printStr(ArrayList<String> list) {
              StringBuilder res = new StringBuilder();
              for(int i = 0; i < list.size()-1; i++) {
                res.append(list.get(i));
                res.append('|');
              }
              res.append(list.get(list.size()-1));
              return res.toString();
            }
          }
public static ArrayList<String> parseCSV1(String str) {
          boolean mark = false;
          boolean dMark = false;
          ArrayList<String> ret = new ArrayList<String>();
          StringBuilder part = new StringBuilder();
          for (int i = 0; i < str.length(); i++) {
                  char c = str.charAt(i);
//                if (dMark) {
//                        if(c == '\"'){
//                                if (i == str.length() - 1) {
```

```
//                                      ret.add(part.toString());
//                                      return ret;
//                                  } else if (str.charAt(i + 1) ==
'\"') {
//                                      //""asdf""
//                                      dMark = false;
//                                      part.append('\"');
//                                      i++;
//                                  } else
//                                      part.append(c);
//                          }else part.append(c);
//                  } else
                    if (mark) {
                            if (c == '\"') {
                                    if (i == str.length() - 1) {
                                            ret.add(part.toString());
                                            return ret;
                                    } else if (str.charAt(i + 1) ==
'\"') {
                                            // “asdf“fdsa”fdas”
                                            part.append('\"');
                                            i++;
                                    } else
                                            mark = false;
                            } else
                                    part.append(c);
                    } else {
                            if (c == ',') {
                                    ret.add(part.toString());
                                    part.setLength(0);
                            } else if (c == '\"') {
                                    if (i == str.length() - 1) {
                                            ret.add(part.toString());
                                            return ret;
                                    }
//                                  else if (str.charAt(i + 1) == '\"')
{
//                                          part.append('\"');
//                                          dMark = true;
//                                          i++;
//                                  }
                                    else
                                            mark = true;
                            } else
                                    part.append(c);
                    }
            }
            ret.add(part.toString());
            return ret;
    }
```

```
public static String parseCSV1(String str) {
        StringBuilder ret = new StringBuilder();
        StringBuilder part = new StringBuilder();
        boolean isQuote = false;
        for (int i = 0; i < str.length(); i++) {
                char c = str.charAt(i);
                if (isQuote) {
                        if (c == '\"') {
                                if (i == str.length() - 1)
                                        return
ret.append(part.toString()).toString();
                                else if (str.charAt(i + 1) == '\"')
{
                                        part.append('\"');
                                        i++;
                                } else
                                        isQuote = false;
                        } else
                                part.append(c);
                } else {
                        if (c == ',') {
                                part.append('|');
                                ret.append(part.toString());
                                part.setLength(0);
                        } else if (c == '\"') {
                                isQuote = true;
                        } else
                                part.append(c);
                }
        }
        return ret.append(part.toString()).toString();
}
```

implementing a simple socket based client (could lookup docs online).

http://cs.lmu.edu/~ray/notes/javanetexamples/

http://blog.mianshi.me/