

```

1  /* Problem 5--Scrabble
2     This is simply a matter of repeated sorting.  Sort the dictionary by
3     value and then by alphabet.  Sort the words by letter.  Sort the
4     tiles by letter.  It's easy to find matches then. */
5
6  import java.io.*;
7  import java.util.*;
8
9  public class prob5 {
10
11     private static Scanner in;
12     private static PrintWriter out;
13     private static int cs;
14     private static int[] values; //the values of each letter
15     private static SS[] Dict; //The Dictionary
16
17     public static void main (String[] args) throws Exception {
18
19         in = new Scanner (new File ("prob5.in"));
20         out = new PrintWriter ("prob5.out");
21         cs = 1;
22         values = new int[] //Load up the values for each letter
23             {1,3,3,2,1,4,2,4,1,8,5,1,3,1,1,3,10,1,1,1,1,4,4,8,4,10};
24         int wct = Integer.parseInt (in.nextLine ());
25         Dict = new SS[wct];
26         for (int i=0; i < wct; i++) //Read in the dictionary
27             Dict[i] = new SS (in.nextLine());
28         DSort (Dict); //Sort the dictionary
29         while (in.hasNext()) { //Read every case
30             String word = in.nextLine();
31             Process (word);
32         }
33         in.close ();
34         out.close ();
35     }
36
37     //Accessor for values array
38     public static int[] getvalues() throws Exception {return values;}
39
40     //Processes each data case with the dictionary
41     public static void Process (String word) throws Exception{
42
43         out.print ("Case "+(cs++)+"": ");
44         String sword = Sort (word); //Sort the word by letter
45         for (int i=0; i < Dict.length; i++) //Find the first match
46             if (Match (Dict[i].getsword(),sword)) {
47                 out.print (Dict[i].getword()+"\r\n\r\n");
48                 return;
49             }
50     }
51
52     //Sort the word by letter using Shell Sort
53     public static String Sort (String word) throws Exception {
54
55         char[] wc = word.toCharArray ();
56         int d = wc.length;
57         while (d > 1) {
58             d = (d==2)?1:(d%2==1)?(d+1)/2:(d%4==0)?d/2+1:d/2+2;
59             for (int i=d; i < wc.length; i++)
60                 for (int j = i-d; j >= 0 && wc[j] > wc[j+d]; j-=d) {
61                     char t = wc[j];
62                     wc[j] = wc[j+d];
63                     wc[j+d] = t;
64                 }

```

```

65     }
66     return new String(wc);
67 }
68
69 //Sorts the Dictionary by value and then by word
70 public static void DSort (SS[] Dict) throws Exception {
71
72     int d = Dict.length;
73     while (d > 1) {
74         d = (d==2)?1:(d%2==1)?(d+1)/2:(d%4==0)?d/2+1:d/2+2;
75         for (int i=d; i < Dict.length; i++)
76             for (int j = i-d;
77                 j >= 0 && (Dict[j].getvalue() < Dict[j+d].getvalue() ||
78                     Dict[j].getvalue() == Dict[j+d].getvalue() &&
79                     Dict[j].getword().compareTo(Dict[j+d].getword())>0); j--=d) {
80                 SS t = Dict[j];
81                 Dict[j] = Dict[j+d];
82                 Dict[j+d] = t;
83             }
84     }
85 }
86
87 //Looks for a match. Easy since the strings are sorted.
88 public static boolean Match (String a, String b) throws Exception {
89
90     int i=0, j=0;
91     while (i < a.length() && j < b.length()) {
92         if (a.charAt(i)==b.charAt(j)) {i++;j++;} //Did we find a match?
93         else if (b.charAt(j) < a.charAt(i)) j++;
94         //Advance longer word if no match
95         else return false; //If we find a letter not in tiles, no match
96     }
97     if (i < a.length()) return false;
98     //Did we not make it through the word?
99     return true;
100 }
101 }
102
103 /* SS is a class containing a word in the dictionary */
104 class SS {
105
106     private String word, sword; //The word and the sorted word
107     private int value; //The value of the word
108
109     //Constructor method
110     public SS (String w) throws Exception {
111
112         word = w; //Get word
113         sword = prob5.Sort (w); //Sort word
114         value = 0; //Compute value
115         for (int i=0; i < word.length(); i++)
116             value += prob5.getvalues()[word.charAt(i)-'A'];
117     }
118
119     //Various accessors
120     public int getvalue () throws Exception {return value;}
121     public String getword () throws Exception {return word;}
122     public String getsword () throws Exception {return sword;}
123
124 }
125

```