

HÁSKÓLI ÍSLANDS

TÖLVUNARFRÆÐI 2

Vikublað 11

Höfundur:
Teitur Guðmundarson

Kennari:
Páll Melsteð

29. mars 2019



1. Insert the keys E A S Y Q U T I O N in that order into an initially empty table of $M = 5$ lists, using separate chaining. Use the hash function $11 \cdot k \% M$ to transform the k th letter of the alphabet into a table index.

```

1  key hash gildi
2    E   0   0
3
4  0 E0
5  1 null
6  2 null
7  3 null
8  4 null
9
10 key hash gildi
11   A   1   1
12
13  0 E0
14  1 A1
15  2 null
16  3 null
17  4 null
18
19 key hash gildi
20   S   4   2
21
22  0 E0
23  1 A1
24  2 null
25  3 null
26  4 S2
27
28 key hash gildi
29   Y   0   3
30
31  0 E0 -> Y3
32  1 A1
33  2 null
34  3 null
35  4 S2
36
37 key hash gildi
38   Q   2   4
39
40  0 E0 -> Y3
41  1 A1
42  2 Q4
43  3 null
44  4 S2
45
46 key hash gildi
47   U   1   5
48
49  0 E0 -> Y3
50  1 A1 -> U5
51  2 Q4
52  3 null
53  4 S2
54
55 key hash gildi
56   T   0   6
57

```

```
58 0 E0 -> Y3 -> T6
59 1 A1 -> U5
60 2 Q4
61 3 null
62 4 S2
63
64 key hash gildi
65   I   4   7
66
67 0 E0 -> Y3 -> T6
68 1 A1 -> U5
69 2 Q4
70 3 null
71 4 S2 -> I7
72
73 key hash gildi
74   0   0   8
75
76 0 E0 -> Y3 -> T6 -> 08
77 1 A1 -> U5
78 2 Q4
79 3 null
80 4 S2 -> I7
81
82 key hash gildi
83   N   4   9
84
85 0 E0 -> Y3 -> T6 -> 08
86 1 A1 -> U5
87 2 Q4
88 3 null
89 4 S2 -> I7 -> N9
```

2. Write a program to find values of a and M , with M as small as possible, such that the hash function $(a*k)\%M$ for transforming the k th letter of the alphabet into a table index produces distinct values (no collisions) for the keys S E A R C H X M P L . The result is known as a perfect hash function.

```
1 public static int[] reykjaHash() { Daemi2.java
2     int[] gildi = new int[2];
3
4     int[] bokstf = {19, 5, 1, 18, 3, 8, 24, 13, 16, 12};
5
6     for(int m = 2; m <= 100; m++) {
7         for(int a = 1; a <= 1000; a++) {
8             Set<Integer> hashes = new HashSet<>();
9
10            for(int i = 0; i < bokstf.length; i++) {
11                int hash = (a * k) % m;
12                hashes.add(hash);
13            }
14
15            if (hashes.size() == 10) {
16                gildi[0] = a;
17                gildi[1] = m;
18                return gildi;
19            }
20        }
21    }
22    return null;
23 }
```

3. Give the contents of a linear-probing hash table that results when you insert the keys E A S Y Q U T I O N in that order into an initially empty table of initial size $M = 4$ that is expanded with doubling whenever half full. Use the hash function $11k \% M$ to transform the k th letter of the alphabet into a table index.

```

1 key hass gildi
2   E   3   0
3
4 0 null      2 null
5 1 null      3 EO
6
7 key hass gildi
8   A   3   1
9
10 0 A1       2 null
11 1 null     3 EO
12
13 Hasstafla af tvöfaldri stærð og M verður að 8
14
15 0 null      4 null
16 1 null      5 null
17 2 null      6 null
18 3 null      7 null
19
20 endurinnsetning A1
21 key hass gildi
22   A   3   1
23
24 0 null      4 null
25 1 null      5 null
26 2 null      6 null
27 3 A1        7 null
28
29 endurinnsetning EO
30 key hass gildi
31   E   7   0
32
33 0 null      4 null
34 1 null      5 null
35 2 null      6 null
36 3 A1        7 EO
37
38 key hass gildi
39   S   1   2
40
41 0 null      4 null
42 1 S2        5 null
43 2 null      6 null
44 3 A1        7 EO
45
46 key hass gildi
47   Y   3   3
48
49 0 null      4 Y3
50 1 S2        5 null
51 2 null      6 null
52 3 A1        7 EO
53
54 Hasstafla af tvöfaldri stærð og M verður að 16
55

```

```

56 0 null      8 null
57 1 null      9 null
58 2 null     10 null
59 3 null     11 null
60 4 null     12 null
61 5 null     13 null
62 6 null     14 null
63 7 null     15 null
64
65 endurinnsetning A1
66 key hass gildi
67   A   11   1
68
69 0 null      8 null
70 1 null      9 null
71 2 null     10 null
72 3 null     11 A1
73 4 null     12 null
74 5 null     13 null
75 6 null     14 null
76 7 null     15 null
77
78 endurinnsetning E0
79 key hass gildi
80   E    7    0
81
82 0 null      8 null
83 1 null      9 null
84 2 null     10 null
85 3 null     11 A1
86 4 null     12 null
87 5 null     13 null
88 6 null     14 null
89 7 E0       15 null
90
91 endurinnsetning S2
92 key hass gildi
93   S    1    2
94
95 0 null      8 null
96 1 S2        9 null
97 2 null     10 null
98 3 null     11 A1
99 4 null     12 null
100 5 null     13 null
101 6 null     14 null
102 7 E0       15 null
103
104 endurinnsetning Y3
105 key hass gildi
106   Y    3    3
107
108 0 null      8 null
109 1 S2        9 null
110 2 null     10 null
111 3 Y3        11 A1
112 4 null     12 null
113 5 null     13 null
114 6 null     14 null
115 7 E0       15 null
116

```

```

117 key hass gildi
118   Q   11   4
119
120 0 null      8 null
121 1 S2        9 null
122 2 null      10 null
123 3 Y3        11 A1
124 4 null      12 Q4
125 5 null      13 null
126 6 null      14 null
127 7 E0        15 null

```

```

128
129 key hass gildi
130   U    7   5
131
132 0 null      8 U5
133 1 S2        9 null
134 2 null      10 null
135 3 Y3        11 A1
136 4 null      12 Q4
137 5 null      13 null
138 6 null      14 null
139 7 E0        15 null

```

```

140
141 key hass gildi
142   T   12   6
143
144 0 null      8 U5
145 1 S2        9 null
146 2 null      10 null
147 3 Y3        11 A1
148 4 null      12 Q4
149 5 null      13 T6
150 6 null      14 null
151 7 E0        15 null

```

```

152
153 key hass gildi
154   I    3   7
155
156 0 null      8 U5
157 1 S2        9 null
158 2 null      10 null
159 3 Y3        11 A1
160 4 I7        12 Q4
161 5 null      13 T6
162 6 null      14 null
163 7 E0        15 null

```

```

164
165 Hasstafla af tvöfaldri stærð og M verður að 32

```

```

166
167 0 null      16 null
168 1 null      17 null
169 2 null      18 null
170 3 null      19 null
171 4 null      20 null
172 5 null      21 null
173 6 null      22 null
174 7 null      23 null
175 8 null      24 null
176 9 null      25 null
177 10 null     26 null

```

```
178 11 null      27 null
179 12 null      28 null
180 13 null      29 null
181 14 null      30 null
182 15 null      31 null
183
184 endurinnsetning A1
185 key hass gildi
186   A   11   1
187
188 0 null      16 null
189 1 null      17 null
190 2 null      18 null
191 3 null      19 null
192 4 null      20 null
193 5 null      21 null
194 6 null      22 null
195 7 null      23 null
196 8 null      24 null
197 9 null      25 null
198 10 null     26 null
199 11 A1       27 null
200 12 null      28 null
201 13 null      29 null
202 14 null      30 null
203 15 null      31 null
204
205 endurinnsetning E0
206 key hass gildi
207   E   23   0
208
209 0 null      16 null
210 1 null      17 null
211 2 null      18 null
212 3 null      19 null
213 4 null      20 null
214 5 null      21 null
215 6 null      22 null
216 7 null      23 E0
217 8 null      24 null
218 9 null      25 null
219 10 null     26 null
220 11 A1       27 null
221 12 null      28 null
222 13 null      29 null
223 14 null      30 null
224 15 null      31 null
225
226 endurinnsetning I7
227 key hass gildi
228   I    3    7
229
230 0 null      16 null
231 1 null      17 null
232 2 null      18 null
233 3 I7       19 null
234 4 null      20 null
235 5 null      21 null
236 6 null      22 null
237 7 null      23 E0
238 8 null      24 null
```



```

239  9 null      25 null
240 10 null      26 null
241 11 A1        27 null
242 12 null      28 null
243 13 null      29 null
244 14 null      30 null
245 15 null      31 null
246
247 endurinnsetning Q4
248 key hass gildi
249   Q   27   4
250
251  0 null      16 null
252  1 null      17 null
253  2 null      18 null
254  3 I7        19 null
255  4 null      20 null
256  5 null      21 null
257  6 null      22 null
258  7 null      23 E0
259  8 null      24 null
260  9 null      25 null
261 10 null      26 null
262 11 A1        27 Q4
263 12 null      28 null
264 13 null      29 null
265 14 null      30 null
266 15 null      31 null
267
268 endurinnsetning S2
269 key hass gildi
270   S   17   2
271
272  0 null      16 null
273  1 null      17 S2
274  2 null      18 null
275  3 I7        19 null
276  4 null      20 null
277  5 null      21 null
278  6 null      22 null
279  7 null      23 E0
280  8 null      24 null
281  9 null      25 null
282 10 null      26 null
283 11 A1        27 Q4
284 12 null      28 null
285 13 null      29 null
286 14 null      30 null
287 15 null      31 null
288
289 endurinnsetning T6
290 key hass gildi
291   T   28   6
292
293  0 null      16 null
294  1 null      17 S2
295  2 null      18 null
296  3 I7        19 null
297  4 null      20 null
298  5 null      21 null
299  6 null      22 null

```

```

300  7 null      23 E0
301  8 null      24 null
302  9 null      25 null
303 10 null      26 null
304 11 A1        27 Q4
305 12 null      28 T6
306 13 null      29 null
307 14 null      30 null
308 15 null      31 null
309
310  endurinnsetning U5
311  key hass gildi
312    U      7    5
313
314  0 null      16 null
315  1 null      17 S2
316  2 null      18 null
317  3 I7        19 null
318  4 null      20 null
319  5 null      21 null
320  6 null      22 null
321  7 U5        23 E0
322  8 null      24 null
323  9 null      25 null
324 10 null      26 null
325 11 A1        27 Q4
326 12 null      28 T6
327 13 null      29 null
328 14 null      30 null
329 15 null      31 null
330
331  endurinnsetning Y3
332  key hass gildi
333    Y     19    3
334
335  0 null      16 null
336  1 null      17 S2
337  2 null      18 null
338  3 I7        19 Y3
339  4 null      20 null
340  5 null      21 null
341  6 null      22 null
342  7 U5        23 E0
343  8 null      24 null
344  9 null      25 null
345 10 null      26 null
346 11 A1        27 Q4
347 12 null      28 T6
348 13 null      29 null
349 14 null      30 null
350 15 null      31 null
351
352  key hass gildi
353    0      5    8
354
355  0 null      16 null
356  1 null      17 S2
357  2 null      18 null
358  3 I7        19 Y3
359  4 null      20 null
360  5 08        21 null

```

```
361 6 null      22 null
362 7 U5        23 E0
363 8 null      24 null
364 9 null      25 null
365 10 null     26 null
366 11 A1       27 Q4
367 12 null     28 T6
368 13 null     29 null
369 14 null     30 null
370 15 null     31 null
371
372 key hass gildi
373 N 26 9
374
375 0 null      16 null
376 1 null      17 S2
377 2 null      18 null
378 3 I7        19 Y3
379 4 null      20 null
380 5 08        21 null
381 6 null      22 null
382 7 U5        23 E0
383 8 null      24 null
384 9 null      25 null
385 10 null     26 N9
386 11 A1       27 Q4
387 12 null     28 T6
388 13 null     29 null
389 14 null     30 null
390 15 null     31 null
```

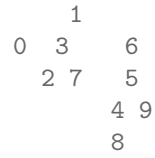
4. *Cuckoo hashing*. Develop a symbol-table implementation that maintains two hash tables and two hash functions. Any given key is in one of the tables, but not both. When inserting a new key, hash to one of the tables; if the table position is occupied, replace that key with the new key and hash the old key into the other table (again kicking out a key that might reside there). If this process cycles, restart. Keep the tables less than half full. This method uses a constant number of equality tests in the worst case for search (trivial) and amortized constant time for insert.

Hér þarf klára útfærsluna á get, put og delete í skránni CuckooST.java

1 `// code` Daemi4.java

5. Draw the tree corresponding to the `id[]` array depicted at right. Can this be the result of running weighted quick-union? Explain why this is impossible or give a sequence of operations that results in this array.

i	0	1	2	3	4	5	6	7	8	9
id[i]	1	1	3	1	5	6	1	3	4	5



Þetta tré getur ekki komið úr running quick-union. Hægra tréið: 1, 6, 5, 4, 9, 8 hefur hæðina 4 sem er meira en $\log N$.

6. *Random connections.* Develop a UF client ErdosRenyi that takes an integer value N from the command line, generates random pairs of integers between 0 and $N-1$, calling `connected()` to determine if they are connected and then `union()` if not (as in our development client), looping until all sites are connected, and printing the number of connections generated. Package your program as a static method `count()` that takes N as argument and returns the number of connections and a `main()` that takes N from the command line, calls `count()`, and prints the returned value. *Erdős-Renyi model.* Use your client from Exercise 1.5.17 to test the hypothesis that the number of pairs generated to get one component is $1/2N \ln N$. Hér þarf bara að skila einu forriti sem leysir seinna verkefnið.

```

1  import edu.princeton.cs.algs4.StdOut;
2  import java.util.ArrayList;
3  import java.util.List;
4
5  public class Daemi6 {
6      private class Tilraun {
7          int fjoldi;
8          int por;
9          public Tilraun(int fjoldi, int por) {
10              this.fjoldi = fjoldi;
11              this.por = por;
12          }
13      }
14      public static void main(String[] args) {
15          new Daemi6().geraTilraun();
16      }
17      private void geraTilraun() {
18          List<Tilraun> tilraunir = new ArrayList<>();
19          int fjoldi = 16;
20          for(int i = 0; i < 10; i++) {
21              int porTil = count(fjoldi, false);
22              Tilraun experiment = new Tilraun(fjoldi, porTil);
23              tilraunir.add(experiment);
24              fjoldi *= 2;
25          }
26      }
27      private static int count(int fjoldi, boolean oskar) {
28          int tengingar = 0;
29          UnionFind unionFind = new UnionFind(fjoldi);
30          while(unionFind.count() != 1) {
31              int stradi = StdRandom.uniform(fjoldi);
32              int viktor = StdRandom.uniform(fjoldi);
33              tengingar++;
34              if (oskar) {
35                  StdOut.println("Tengingar: " + stradi + " - " + viktor);
36              }
37              if (!unionFind.connected(stradi, viktor)) {
38                  unionFind.union(stradi, viktor);
39              }
40          }
41          return tengingar;
42      }
43  }

```
