EC 504
Spring, 2022
HW 2

Due Saturday, Feb. 19, 8PM on Gradescope.

1. (20 pts) Consider a Lazy Sorting Data Structure, which maintains two ArrayLists: unsorted [U] and sorted [S], and implements the following public methods:

   - INSERT(Integer x): inserts x into U
   - QUERY(Integer x): Performs a binary search to see whether x is in S, and if not found, searches linearly to see whether x is in U. Returns true if x is found in either array.

   The data structure has a private access to a resorting method, which works as follows:

   - RESORT(): Removes all elements from U and appends them to S, and then mergesorts the array.

   Compute (with a reasonable explanation) the amortized cost of a worst-case sequence of n Lazy Data Structure public operations in these cases.

   (a) All elements are stored in U so one never does a sort.
   Solution: Consider INSERT. The time for n insertions will be $\Theta(n)$, as ArrayList does insertion in amortized cost $\Theta(1)$. The amortized analysis is needed for resizing the array every time it gets large, where the array size is typically doubled and elements have to be copied. If we simply pay a cost of 2 per insert, this takes care of the extra cosf of copying when the array size is increased.

   Now consider QUERY. The cost of a single query is the size of the list, which therefore will be $\Theta(n)$. There is no way to reduce this.

   Hence, a worst case sequence of operations will insert $n/2$ elements at $\Theta(1)$ complexity each, then perform $n/2$ queries, each of which will be $\Theta(n)$. The amortized complexity for the operations will be $\Theta(n)$.

   (b) RESORT is called whenever the size of U reaches 10. **Solution:** Let's consider a sequence of n insertions, and try to compute the average cost per insertion. The total number of operations looks like:

   $$T(n) = 10 + \Theta(10 \log 10) + 10 + \Theta(20 \log 20) + \ldots + 10 + \Theta(n \log n)$$

   So, assuming we consider n as a multiple of 10, there are $n/10 * 10 = n$ constant terms, and there is the added term, so

   $$T(n) = n + \sum_{k=1}^{n/10} 10k \log(10k) = \Theta(n^2 \log n)$$

   This comes from the integral approximation

   $$\int_1^c x \ln(x)\, dx = \frac{1}{2}c^2 \ln(c) - \frac{c^2}{4} + D$$

   for some constant D. Averaging over n inserts, we get that each insert is roughly $\Theta(n \log n)$.

   For QUERY, we can now do a binary search, so a query will involve searching over the list U of 10 elements, and then binary searh over T, which has $\Theta(n)$ elements. The amortized complexity will be $\Theta(\log n)$ per QUERY.

   The worst case sequence of operations is n insertions, which will be $\Theta(n \log n)$ per operation, amortized.

(c) RESORT is called whenever the size of U reaches the size of S. **Solution:** In this case, note that the size of S doubles every so often. Let's consider a sequence of n inserts. Again, the total time is, for n a power of 2:

$$T(n) = 2 + \Theta(4\log 4) + 4 + \Theta(8\log 8) + \ldots + n/2 + \Theta(n\log n)$$

Grouping the constant terms and the other terms, we get

$$T(n) = n(1/2 + 1/4 + \ldots + \frac{1}{2^{\log_2(n)-1}}) + \sum_{k=2}^{\log_2(n)} \Theta(k * 2^k)$$

The first terms are in $\Theta(n)$, as the sum in parenthesis is bounded by 1. The second terms in the summation are in $\Theta(n\log n)$ (easy to see by integrating xe^x from 2 to $\log x$.) Hence, the amortized cost for each insertion averaged over n insertions is $\Theta(\log n)$.

QUERY now takes longer, because we have to search a U ArrayList of size $\Theta(n/2)$, so the search time will be $\Theta(n)$. The worst case sequence of n operations will be $n/2-1$ inserts, followed by $n/2+1$ queries, where n is a power of 2, so that the size of U is maximized. In this sequence, the average amortized complexity of each operation will be $\Theta(n)$.

2. (20 pts) Insert the following items into an empty B^+-tree where every node and leaf must have between 1 and 3 keys in it (so internal nodes have a minimum of 2 children and a maximum of 4 children):

   12; 10; 20; 9; 13; 18; 4; 2; 1; 15; 19; 21; 5; 8; 14; 17; 7; 11; 16;

   Draw the tree after the inserts of 18, 8, 16.

   Solution: Here is the full sequence of trees:

```
 10,12,20               12                        12
                    /       \                  /       \
                 9,10     12,20             9,10   12,13,20
```

```
             12 , 18
          /     |    \
       9,10  12,13   18,20
```

```
                    12 ,18             9,   12 ,   18                    9,   12 ,  18
                /     |    \        /     \    \      \                /      |   |     \
         4,9,10  12,13  18,20      2,4  9,10   12,13  18,20        1,2,4  9,10  12,13  18,20
```

```
                9,    12 ,   18                     9,    12 ,     18
            /     |       \      \                /     |      \      \
         1,2,4  9,10  12,13,15  18,20           1,2,4  9,10  12,13,15  18,19,20
```

```
                         18                                        18
                 /           \                             /            \
            9,   12             20                    4,   9,   12            20
          /   |      \        /    \                /   |   \     \        /   \
      1,2,4  9,10  12,13,15 18,19  20,21       1,2  4,5 9,10 12,13,15  18,19  20,21
```

```
                       18
             /                 \
      4,   9,    12             20
  /      |   \     \          /    \
1,2  4,5,8  9,10 12,13,15 18,19,  20,21
```

```
                 12,         18
        /               |        \
    4,    9             14          20
  /    |    \        /    \       /    \
1,2  4,5,8  9,10 12,13  14,15   18,19  20,21
```

```
           12,                  18
         /             |            \
     4,    9           14             20
   /    |   \        /    \          /   \
1,2  4,5,8  9,10 12,13 14,15,17   18,19  20,21
```

```
                      12,               18
             /                  |            \
     4,   7,  9                 14             20
   /   |   |   \             /    \          /     \
1,2  4,5  7,8  9,10       12,13 14,15,17   18,19  20,21
```

```
                12,                   18
             /               |              \
     4,  7,   9              14                20
   /    |  |     \         /    \            /    \
1,2  4,5  7,8  9,10,11    12,13 14,15,17   18,19  20,21
```

```
                   12,                    18
          /                        |           \
    4,  7,  9                   14 , 16         20
  /   |   |   \                /  \    \       /   \
1,2  4,5  7,8  9,10,11    12,13  14,15 16,17 18,19  20,21
```

3. (20 pts) Given input 4371, 1323, 6173, 4199, 4344, 9679, 1989 and a hash function $h_1(x) = x \mod 10$, and a second hash function $h_2(x) = (x/10) \mod 10$, show the following:

   (a) The result of inserting these into a hash table with 10 entries with hash function $h_1(x)$ using linear probing.

   (b) The result of inserting these into a cuckoo hash table with two tables of length 10, with hash functions $h_1(x), h_2(x)$ respectively.

   Solution:

   Part (a):

   ```
   9679
   4371
   1989
   1323
   6173
   4344
   xxxx
   xxxx
   xxxx
   4199
   ```

   Part (b):

   ```
   xxxx       xxxx
   4371       xxxx
   xxxx       1323
   6173       xxxx
   4344       xxxx
   xxxx       xxxx
   xxxx       xxxx
   xxxx       9679
   xxxx       xxxx
   1989       4199
   ```

4. (10 pts) Please answer the following questions:

(a) How many binomial trees will be part of a Binomial heap with 12 elements?

Solution: Expand 12 in binary, and it is 8 + 4, so there are only two trees: one of size 8 and one of size 4.

(b) Draw the three binary min-heaps that results from the following sequence of operations:

- 1) Insert 1,2,3,4,5,6,7,8,9,10,11,12. 2) Then delete 1. 3) Then, delete 2.

Solution: Since they are in order, the first 12 elements insert in order into a complete tree, as

```
           1
       /       \
      2         3
    /  \       /  \
  4     5     6    7
 / \   / \   /
8   9 10 11 12
```

To delete 1, we pop it, move last element to root, and downheap, resulting in

```
           2
       /       \
      4         3
    /  \       /  \
  8     5     6    7
 / \   / \
12  9 10 11
```

Similarly, we delete 2 and get:

```
           3
       /       \
      4         6
    /  \       /  \
  8     5    11    7
 / \   /
12  9 10
```

5. (15 pts) Draw the Binomial heaps that result at the end of each of the following sequence of operations:

- Insert 1,2,3,4,5,6,7,8,9,10,11,12 using non-lazy inserts (merging when necessary).
- Then delete 1 (note it is the minimum, so this is extract min)
- Then delete 2. (again, this is extract min)

Solution: See pictures below.

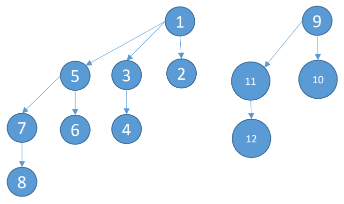

Figure 1: Solution for Problem 4.b

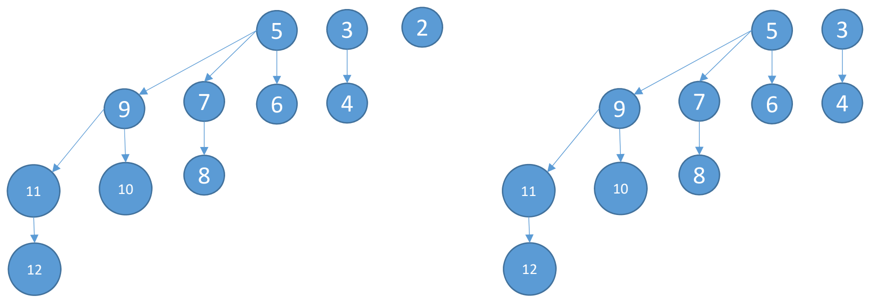

Figure 2: Solution for Problem 4.b, parts 2 and 3.