# Section 04: Heaps, LLRB Tree, Hashing

## Heaps

(a) Insert the following values into an initially empty binary max-heap: 10, 7, 3, 6, 15, 12, 20, 32

(b) How would you represent the heap above as an array?

(c) Insert the following values into an initially empty **trinary** max heap: 1, 10, 2, 15, 6, 8, 16, 17, 13, 9, 12, 18, 20, 4, 7, 3, 11, 19, 14, 5

(d) What would the heap from part c look like after calling removeMax() twice?

(e) Consider a binary heap implementation using an array with the root at index 1. What is the index of the parent of the element at index i? What is the index of its left and right children?

(f) Consider a binary heap implementation using an array with the root at index 1. Fill in the blanks for the insert method below:

```
int[] elements = ...;
int size = ...;
public static void insert(int num) {
  int k = size++;
  elements[k] = num;

  while (k > 1 && elements[_________] > elements[_________]) {
    // Swap the element with its parent
    int temp = elements[k];

    elements[k] = elements[_________];

    elements[_________] = temp;

    k = _________;
  }
}
```

# Big-$O$ BST, B-Tree, Left-Leaning Red-Black (LLRB) Tree

## 0.1. Runtime

Write down a tight big-$O$ for each of the following operation. Unless otherwise noted, give a bound in the worst case.

(a) Insert and find in a BST Insert : $O$(_____) Find : $O$(_____) One sentence Explanation:

(b) Insert and find in a 2-3 Tree Insert : $O$(_____) Find : $O$(_____) One sentence Explanation:

(c) Insert and find in a LLRB Tree Insert : $O$(_____) Find : $O$(_____) One sentence Explanation:

(d) Find the minimum value in a LLRB tree containing n elements $O$(_____) One sentence Explanation:

(e) Find the k-th largest item in a LLRB tree containing n elements $O$(_____) One sentence Explanation:

(f) List elements of LLRB tree in sorted otder $O$(_____) One sentence Explanation:

## 0.2. Dictionaries Comparison

(a) What are the similarities in the constraints on the data type you can store in a B-Tree and Red-Black Tree?

(b) When is using a Red-Black Tree preferred over a BST?

(c) When is using a BST preferred over a Red-Black Tree?

(d) When is a Red-Black tree preferred over another dictionary implementation, such as a HashMap?

## 0.3. Design Choice

(a) Imagine a database containing information about all trains leaving the Washington Union station on Monday. Each train is assigned a departure time, a destination, and a unique 8-digit train ID number. What data structures you would use to solve each of the following scenarios. Depending on scenario, you may need to either (a) use multiple data structures or (b) modify the implementation of some data structure. Justify your choice.

(b) Suppose the schedule contains 200 trains with 52 destinations. You want to easily list out the trains by destination.

(c) In the question above, trains were listed by destination. Now, trains with the same destination should further be sorted by departure time

(d) A train station wants to create a digital kiosk. The kiosk should be able to efficiently and frequently complete look-ups by train ID number so visitors can purchase tickets or track the location of a train. The kiosk should also be able to list out all the train IDs in ascending order, for visitors who do not know their train ID. Note that the database of trains is not updated often, so the removal and additions of new trains happen infrequently (aside from when first populating your chosen DS with trains).

# Hashing

## 0.4. Hash table insertion

For each problem, insert the given elements into the described hash table. Do not worry about resizing the internal array.

(a) Suppose we have a hash table that uses separate chaining and has an internal capacity of 12. Assume that each bucket is a linked list where new elements are added to the front of the list.

Insert the following elements in the EXACT order given using the hash function $h(x) = 4x$:

0, 4, 7, 1, 2, 3, 6, 11, 16

(b) Suppose we have a hash table that uses linear probing and has an internal capacity of 13.

Insert the following elements in the EXACT order given using the hash function $h(x) = 3x$:

2, 4, 6, 7, 15, 13, 19

(c) Suppose we have a hash table that uses quadratic probing and has an internal capacity of 10.

Insert the following elements in the EXACT order given using the hash function $h(x) = x$:

0, 1, 2, 5, 15, 25, 35

(d) Suppose we have a hash table implemented using separate chaining. This hash table has an internal capacity of 10. Its buckets are implemented using a linked list where new elements are appended to the end. Do not worry about resizing.

Show what this hash table internally looks like after inserting the following key-value pairs in the order given using the hash function $h(x) = x$:

$$(1, a), (4, b), (2, c), (17, d), (12, e), (9, e), (19, f), (4, g), (8, c), (12, f)$$

## 0.5. Evaluating hash functions

Consider the following scenarios.

(a) Suppose we have a hash table with an initial capacity of 12. We resize the hash table by doubling the capacity. Suppose we insert integer keys into this table using the hash function $h(x) = 4x$.

Why is this choice of hash function and initial capacity suboptimal? How can we fix it?

(b) Suppose we have a hash table with an initial capacity of 8 using quadratic probing. We resize the hash table by doubling the capacity.

Suppose we insert the integer keys $2^{20}, 2 \cdot 2^{20}, 3 \cdot 2^{20}, 4 \cdot 2^{20}, \ldots$ using the hash function $h(x) = x$.

Describe what the runtime of the dictionary operations will over time as you keep adding these keys to the table.