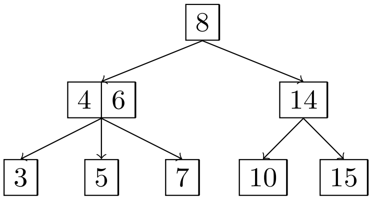
CS 61B | LLRBs, Hashing

Fall 2020 | Discussion 8: October 12, 2020

# 1 2-3 Trees and LLRB's

(a) Draw what the following 2-3 tree would look like after inserting 18, 38, 12, 13, and 20.



(b) Now, convert the resulting 2-3 tree to a left-leaning red-black tree.

(c) If a 2-3 tree has depth `H` (that is, the leaves are at distance `H` from the root), what is the maximum number of comparisons done in the corresponding red-black tree to find whether a certain key is present in the tree?

# 2 Hashing

(a) Here are three potential implementations of the `Integer`'s `hashCode()` function. Categorize each as either a valid or an invalid hash function. If it is invalid, explain why. If it is valid, point out a flaw or disadvantage.

```
public int hashCode() {
    return -1;
}
```

```
public int hashCode() {
    return intValue() * intValue();
}
```

```
public int hashCode() {
    return super.hashCode();
}
```

(b) For each of the following questions, answer **Always**, **Sometimes**, or **Never**.

1. When you modify a key that has been inserted into a `HashMap` will you be able to retrieve that entry again? Explain.

2. When you modify a value that has been inserted into a HashMap will you be able to retrieve that entry again? Explain.

## 3 Even More Asymptotics *Extra*

Give the runtime of the following functions in theta notation.

(a) $\Theta(\qquad)$

```java
public static void f1(int N) {
    for (int i = 2; i < N; i *= i) { }
    System.out.println("Hi");
}
```

(b) $\Theta(\qquad)$

```java
public static void f2(int N) {
    for (int i = 0; i < N; i++) {
        int jLimit = Math.pow(2, i + 1) - 1;
        for (int j = 0; j < jLimit; j += 2) {
            System.out.println("Hi");
        }
    }
}
```

(c) *This problem is really hard and not in scope but its fun.*
$\Theta(\qquad)$

```java
public static void f3(int N) {
    for (int i = 0; i < N * N; i++) {
        for (int j = 0; j < i; j *= 2) {
            System.out.println("Hi");
        }
    }
}
```