Week 14 - Wednesday

COMP 2100

Last time

- What did we talk about last time?
- Heaps
- Heapsort
- Timsort
- Sorting visualization

Questions?

Assignment 7

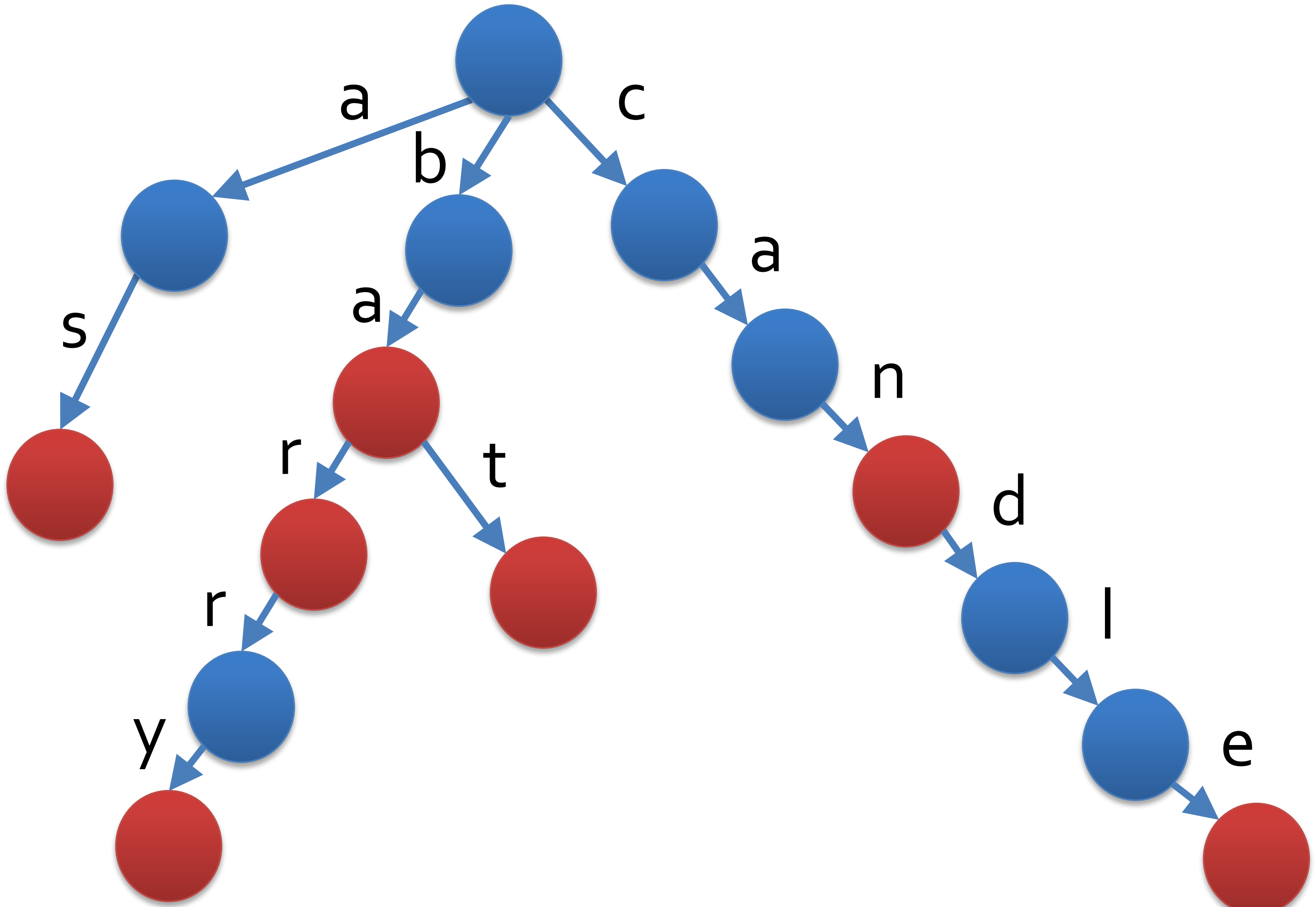
Project 4

Tries

Storing strings (of anything)

- We can use a (non-binary) tree to record strings implicitly where each link corresponds to the next letter in the string
- Let's store:
  - ba
  - bar
  - bat
  - barry
  - can
  - candle
  - as

Trie this on for size



Trie practice

- Now you add:
  - he
  - she
  - her
  - help
  - sat
  - rat

Trie implementation

```
public class Trie {
  private static class Node {
    public boolean terminal = false;
    public Node[] children = new Node[128];
  }

  private Node root = new Node();
}
```

Trie Contains

Signature for recursive method:

```
private static boolean contains(Node node, String
  word, int index)
```

Called by public proxy method:

```
public boolean contains(String word) {
  return contains(root, word, 0);
}
```

Trie Insert

Signature for recursive method:

```
private static void insert(Node node, String word,
  int index)
```

Called by public proxy method:

```
public void insert(String word) {
  insert(root, word, 0);
}
```

Trie Traversal

```
private static void inorder(Node node, String prefix)
```

Called by public proxy method:

```
public void inorder() {
  inorder(root, "");
}
```

Cost

- Let ***m*** be the length of a particular string
- Find Costs:
  - O(***m***)
- Insert Costs:
  - O(***m***)

Trie implementations

- Keeping an array of length equal to all possible characters (usually) wastes space
- Alternatives:
  - **Ternary search tries:** A lot like a binary search tree, with smaller characters to the left, larger characters to the right, and continuations from the current character beneath
  - Keeping an array (or linked list) of the characters used, resizing as needed

Ticket Out the Door

Upcoming

Next time…

- String searches

Reminders

- Work on Project 4
- Read 5.3