Matching Tags in Markup

→ this algorithm runs in O(n) time
→ make left → right pass thru raw string, using index j to track progress and the find method to locate '<' or '>' which define beginning & end of tags

```
def is_matched_html(raw):
    S = ArrayStack()
    j = raw.find('<')                    // find first < (if any)
    while j != -1:
        k = raw.find('>', j+1)           // find next >
        if k == -1:
            return False                 // invalid tag
        tag = raw[j+1:k]                 // strip away < >
        if not tag.startswith('/')       // opening tag
            S.push(tag)
        else:                            // closing tag
            if S.is_empty()
                return False             // nothing to match
            if tag[1:] != S.pop():
                return False             // mismatched delimiter
        j = raw.find('<', k+1)           // find next < if any
    return S.is_empty()                  // were all tags matched?
```

Queues

→ collection of objects inserted and removed according to FIFO principle
→ elements can be inserted anytime, but only element which has been in longest can be popped off
→ Queue ADT:
- Q.enqueue(e): add e to back of q
- Q.dequeue(): remove longest wait