Homework 4

David Trinh

October 14, 2024

- **Question 1**

```
class Node:
    private value
    private next

    constructor(value):
        this.value = value
        this.next = null

class LinkedList:
    private head
    private tail
    private length

    constructor():
        this.head = null
        this.tail = null
        this.size = 0

    public method length():
        return this.size

    public method isEmpty():
        if this.size == 0:
            return True
        else:
            return False

    // Insert at the tail of the list
    public method insert(value):
        node = new Node(value)
        if this.head is null do
            this.head = node
            this.tail = node
        else do
            this.tail.next = node
            this.tail = this.tail.next

        this.size = this.size + 1

    // Returns the first occurrence of the value
    public method find(value):
        if this.isEmpty():
```

```
        return null

    current = this.head
    while current is not null do
        if current.value == value do
            return current

        current = current.next

    return null

// Only deletes the first occurrence of the value
public method delete(value):
    if this.isEmpty() do
        return null

    current = this.head
    if this.length == 1 and current.value == value do
        this.head = null
        this.tail = null
        this.size = 0
        return current

    previous = null
    while current is not null do
        if current.value == value do
            previous.next = current.next
            this.length = this.length - 1
            return current

        previous = current
        current = current.next

    return null
```

- **Question 2**

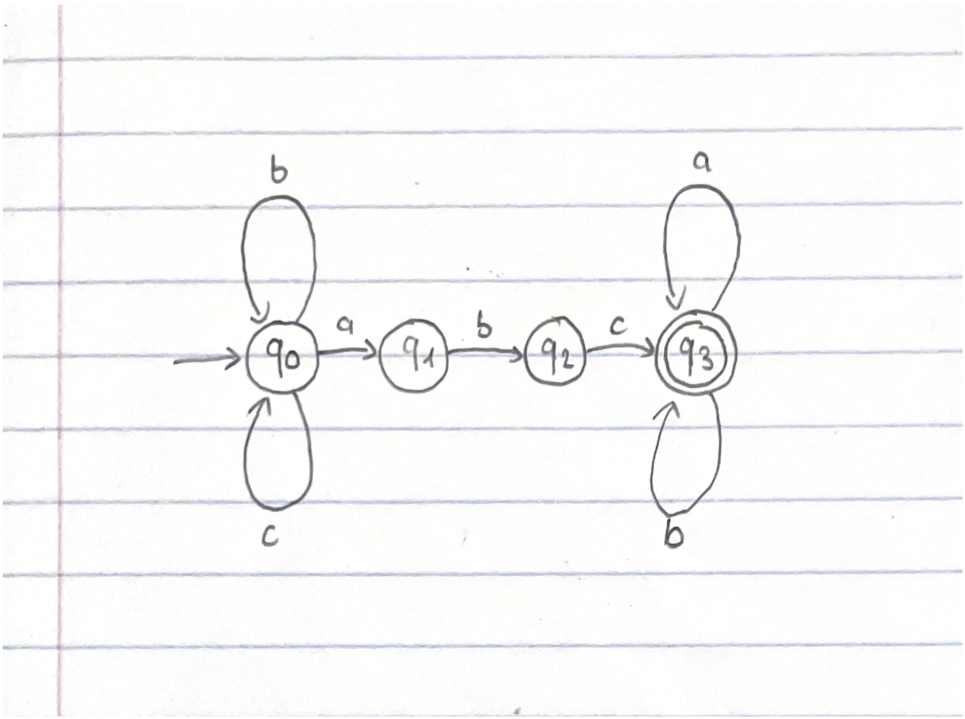
  Description of accepted strings:

  * Assume that any combination includes empty string
  * Starts with any combination of b's and c's
  * Has the definite string abc in the middle
  * Ends with any combination of a's and b's

  accept: bccabcbaab, abc

  rejected: cab, bcabcb

  $[bc] * abc[ab]*$



- **Question 3**

  The output is capped at 13 characters, including:

  - Capital letters A to Z: 26
  - Non-capital letters a to z: 26
  - Digits 0 to 9: 10

  An output of 13 characters, each with 62 possible alphanumeric character would yield $62^{13} = 2 \times 10^{23}$ unique shortened URLs. This is more than plenty. Assuming that the number of URLs in the world is less than 2×10^{23} (most likely), we can implement a hashing algorithm to generate unique shortened URLs.

  The only rule our hashing algorithm requires is that all shortened URL are unique, such that each will link to only one real URL. In that spirit, having a counter from "0000000000000" to "ZZZZZZZZZZZZZ" will work fine. This is because there isn't much of a security concern. However, it doesn't hurt to implement a hashing algorithm like SHA256 and take the first $6 * 13 = 78$ bits to generate a 13 character string, 6 bits for each alphanumeric character.

  Then, we can create a HashTable where the keys are the shortened string and the values are the real URL. Whenever a user search for tinyurl.com/xxxxxxxxxxxxx, the server will look at the value (real URL) assigned to the key xxxxxxxxxxxxx in the database HashTable and redirect them there.