Philippe Carrier 40153985

Assignment 3 – Programming questions

CVR class Variables:

threshold

numEntries

keyLength

sequence

tree

isBelowThreshold

**Algorithm** setThreshold(Threshold):

**Input** The value separating when to use sequence or a more complex ADT

class.threshold = Threshold

**if** numEntries < Threshold **then**

isBelowThreshold = true

**Algorithm** setKeyLength(length):

**Input** length between 10 and 17 inclusively of the VIN

keyLength = length

**Algorithm** generate(n):

**Input** the number of VIN to generate

**Output** The generate keys

keys = []

**for** i=1 **to** n **do**

key = new StringBuilder()

**for** y = 1 **to** keyLength **do**

key.append(randomLetterOrNumber)

keys.push(key)

**return** keys

**Algorithm** allKeys():

**Output** all the keys inside the data structure

String[] keys

**if** isBelowThreshold **then**

current = sequence.head

**while** (current != null) **do**

keys.append(current.key)

current = current.next

**else**

keys = tree.traversal()

**return** keys

**Algorithm** add(key, value):

**Input** key: the VIN of the car

**Input** value: information about the car

node = Node(key, value)

**if** isBelowThreshold **then**

sequence.add(node)

**else**

tree.add(node)

**Algorithm** getValue(key):

**Input** the VIN of a specific car

**Output** the values of the node

**if** isBelowThreshold **then**

**return** sequence.get(node).value

**else**

**return** tree.get(node).value

**Algorithm** nextKey(key):

**Input** the VIN of a specific car

**Output** the key of the next car VIN

**if** isBelowThreshold **then**

node = sequence.get(node)

**return** node.next(key)

**else**

node =tree.get(node)

**if** node.leftChild != null **then**

**return** node.leftChild.key

**else** **if** node.rightChild != null **then**

**return** node.rightChild.key

**else**

**return** null

**Algorithm** prevKey(key):

**Input** the VIN of a specific car

**Output** the key of the previous car VIN

**if** isBelowThreshold **then**

node = sequence.get(node)

**return** node.prev(key)

**else**

node =tree.get(node)

**if** node.parent != null **then**

**return** node.parent.key

**else**

**return** null

**Algorithm** prevAccids(key):

**Input** the VIN of a specific car

**Output** the accident of the car

node = getValue(key)

**return** node.value

3. Discuss the difference in complexity between linked list and array.

When below the threshold, in the sequence part, the time complexity is fairly similar. The difference comes from the space complexity. When using linked list, the node is taking more space as more information has to be save within the node (next node, previous node, etc.) which is not something that an array base sequence needs to worry about since the use a pointer to point toward the last entry. Another aspect of the space complexity is the size of the array. If the array size is fixed, the amount of space it takes in memory is bigger than a linked list in the beginning since there is no need to fix the size of an array as it grows with its data.

As for when using a tree, the linked list is more efficient when moving through the data structure since there is no need to calculate each time the position of the child of the array. Once again, the array size has to be fixed at the beginning to allow the data structure to work properly or it could be fixed dynamically but the insertion method time complexity would go from O(log n) to O(n) which is not a desirable behavior when there are a lot of nodes in the tree.

a) Design decisions and specification:

For my ADT, I decide to use a linked list sequence for the data structure of the data when the amount of information is below the given threshold. The reason I used a linked list is that it provides an easy way to select the previous and next value without having to save a pointer. Also, when removing values from my sequence, there is no need to fill the place left by the removed value like in it would be needed if done using an array. That way, removing a value from my sequence would be faster than when using an array. The drawback of using a linked list is that that node takes more memory than using an array. However, I don’t believe that it will have a significant impact on the program.