Q1.

setThreshold(int threshold, String fis)

BufferedReader br <- null, String s, String lineContent

Br <- new BufferedReader(fis) throws FileNotFoundException

```
If threshold > 500000
    registry <- new CargistryLL
    while lineContent != null
        do lineContent <- readLine
        if lineContent != null
            registry add(lineContent, Car)
        end if
    end while
else if threshold > 100
    registry <- new AVLTree
    while lineContent != null
        do lineContent <- readLine
        if lineContent != null
            registry add(lineContent, Car)
        end if
    end while
else
    registry <- new CargistryLL
    while lineContent != null
        do lineContent <- readLine
        if lineContent != null
            registry add(lineContent, Car)
        end if
    end while
end
```

```
setKeyLength(int length)

if length > 12 OR length < 6
        return -1
end if
return length

generate(int length)
        String newKey
        String randomChar

        for i<-0, i < length, i++ do
                randomChar <- GenerateRandomChar()
                newKey atChar(i) <- randomChar
        end for

        while checkDuplicate(newKey)
                generate(length)
        end while

end


allKeys()

String[] sortedSequence
int index <- 0
Node tempd <- head

If size EQUALS 0
        return sortedSequence
end if

While temp != tail do
        sortedSequence[index] <- temp.getValue
        temp <- temp.next
        index++;
end while

Return sortedSequence
end
```

```
add(String key)

if head EQUALS null
    Node temp <- new Node
    head <- temp
    head.next <- head
    head.prev <- head
    tail <- head
else
    Node temp <- new Node
    head.prev <- temp
    tail.next <- head
    tail <- head
    temp <- null
end if
size++
end

remove(String key)

int index <- 0
Node temp <- head

checkKeyLength()

while key.compareTo(key.getCar.getKey) != 0 do
    temp <- temp.next
    index++
end while

if index EQUALS 0 do
    removeHead()
else if index EQUALS size-1 do
    removeTail()
else
    removeNode()
end if
end

getValues(key)
    getADT.getValue EQUALTO key
end

nextKey(key)
    getADT.getValue EQUALTO key.next
end
```

```
prevKey(key)
    getADT.getValue EQUALTO key.prev
end

previousCars(key)

ArrayList<Car> duplicatedCar
int index <- 0
Node temp <- head

If size EQUALS 0 do
    return
end if

While temp != tail do
    if key.compareTo(temp.getCar.getKey) EQUALS 0 do
        duplicatedCar.add(temp.getCar)
    end if
    temp <- temp.next
    index++
end while
return duplicatedCar

End
```

Q3

The space complexity for a threshold of <100 and >500’000 is O(n) growing linearly.

If the amount of data is >100 and <=500’000, we would use AVLTree and the space would be O(n) while the runtime would be O(logn)