Group Members

Kelvin Wilch

Eliot, Cole

Bill Ngoga

Candida, Gabriella

Part 05 Write for the RaggedArrayList.

Output

testing routine for RaggedArrayList

insertion order: qwertyuiopasdfghjklzxcvbnmaeiou

The number of comparison to build the RaggedArrayList = 169

TEST: after adds - data structure dump

[0] -> [a][a][b][c][ ][ ][ ][ ]

[1] -> [d][e][e][f][ ][ ][ ][ ]

[2] -> [g][h][i][i][j][ ][ ][ ]

[3] -> [k][l][m][n][ ][ ][ ][ ]

[4] -> [o][o][p][q][ ][ ][ ][ ]

[5] -> [r][s][t][u][u][v][ ][ ]

[6] -> [w][x][y][z][ ][ ][ ][ ]

[7] -> null

STATS:

list size N = 31

square root of N = 5.568

level 1 array 7 of 8 used.

level 2 array sizes: min = 4 used, avg = 4.4 used, max = 6 used

Successful search: min cmps = 4, avg cmps = 8.7, max cmps 14

TEST: contains("c") ->true

TEST: contains("7") ->false

TEST: toArray

[a][a][b][c][d][e][e][f][g][h][i][i][j][k][l][m][n][o][o][p][q][r][s][t][u][u][v][w][x][y][z]

TEST: iterator

[a][a][b][c][d][e][e][f][g][h][i][i][j][k][l][m][n][o][o][p][q][r][s][t][u][u][v][w][x][y][z]

TEST: sublist(e,o)

[0] -> [e][e][ ][ ]

[1] -> [f][g][ ][ ]

[2] -> [h][i][ ][ ]

[3] -> [i][j][k][l][m][n][ ][ ]

[4] -> null

[5] -> null

[6] -> null

[7] -> null

The worst-case times for each operation on a RaggedArrayList with N item:

1.**contains():**

This method has to part where the first one the findfront method should search for the specified item and return its location which is done in O(N).After is check if the item at that location contains the target, this comparison could be done in constant time but since we consider one that dominants contain() has a complexity if O(N)

2.**Iterating through the whole list:**

The worst-case for iteration through the entire list will be O(N)complexity.This is because iterating through the entire list involves accessing each element once and there are N elements.

3**. toArray():**

The toArray() method checks the array size which is done in a constant time, and later transverse the list using the iterator n times. This operation happens in linear time O(N). Therefore, the dominant factor in the worst-case time complexity is the iteration through the list, making the overall complexity O(N).

4**. subList():**

The subList(E fromElement, E toElement) method has a worst-case time complexity of O(N), Because This complexity arises from the method's findFront() & findEnd()which efficiently locate the starting and ending positions of the sublist in O(N) time and add the element.