**Assignment 8**

Consider the multi-way search tree of Lecture 8, slide 39. Why isn’t it a valid (2,4) tree? Justify your answer. What could we do to make it into a valid (2,4) tree? Draw the valid (2,4) tree.

A picture containing line, diagram, circle, drawing

Description automatically generated

Answer:

The multi-way search tree above not follows the Depth Property: all the external nodes have the same depth, so it isn’t a valid (2,4) tree.

Suppose the last key inserted into the tree is 30, we put 30 between 27 and 32 then it is a valid (2,4) tree.

A picture containing circle, diagram, line

Description automatically generated

R-3.10 A certain Professor Amongus claims that a (2,4) tree storing a set of items will always have the same structure, regardless of the order in which the items are inserted. Show that Professor Amongus is wrong.

1. Insert the following sequence of keys into an initially empty 2-4 tree in this order:

(16, 5, 22, 45, 2, 10, 18, 30, 50, 12, 1, 33)

Use the following URL to check at each insertion in A above. http://cs.armstrong.edu/liang/animation/web/24Tree.html

A picture containing circle, diagram, line, sketch

Description automatically generated

Insert the following sequence of keys into an initially empty 2-4 tree in this order:

(1, 33, 16, 5, 22, 45, 2, 10, 18, 30, 50, 12)

A picture containing circle, diagram, line, sketch

Description automatically generated

This result says a (2,4) tree storing a set of items will have the structure depend on the order in which the items are inserted. So, Professor Amongus was wrong.

C-4.11 Suppose we are given an *n*-element sequence S such that each element in S represents a different vote in an election, where each vote is given as an integer representing the ID of the chosen candidate. Suppose we know who the candidates are and the number of candidates running is *k* < *n*. Describe an *O(n* log *k)*-time algorithm for determining who wins the election.

Answer:

Algorithm **findWinner**(S)

if S.isEmpty() then

throw EmptySequenceException

D 🡨 new Dictionary(HashTable)

p 🡨 S.first()

max 🡨 1

D.insertItem(p.element(), max)

while !S.islast(p) do

p 🡨 S.after(p)

id 🡨 p.element()

cnt 🡨 D.findElement(id)

if cnt = NO\_SUCH\_KEY then

cnt 🡨 0

cnt = cnt + 1

D.insertItem(id, cnt)

if max < cnt then

max 🡨 cnt

results 🡨 new List

for each item of D.items()

if item.value() = max then

results.insertLast(item)

return (max, results)

C-4-22 Let A and B be two sequences of n integers each. Given an integer x, describe an O(n log n)-time pseudo code algorithm for determining if there is an integer *a* in A and an integer *b* in B such that x = *a* + *b*.

Answer:

Algorithm **determiningSum**(A, B, x) O(n log n)

if A.isEmpty() or B.isEmpty() then 1

return F 1

lo 🡨 A.first() 1

hi 🡨 A.last() 1

**quickSort**(A, A.rankOf(lo), A.rankOf(hi)) n log n

lo 🡨 A.first() 1

hi 🡨 A.last() 1

for each b in B.elements() do n

a 🡨 **binarySearch**(A, (x-b), A.rankOf(l), A.rankOf(h)) log n

if a != NO\_SUCH\_VALUE then n

return T n

return F 1

Algorithm **quickSort** (S, lo, hi)

if lo < hi then

p 🡨 **partition**(S, lo, hi)

**quickSort** (S, l, p-1)

**quickSort** (S, p+1, hi)

Algorithm **partition** (S, lo, hi)

p 🡨 random(lo, hi)

S.swapElements(S.atRank(lo), S.atRank(p))

pi 🡨 S.atRank(lo).element()

j 🡨 lo+1

k 🡨 hi

while j <= k do {

while k >= j and S.atRank(k).element() >= pi do

k--

while j <= k and S.atRank(j).element() <= pi do

j++

if j < k then

S.swapElements(S.atRank(j), S.atRank(k))

}

S.swapElements(S.atRank(lo), S.atRank(k))

return k

Algorithm **binarySearch**(S, x, lo, hi)

mid 🡨 floor((lo+hi)/2)

p🡨S.atRank(mid)

if p.element() = x then

return p

else if lo >= hi then

return NO\_SUCH\_VALUE

else if p.element() < x then

return **binarySearch**(S, x, lo, mid)

else

return **binarySearch**(S, x, mid+1, hi)