COMP2054 2023-24 ADE Coursework THREE (12.5%)
Mon. 13-MAY-2024

Time: 30 minutes.

Do not turn over page until instructed.
Answer ALL questions for a potential total of 25 marks.

(Standard policy) If you think something in a question is incorrect, then please just answer the question as it is – but write a short note about anything you think is wrong. The test is distributed over multiple rooms and times and so it is not possible to do corrections "live". If there an error, then this will be taken account of during the marking.

PARTIAL ANSWERS

Question 1. Hashmaps using linear probing [9 marks]

a) Consider a hash table of size N= 7, with the (very simple) hash function

h(k) = (k+4) mod 7

and using **linear probing.**

Starting from an empty hash table give the results after each stage of the following sequence of 3 insertions.

You can use that: **h(2)=6**, **h(3)=0**, and **h(9)=6**

Show the result after doing insert(2), after starting from an empty table:

Index	0	1	2	3	4	5	6
Entry							2

.. and then, insert(3) gives

Index	0	1	2	3	4	5	6
Entry	3						2

.. and then, insert(9) gives

Index	0	1	2	3	4	5	6
Entry	3	9					2

Note that you do not need to write justifications, just add entries to the tables.

Question 1. (cont)

b) Using the same hash table and function as in Q1a, and again using linear probing a sequence of operations resulted in the following state of the table:

Index	0	1	2	3	4	5	6
Entry	~~5~~ **(should be 1)**					8	2

You can use that: **h(8)=5**, **h(2)=6**, and **h(1)=5**

NOTE: The '5' in index 0 was a typo, and it should have been '1'. This was the natural interpretation: note the given data that "h(1)=5" which meant insertion of a '1' would first have tried index=5, but linear probing must have found the index 5 and 6 already occupied and then so been pushed into index=0.
Note that h(5) = (5+4) mod 7 = 2, and so a (less likely) interpretation was that it did mean to have a 5 in the table, but the error was that it should be in index=2.

In the marking both possibilities were allowed in the answer, and so marks would not suffered.

The task is then to do remove(8).
State the problem that will arise if the entry for 8 is simply set to be blank (empty):

With the '5' in index 0 which was a typo for '1'
We will lose the entry at index 0 .

Note that entry=2 is at its "home index" of 6, and so would not be affected.

Briefly describe ONE scheme for handling removals when using linear probing:

EITHER
Mark the deleted cell in some way (e.g. -1, DEL, D etc.), then clean up such cells at a later stage – this can be called "lazy deletion".

OR
Reinsert all items to the right of the removed one

Show the final result of the table after using the scheme you suggested,

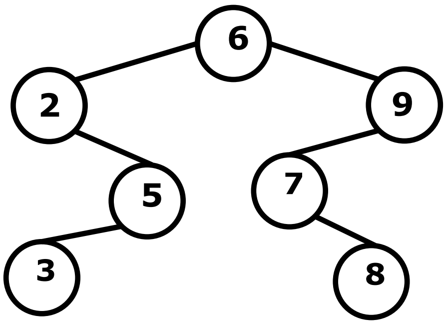
Lazy deletion:

Index	0	1	2	3	4	5	6
Entry	**5 (1)**					**D**	**2**

Reinsertion:

Index	0	1	2	3	4	5	6
Entry			***5 here was also allowed***			**5 (1)**	**2**

Question 2. Binary Search Trees (BSTs) [9 marks]



Consider the BST above: You are to **delete(6)** (i.e. remove 6 from the set of keys) using the method as given in the lectures.

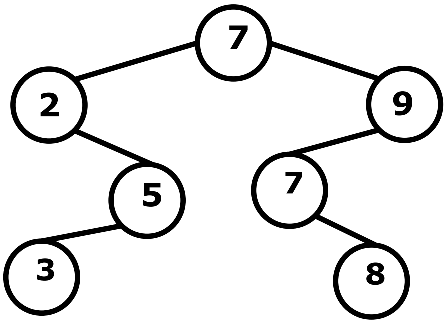
Show and **briefly explain** the process to delete(6), and give the final BST that results after the deletion:

The first step is to identify either the 'previous node to 6' or else the 'next node from 6' in an in-order traversal, and at will be used to replace 6.
The process is same for each, so we just show using the 'next node'.

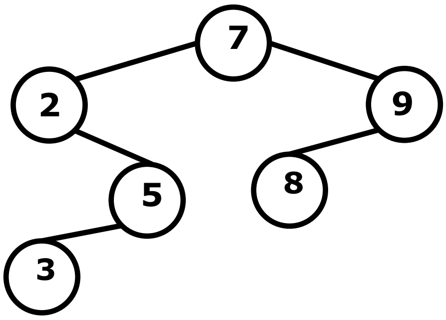
The "next node from 6" is '7'.

So we overwrite 6 with 7.

This gives an intermediate stage:



And now we need to delete the old "7".
In this case it is straightforward and it is just replaced by it right child, node '8', to give



TURN OVER FOR Q3 !

Question 3. Change giving [7 marks]

Consider a **set of coins {2,5,4} and a target of change of 6.**
Only ONE of each coin in the set is available.
Consider the algorithm of the lectures based on an array A, with meaning:

A[i]=T iff a change of i is possible using the coins considered so far

Complete the following tables to show the state of the array after every iteration. In the arrays, leave the entry as blank to signify “F” for false, and just add the “T” for true – meaning that the sum given by the index (given in the top row) is achievable using the coins so far.

- *The progress of the array is given below.*
- *It was also allowed, but unnecessary, to do the “min-coins” version in which case the entries are numbers.*

Using the starting array:

0	1	2	3	4	5	6
T						

Complete the array after adding coin ‘2’:

0	1	2	3	4	5	6
T		**T**				

Complete the array after also adding coin ‘5’:

0	1	2	3	4	5	6
T		**T**			**T**	

Complete the array after also adding coin ‘4’:

0	1	2	3	4	5	6
T		**T**		**Can be either T or empty**	**T**	**T**

The option in index 4, just arises from whether or not the code stops in find the T at index=6

From your final array:

Must be correct from your final array, and not just from "knowing the answer"

Is it possible to give a change of 6? Yes ~~/ No~~ (circle one)

Is it possible to give a change of 3? ~~Yes /~~ No (circle one)