Home work 3

Vasanth Reddy Baddam

02/20/2020

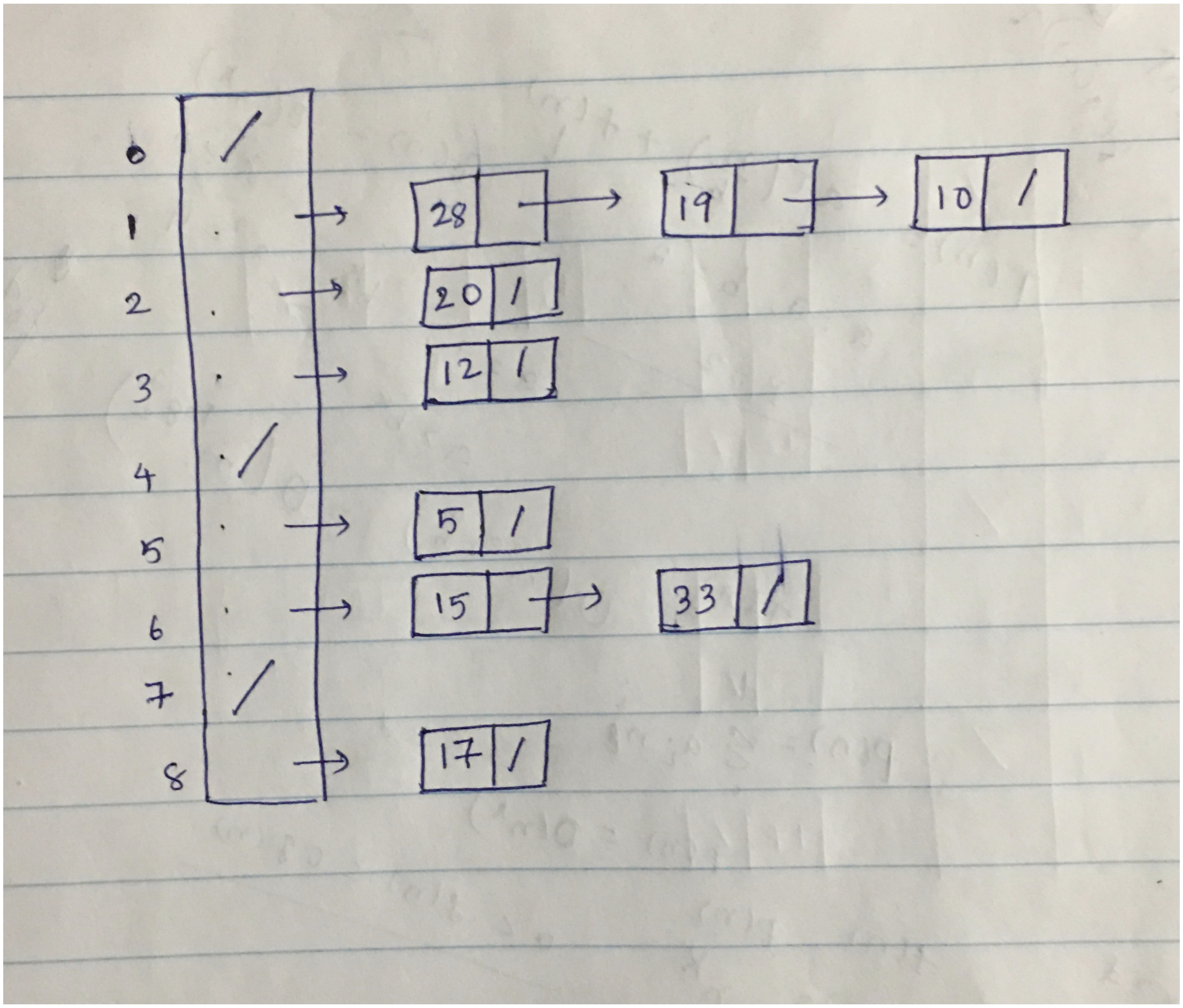
I pledge that this test/assignment has been completed in compliance with the Graduate Honor Code and that I have neither given nor received any unauthorized aid on this test/assignment
Name: Vasanth Reddy Baddam
Signature: VB

Q1.

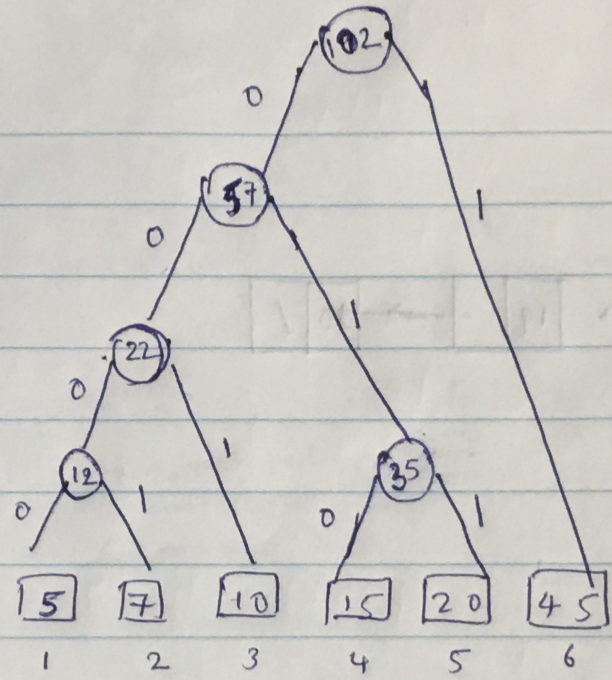
Given that m = 1000, A = $(\sqrt{2} - 1)/2$
$A = (\sqrt{2} - 1)/2 = 0.6180339887....$

k	kA	kA mod 1	m(kA mod 1)	h[k] = *m(kA mod 1)
32	19.77708764	0.77708764	777.08764	777
45	27.81152949	0.81152949	811.52949	811
56	34.60990337	0.60990337	609.90337	609
62	38.31810730	0.31810730	318.10730	318
78	48.20665112	0.20665112	206.65112	206
90	55.62305899	0.62305899	623.05899	623

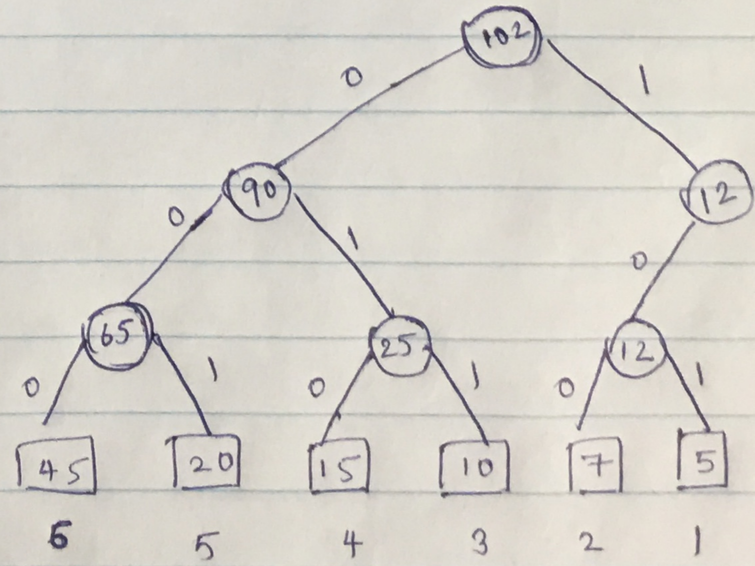
Q4.
$h(k)$ values for the given are: 5, 1, 1, 6, 2, 6, 3, 8, 1
keys 28, 19, and 10 have the same hash function values 1



Q3.



— variable length —



— fixed length —

	1	2	3	4	5	6	Total bits
Frequency	5	7	10	15	20	45	N/A
Variable	0000	0001	001	010	011	1	228
Fixed	101	100	011	010	001	000	306

Q2.

Algorithm 1 Tree Predecessor

```
procedure TREE-PREDECESSOR(x)
    if x.left ≠ NIL then
        TREE-MAXIMUM(x.left)
    y = x.p
    while y ≠ NIL and x == y.left do
        x = y
        y = y.p
    return(y)
```

Q5.

a).

We have k arrays each having length of 2^i. where i is the ith array.

For array of length n, running time to search using binary search is $O(\lg n)$. Worst case would be searching every given array. That is summation of Big O over all the arrays.

The worst running time:

$$\begin{aligned} &\sum_{k=1}^{\lg n} \lg 2^k \\ =&\sum_{k=1}^{\lg n} k \\ =&\frac{(1+\lg n) \times \lg n}{2} \\ =&\frac{\lg^2 n}{2} + \frac{\lg n}{2} \end{aligned} \tag{1}$$

So the worst-case running time is $O(\lg^2 n)$.

b).

Let's say that the element is added in an array. That length will be 1. Then adding this array to A_0,

then updating the array. Then the obtained array will be merged with A_1 and then later updated. This process goes on until all the arrays merges and get updated. This results in k merges.
Worst running would result when all the arrays merge.
Worst running time is $O(2^k)$ as there are k merges.

$$O(2^k) = O(2^{log(n+1)})$$
$$= O(n+1)$$

Worst running time is $O(n)$