COP 3503 4/21/22

Probs from Review Qs

Question: 2, 3, 4, 6, ~~8~~, 9, 10, 12

$c_1 = r_2$

mult $r_1 \times c_1$ $r_2 \times c_2$

#mult = $\boxed{r_1 \times c_1 \times c_2}$

2) $A = 2 \times 5$, $B = 5 \times 4$, $C = 4 \times 1$, $D = 1 \times 5$

	A	B	C	D
A	0	40	30	40
B		0	20	45
C			0	20
D				0

$AB = 2 \times 5 \times 4 = 40$

$BC = 5 \times 4 \times 1 = 20$

$CD = 4 \times 1 \times 5 = 20$

$ABC = (AB)C = 40 + 2 \times 4 \times 1 = 48$ (AB 2×4, C 4×1)

$A(BC) = 20 + 2 \times 5 \times 1 = \boxed{30}$ (A 2×5, BC 5×1)

$BCD: (BC)D = 20 + 5 \times 1 \times 5 = \boxed{45}$ (BC 5×1, D 1×5)

$B(CD) = 20 + 5 \times 4 \times 5 = 120$ (B 5×4 CD 4×5)

$ABCD: (ABC)D = 30 + 2 \times 1 \times 5 = \boxed{40}$ (ABC 2×1, D 1×5)

$(AB)(CD) = 40 + 20 + ? > 40$ (can't be best)

$A(BCD) = 45 + ?? > 40$ (can't be best)

4) $817 - 1 = 816 = 2^4 \times 51$

$\boxed{51,102,204,408}$ $\boxed{816}$

2 | 816

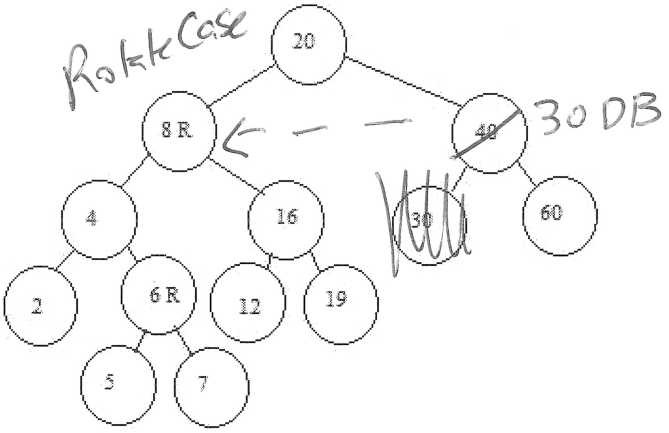
2 | 408

2 | 204

2 | 102

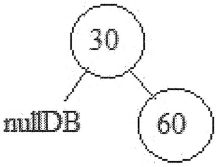
51

4) (10 pts) Draw the result of deleting the value 40 from the red-black tree shown below. In the drawing below, red nodes are indicated with a letter 'R' next to the number stored in the node. In your solution, please put an 'R' in each node that is red. (Note: use the regular binary tree rules for deleting a value which is stored in a node with 2 children.) If you explain your thinking, you may get partial credit for incorrect solutions.

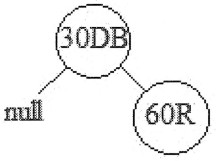


Note: When deleting 40, we can either delete the physical location of node 30 or 60. Due to the length of the solution, this solution only shows the result of deleting the physical location of the 30. A second correct answer exists (and was properly graded for) where the physical node storing 60 is deleted.

After we denote the physical node storing 30 to be deleted, we copy 30 into the node that originally stored 40. Here is the state of affairs from that point, just for the subtree rooted at 30:

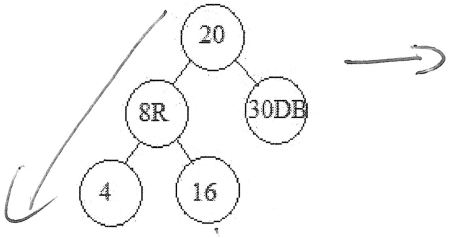


Initially, the null node is a double black.

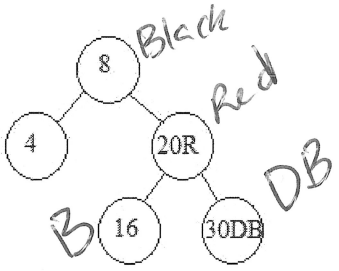


We push the double black up one level.

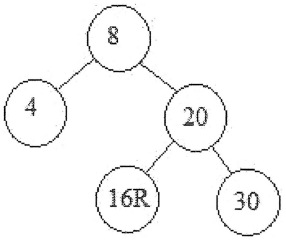
Then, we deal with a double black that has a red sibling:



This is the original picture.

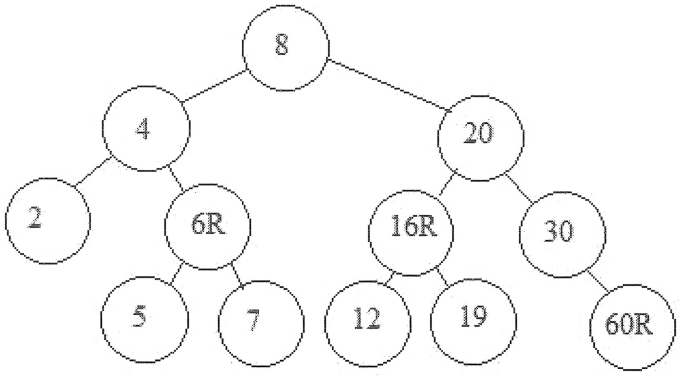


Here is when we right rotate making 8 the root and 20 red.



Push the double black up one level, fixing the issue.

Here is the final tree after those changes are made:



Note: if we had deleted the physical node storing 60 instead of the one storing 30, the only difference in the final answer is that 60 would be a black child of 20 and 30 would be a red left child of 60.

Grading: Give full credit if final tree is correct. If incorrect, give partial as follows:

1 pt - deleting physical node of 30 or 60.
2 pts - db null node
2 pts - pushing up db to 30 or 60.
3 pts - restructuring with 8 at root, 4 to left and 20 to right
2 pts - push up double black

6)
```
public static int lis(int[] seq) {
    int[] sorted = copyOf(seq, seq.length);
    Arrays.sort(sorted);
    return lcs(seq, sorted);
}
```

Seque... Problem Reduction Example

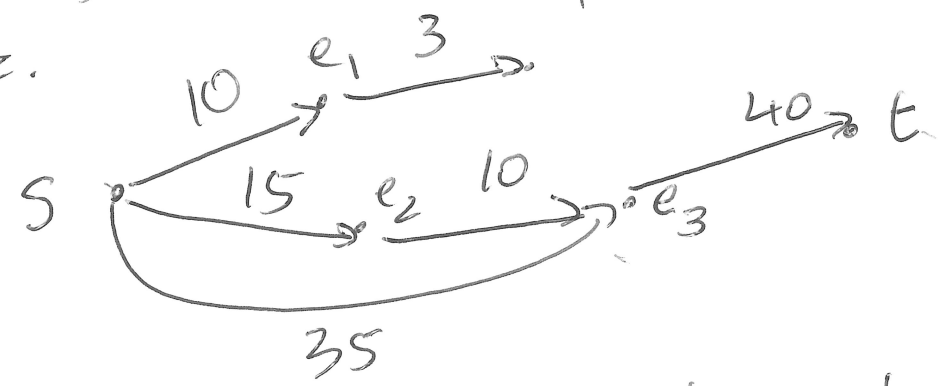
Room Scheduling - single room
max time it's used
requested

	s_1	e_1	
e_1	10	40	✓
e_2	15	25	
e_3	35	60	
e_4	43	47	✓
e_5	50	70	✓
etc.			

Can schedule 2 events

$e_i \leq s_j$

100



for all path from s→t distance = time unused

run Dijkstra's get shortest path!

12) Define dp[i] = probability i gets it to player n-1.

```
dp[n-1] = 1;
for (i=n-2; i>=0; i--) {
    double tmp = prob[i][i+1] * dp[i+1];
    dp[i]
    for (int j = i+2; j<n; j++)
        dp[i] = max(dp[i], prob[i][j] *
                dp[j]);
}
return dp[0];
```

(pty i gets i+1; i+1 gets to end; pass from i to j; j succeeds)

$$T(n) = 1 + \sum_{i=0}^{n} \binom{n}{i} \left(\frac{1}{k}\right)^i \left(\frac{k-1}{k}\right)^{n-i} T(n-i)$$

$$T(n) = 1 + \left(\sum_{i=1}^{n} \binom{n}{i} \left(\frac{1}{k}\right)^i \left(\frac{k-1}{k}\right)^{n-i} T(n-i) \right) + \left(\frac{k-1}{k}\right)^n T(n)$$

$$T(n)\left(1 - \left(\frac{k-1}{k}\right)^n\right) = 1 + \sum_{i=0}^{n-1} \binom{n}{i} \left(\frac{1}{k}\right)^i \left(\frac{k-1}{k}\right)^{n-i} T(n-i)$$

(2) DIVIDE BY THIS!

(1) SOLVE THIS