Fall 2012: CS 3323 Final Exam

Total Time: 90 minutes Total Points: 60

Write your name clearly. Answer all the questions.

Reead all the questions. If you need more space, use the other side of your question sheet.

Name:_____ **Date:**_____

****

Question 1. True/False **[10]**

T 1. Queue ensures that the items are processed in the order they are received.
F 2. Item insertion and deletion in a linked list requires significant data movement.
T 3. The shortest path is the path with the smallest weight.
T 4. The breadth first traversal traverses the graph from each vertex that is not visited.
T 5. Performance of search in an ordered Tree depends on the shape of the Tree.
T 6. The value of null pointer is zero.
T 7. Insertion in Red-Black Tree will give black height imbalance problem but not double red.
T 8. After inserting the node in an AVL tree, the reconstruction can occur at any node on the path back to the root node.
T 9. Hash table mapping scheme must be able to minimize collisions.
T 10. When data is being organized, a programmer's highest priority is to organize it in such a way that item insertion, deletion, and lookups (searches) are fast.

Question 2. Code Analysis **[5+5]**

(a) For next set of problems, assume that `q` is a queue implemented by using circular arrays with **`QueueElement = char`** and **`capacity = 5`.**

Show the value of `myFront` and `myBack` and the contents of `myArray` for the Queue object q after the program segment has been executed; also indicate any errors that occur.

`q.enqueue('A');`
`q.enqueue('B');`
`q.enqueue('C');`
`ch= q.front();`
`q.dequeue();`
`q.enqueue(ch);`
	`myFront =` B `myBack =` A
	`myArray:`
	[0]='B' [2]='A'
	[1]='C'

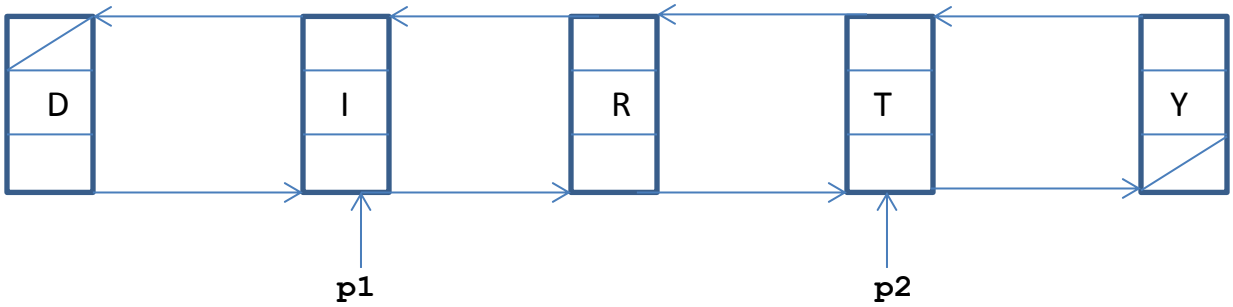
```
q.enqueue('X');
q.enqueue('Y');
q.enqueue('Z');
while(!q.empty()){
ch = q.front();
q.dequeue();
}
```

myFront = myBack =

myArray:

ERROR

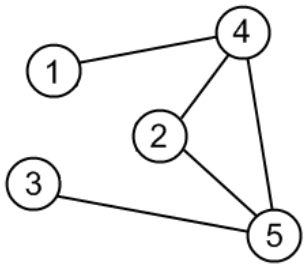
(b) Assume the following doubly linked list with the two pointers p1 and p2:



Find the value of each expression.

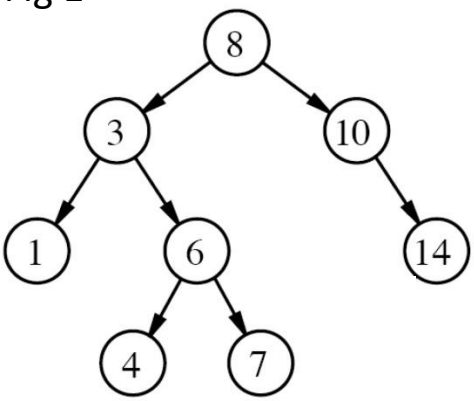
(i) P1->next->data R
(ii) P1->next->next T
(iii) P1->prev->next
(iv) P2->prev->prev->next->data
(v) P2->prev->prev->prev->prev

Question 3. Give the sequence of nodes visited in (a) Depth First and (b)Breadth First Traversals on graph shown in figure below. Starting point is node 1 [4]



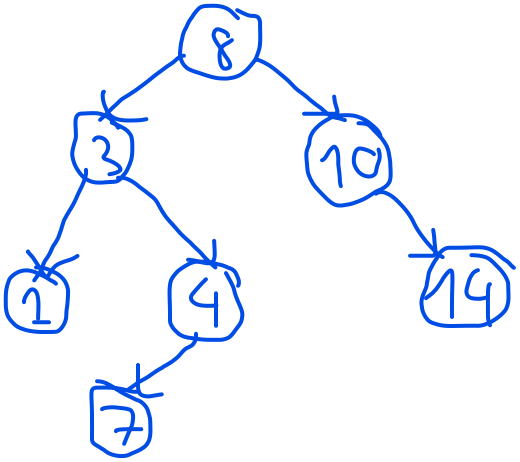
Question 4. Draw the resulting AVL tree, in each sequence of following operations such that resulting tree follows AVL property: **[8]**

Fig 1

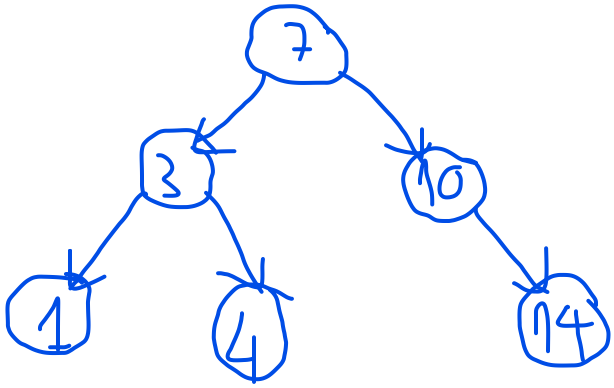


After:

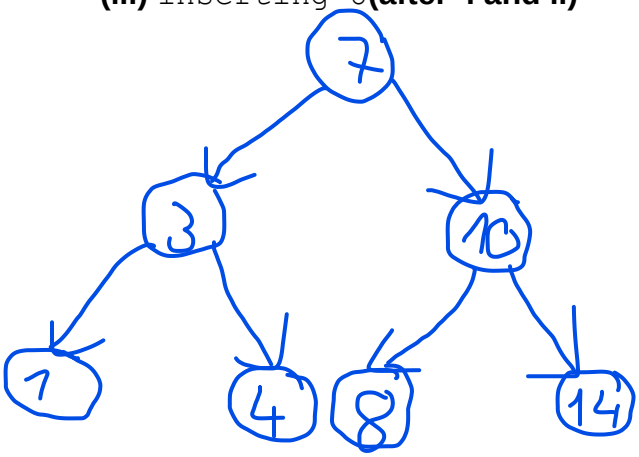
(i). `Deleting 6`



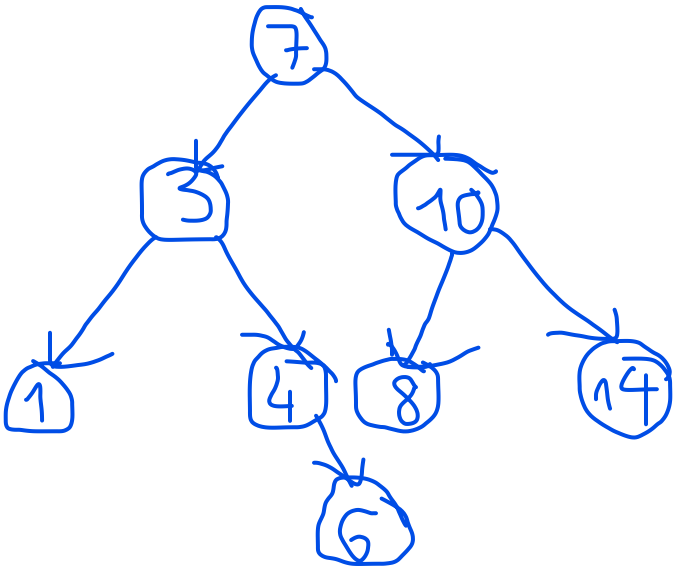
(ii) `Deleting 8` (after i)



(iii) `inserting 8`(after i and ii)



(IV) `inserting 6` (after i, ii and iii)

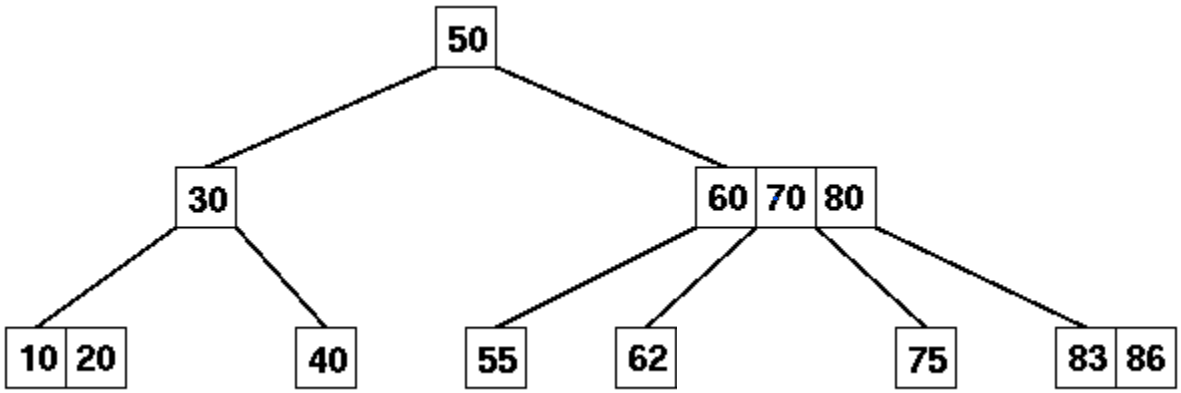


Question 5. Provide a suitable ADT for Red-Black Tree in c++ with a suitable node structure (with a default and overloaded constructor methods) to represent this tree. [8]

Note: You should just provide the definition of the ADT, you do not have to provide the implementation of any methods except the constructor for the node data structure.

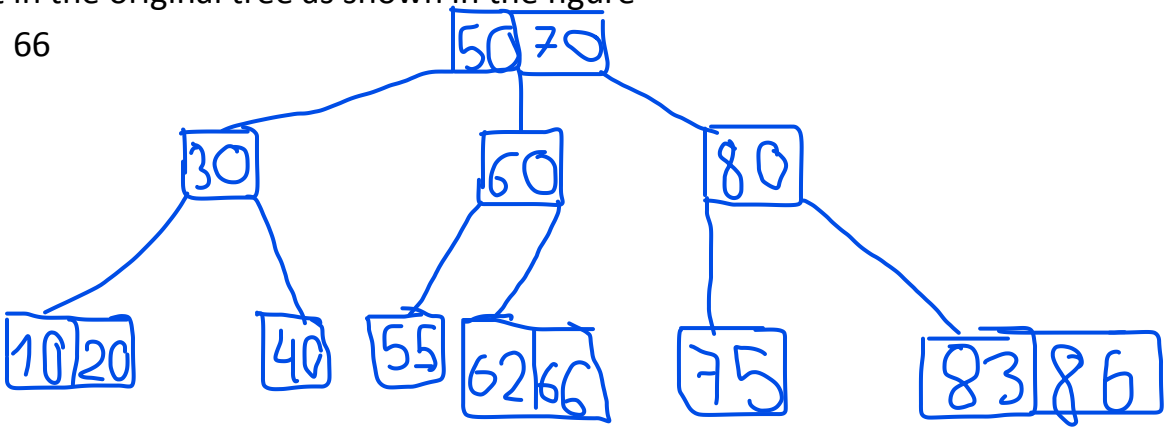
Question 6. Describe how would you design a stack using two queues. In particular, how would you perform operation for `empty`, `top`, `push` and `pop` by using stack operations (such as `front`, `enqueue`, `dequeue` and `empty`) on these two stack. [1+1+3+3]

Question 7. In 2-3-4 tree below, show the tree after each of the following operations [6]



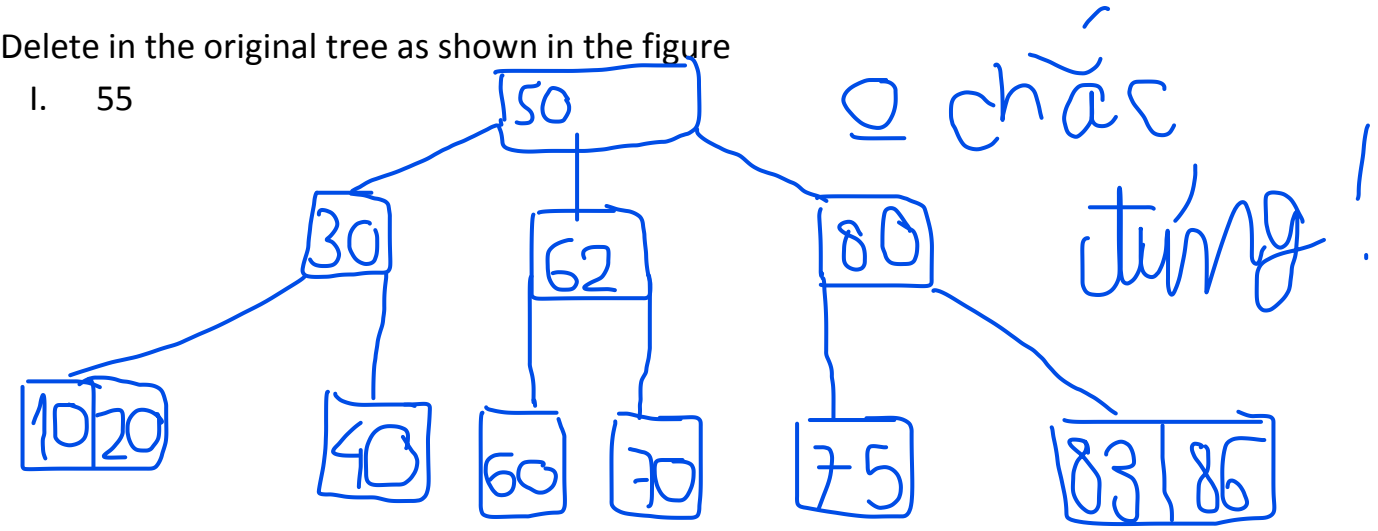
a) Insert in the original tree as shown in the figure

I. 66



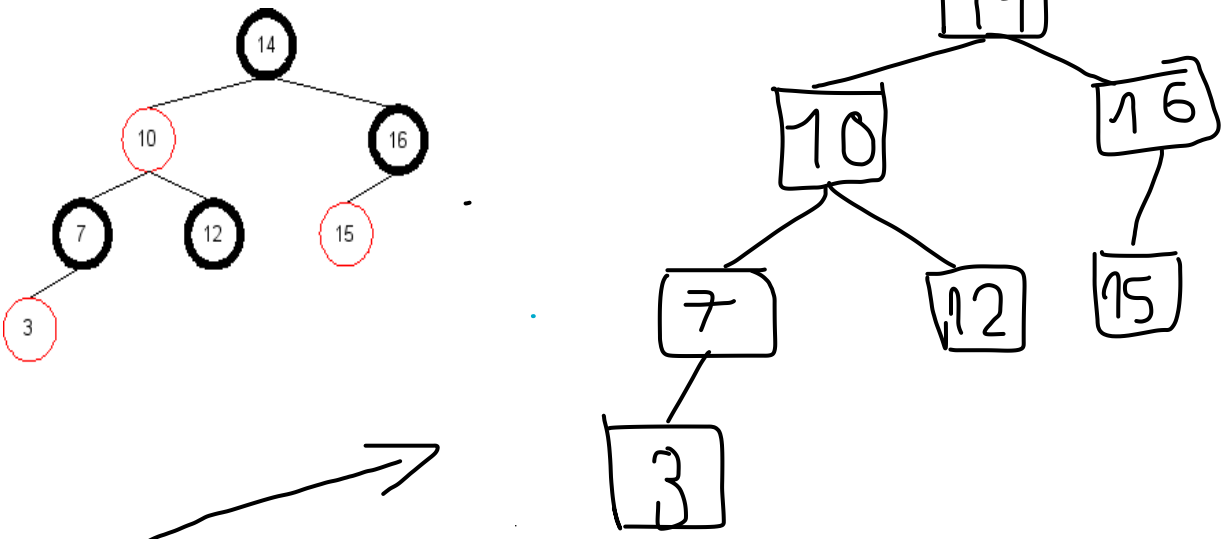
b) Delete in the original tree as shown in the figure

I. 55

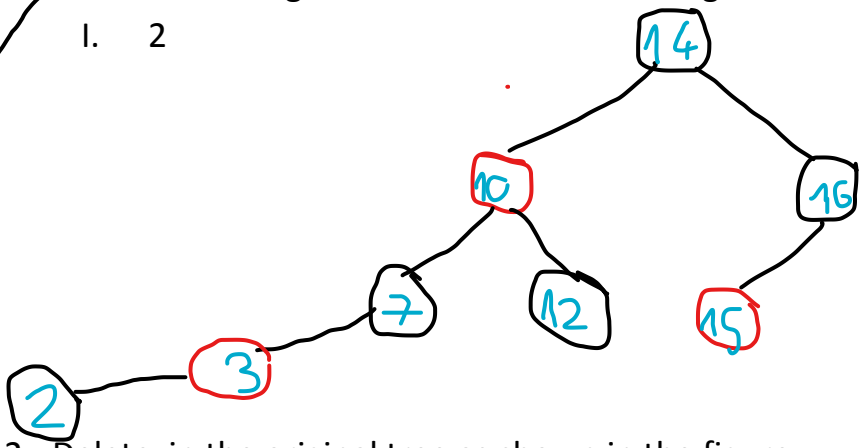


c) Convert the original 2-3-4 tree as shown in the figure into a red black tree

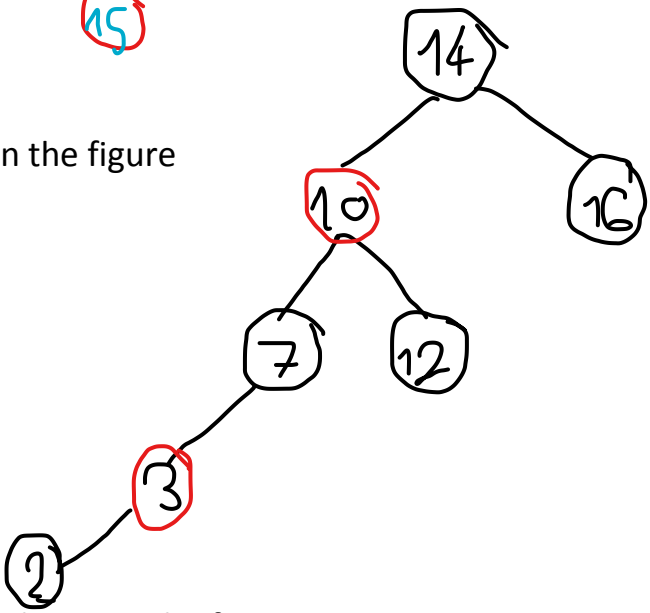
Question 8. Observe following red black tree and show the tree after each following operations **[6]**



1. Insert in the original tree as shown in the figure
   I. 2



2. Delete in the original tree as shown in the figure
   a. 15



3. Convert the original red black tree as shown in the figure into a 2-3-4 tree

Bonus Question . Provide C++ methods to compute the following in a BST that stores integer values. Prototype of the method is given as below: [6]

(i) Count the number of nodes

```
int NodeCount (BinNode *node)
```

(ii) Summation of all the value stored in a BST

```
int SumValue (BinNode *node)
```