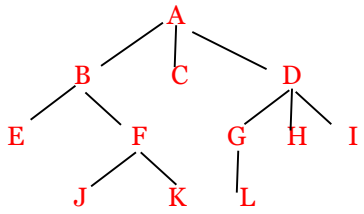
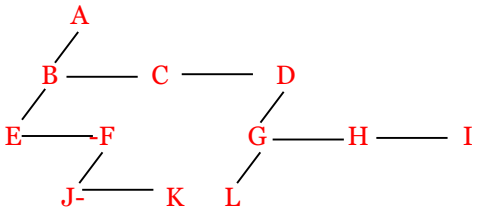
Quiz #4 2020. 5.18

학번: 이름:

1. [이진트리] 어떤 트리의 list representation이 다음과 같을때, 아래 질문에 답하시오(25점)
List Representation: (A, (B (E, F (J, K)), C, D (G (L), H, I)))

1) 위 List 표현을 트리로 그리시오 2) 1)번의 트리를 left-child Right-sibling (LCRS)으로 그리시오

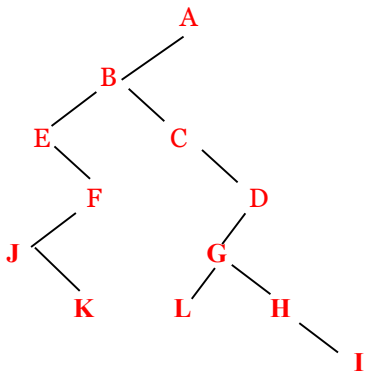




3) 위 2)번의 LCRS트리를 이진트리(Binary tree)로 그리고, 3가지 트리순회의 결과를 쓰시오

- 이진트리

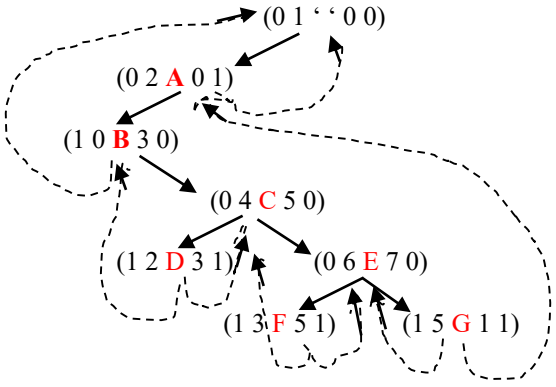
Inorder: E, J, K, F, B, C, L, G, H, I, D, A
Preorder: A, B, E, F, J, K, C, D, G, L, H, I
Postorder: K, J, F, E, L, I, H, G, D, C, B, A



2. 다음 Threaded tree의 데이터를 분석하여 (15점).

// **각 노드는 5개 필드로 구성되어 있다. (LeftThread, Leftchild, Data, Rightchild, RightThread)**
TTREE TREE[10] = {{0,1,' ',0,0},{0,2,'A',0,1},{1,0,'B',3,0},{0,4,'C',5,0},{1,2,'D',3,1},
{0,6,'E',7,0},{1,3,'F',5,1},{1,5,'G',1,1} };

1) 위데이터에 맞게 Threaded binary Tree를 그려보시오
2) Inorder Threaded tree Traversal을 한 결과를 쓰라 **결과: B D C F E G A**
3) Node G의 후속자는 무엇인가? A



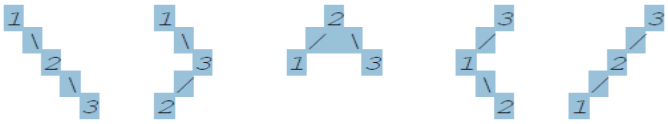
3. 위 2번 문제의 트리에 아래 코드를 실행시키면 출력되는 결과는? (10점)

```
void unknown(struct node *p) {
  while(p) {
     cout << p->data;
     if (p->right != NULL)      enqueue(Queue, p->right);
     if (p->left != NULL)       enqueue(Queue, p->left);
     p = dequeue(Queue);
   }}
```

(답): A B C E D G F

4. [BST] 노드의 데이터가 1,2, 3일 때, 이 3개의 노드로 가능한 모든 BST를 그리시오 (10점)

(Draw all possible binary search trees (BSTs) with the node 1, 2, and 3)



5. 다음 BST의 Insert 알고리즘을 Non-Recursive Version(iterative version) 으로 작성하시오(10점)

```
 Procedure INSERT (ptr, key) {
    if (ptr = NULL) {    // if empty
        create new_node(ptr);
        return ptr;
    }
    if   (key < ptr->data)   ptr->left =    INSERT(ptr->left, key);
    else                     ptr->right = INSERT(ptr->right, key);
    return ptr;
}
```

```
    void Tree::insertTree(int key){
          Node *p = root;    Node *prev = 0;
          while (p != 0) {
                prev = p;
                if (p->data < key)  p = p->right;
                else                p = p->left;
          }
          if (root == 0)                root = new Node(key);
          else if (prev->data < key)    prev->right = new Node(key);
          else if (prev->data > key)    prev->left = new Node(key);
}
```

6. 지난주 Lab9 BST 에서 leaf 노드와 nonleaf 노드의 갯수를 각각 구하였다. Leaf/nonleaf 알고리즘을 참조하여, **트리의 전체 노드 개수를 구하는 TotalBSTnodes 알고리즘을 작성하시오** (조건: Leaf/nonleaf 의 결과값을 더하면 안되며, recursive algorithm으로 작성해야 함) (10점)

```
Procedure TotalBSTNodes(Node *p, int count){ // count는 전체 노드의 개수를 반환

          if (p)
                count = Nonnodes(p->left, count) + Nonnodes(p->right, count) + 1;
          else
                count = 0;
          return count;
}
```