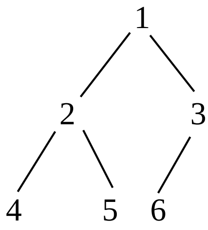
Data Structure - Exam II

2020/11/26

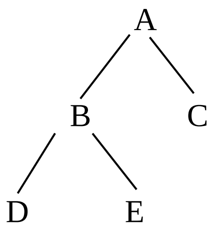
1. (10%) For any nonempty binary tree, if n_0 is the number of leaf nodes and n_2 the number of nodes of degree 2, prove that $n_0 = n_2 + 1$.

2. (5%) A **complete binary tree** of depth k (in which the root is at level 1 and the leaves are at level k) has totally at least x nodes and at most y nodes. What will be x and y?

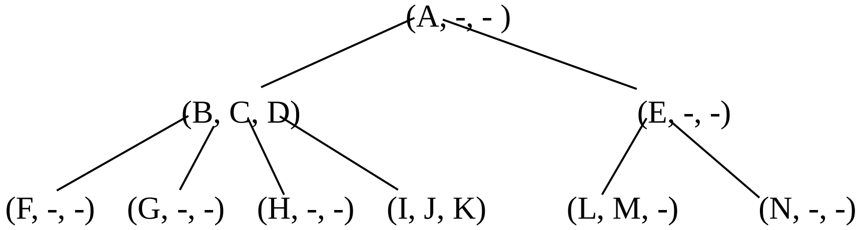
3. (5%) Show your result step by step if one deletion is made to the following min heap.



4. (5%) Show your result step by step if X is inserted into the following max heap, where X>A and D<E<B<C.



5. (10%) In the following B-tree, each (*,*,*) is a node and "-" stands for a NULL.



Answer the following questions. (NOTE: They are independent questions.)

(a) Give the resulting B-tree after inserting X into the tree, where X<A and X>K. Show how you get the answer step by step.

(b) Give the result after deleting N and then E from the tree. Also show your result step by step.

6. (15%) Let the tree in Question 5 be a 2-3-4 tree. Answer the following questions. (NOTE: They are independent questions.)

   (a) Show your result step by step when inserting X into the tree, where X<A and X>K.

   (b) Show your result step by step when deleting N from the tree.

   (c) Draw the corresponding red-black tree of this 2-3-4 tree.

7. (20%) Answer the following questions about in-order, pre-order, and post-order traversals of a binary tree.

   (a) (10%) We know that given the pre-order and the in-order traversals, we can determine a unique binary tree. Now let the pre-order traversal of a binary tree be ABCDEFGHIJ and the in-order traversal be DCBEFAGIHJ. Give a step-by-step inference of what the binary tree will look like.

   (b) (5%) If the in-order and the post-order traversals are known, can we determine a unique binary tree? Explain your reasons.

   (c) (5%) If the pre-order and the post-order traversals are known, can we determine a unique binary tree? Explain your reasons.

8. (10%) Answer the following questions about a height-balanced binary tree.

   (a) Give the formal definition of a height-balanced binary tree.

   (b) Let L_1 and L_2 be any two leaf nodes of a height-balanced binary tree. Can the maximum difference between L_1 and L_2 be greater than one? Draw a diagram to explain your reasons in detail.

9. (10%) The following is a C code segment. This program will turn a sequence of data into a max heap. Please fill in appropriate answer (an expression or a variable) in each blank space (i.e., in the rectangle).

```
void adjust_to_maxheap(int list[], int n, int i){
   int temp, large;
   temp = list[i];
   while(    (1)    < n){
      large = 2*i + 1;
      if((large + 1 ) < n &&      (2)     )
         large = large +1;
      if(temp >=   (3)   )
         break;
      list[i] =   (4)   ;
      i = large;
   }
   list[i] =   (5)   ;
}
void construct_heap(int list[], int n) {
   for(int i = n/2; i>=0; i--){
      adjust_to_heap(list, n , i);
   }
}
int main(void){
   int list[N]={15,42,29,66,73,15,10,19};
   construct_heap(list, N);
   for(int i = 0; i < N; i++ )
      printf("%d ", list[i]);
   return 0;
}
```

10. (10%) The following is a C code segment. By given the pre-order and the in-order traversals (which are strings) of a binary tree, this program will return the post-order traversal of this binary tree. Please fill in appropriate answer (an expression or a variable) in each blank space (i.e., in the rectangle).

```
typedef struct treenode{
   char data;
   struct treenode *left, *right;
} TreeNode;
TreeNode *genTreeNode(int data, TreeNode *left, TreeNode
*right) {
   TreeNode *node = malloc(sizeof(TreeNode));
   assert(node != NULL);
   node->data = data;
   node->left = left;
   node->right = right;
   return node;
}
void postorder(TreeNode *root) {
   if(root == NULL) return;
   postorder(  (1)   );
   postorder(  (2)   );
   printf("%c",  (3)   );
}
TreeNode *reconstruct(int n, char pre[], char ary[]) {
   if (n == 0) return NULL;
   int leftn = strchr(ary, pre[0]) - ary;
    /* strchr(): return a pointer to the first occurrence of
    pre[0] in ary */
   int rightn = n - leftn -1;
   return genTreeNode(  (4) , reconstruct(  (5) ,  (6) ,
    (7) ), reconstruct(  (8) ,  (9) ,  (10) ));
}
int main(void){
   char preorder[N], inorder[N]; /* N=80 */
   scanf("%s%s", preorder, inorder);
   int length = strlen(preorder);
   assert(length == strlen(inorder) && length < N);
   TreeNode *root = reconstruct(length, preorder, inorder);
   postorder(root);
   freeTree(root); /*free spaces of all nodes of the tree*/
   return 0;
}
```