COMP 2710 – Spring 2020
Midterm Exam2(take-home)

Instructions:

This semester we have lots of uncertainties during an online transition period. Dr. Li greatly appreciates all students' cooperation and patient, positive attitudes. Hopefully you stay healthy, stay safe and have a restful and sweet family time.

Maximum points possible: 100

Time: 11:59pm CST Mar 26 - 11:59pm CST Mar 27 (24 hours)

What to submit:

1. Your solution file with a format of "LastName_UserID.cpp" (It is tested in AU Server)

2. A .pdf file of your answers with a format of "Exam2_LastName_UserID.pdf". (Please type answers with your keyboard, hand-written answers are NOT accepted)

1. Tower of Hanoi. (40 points)

The Tower of Hanoi puzzle was invented by the French mathematician Edouard Lucas in 1883. He was inspired by a legend that tells of a Hindu temple where the puzzle was presented to young priests. At the beginning of time, the priests were given three poles and a stack of 64 gold disks, each disk a little smaller than the one beneath it. Their assignment was to transfer all 64 disks from one of the three poles to another, with two important constraints. They could only move one disk at a time, and they could never place a larger disk on top of a smaller one. The priests worked very efficiently, day and night, moving one disk every second. When they finished their work, the legend said, the temple would crumble into dust and the world would vanish.
(https://runestone.academy/runestone/books/published/cppds/Recursion/TowerofHanoi.html)

1.1 Please implement a "moveDisk" function in a given TowerOfHanoi_plain.cpp (20 points)

I/O format

```
Enter the number of disks:3
Move Disk 1 from A to C
Move Disk 2 from A to B
Move Disk 1 from C to B
Move Disk 3 from A to C
Move Disk 1 from B to A
Move Disk 2 from B to C
Move Disk 1 from A to C
The elapsed time is 0 seconds for moving 3 disks
```

1.2 Assume moving one single disk each time costs one second, how many seconds are spent if there are 3 disks, 8 disks and 10 disks. (6 points. 2 points each)

3 disks: 7
8 disks: 255
10 disks: 1023

1.3 How many seconds are spent if we have N disks. (5 points)

2 ^ n - 1

1.4 Since this is a 24-hour exam, how many disks can you successfully move from Peg A to Peg C? Don't guess it. List your brief calculation. (5 points)

24 hours = 86400 seconds

2 ^ n – 1 = 86400
Log(2 ^n) = log(86401)
n = 16.4 disks
so 16 disks

1.5 Running your own solution--TowerOfHanoi_plain.cpp, how many seconds does AU server spend on 16 disks? (4 points)

1 second

2. Usually a singly link list is used to denote linear data structures such as Array, Stack, while a doubly linked list is used to denote non-linear data structures such as Tree (Fig.1). (48 points)

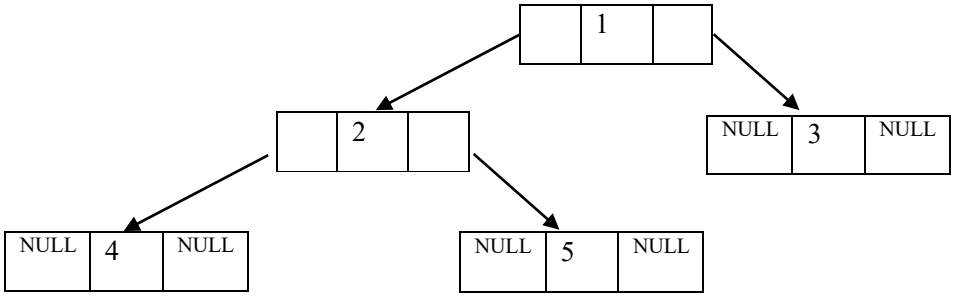


Fig. 1

Back to my college life, I worked on optimization of Tree Traversal in *Facebook*. Based on my 3-month work, all tuned algorithms were from three basic traversals (below).

(a) Inorder (Left, Root, Right) : 4 2 5 1 3

(b) Preorder (Root, Left, Right) : 1 2 4 5 3

(c) Postorder (Left, Right, Root) : 4 5 2 3 1

2.1 Please define each node of Tree from Fig.1 (12 points)

```
struct Node
{
   int data;
   struct Node *prev;

    struct Node *next;
   Node(int data)
   {
      this->data = data;

       prev = null;

       next = null;
   }
};
```

2.2 Please print 5 numbers with Inorder (pay attention to spaces between two numbers) (12 points)

```
// Algorithm Inorder(tree)
//1. Traverse the left subtree, i.e., call Inorder(left-subtree)
//2. Visit the root.
//3. Traverse the right subtree, i.e., call Inorder(right-subtree)
void printInorder(struct Node* node)
{
    if (node == NULL)
        return;

    /* first recur on left child */

    printInorder(node->prev);

    /* then print the data of node */

    cout << node->data <<  " ";

    /* now recur on right child */

    printInorder(node-> next);
}
```

2.3 Please print 5 numbers with Preorder (pay attention to spaces between two numbers) (12 points)

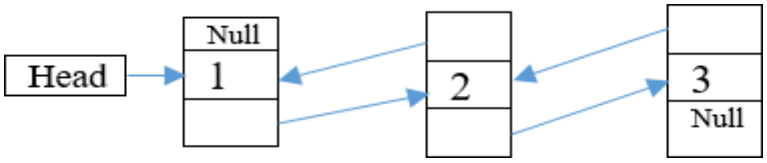
```
//Algorithm Preorder(tree)
//1. Visit the root.
//2. Traverse the left subtree, i.e., call Preorder(left-subtree)
//3. Traverse the right subtree, i.e., call Preorder(right-subtree)
void printPreorder(struct Node* node)
{
   if (node == NULL)
       return;

   /* first print data of node */
    cout << node -> data << " ";

   /* then recur on left subtree */

    printPreorder(node -> prev);

   /* now recur on right subtree */

   printPreorder(node -> next);
}
```

2.4 Please print 5 numbers with Postorder (pay attention to spaces between two numbers) (12 points)

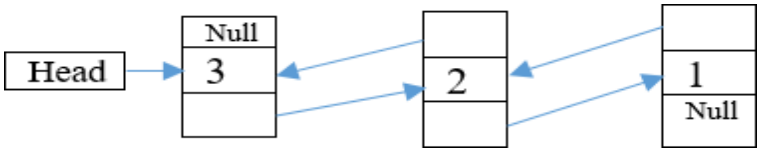
```
//Algorithm Postorder(tree)
//1. Traverse the left subtree, i.e., call Postorder(left-subtree)
//2. Traverse the right subtree, i.e., call Postorder(right-subtree)
//3. Visit the root.

void printPostorder(struct Node* node)
{
    if (node == NULL)
        return;

    // first recur on left subtree

     printPostorder(node -> prev);

    // then recur on right subtree

    pritnPostorder(node-> next);

    // now deal with the node

    cout << node -> data << " ";
}
```

3. Please fill out blank lines to reverse a given doubly link list with a given Node struct. (12 points)

Before:



After:




```
struct Node
{
   int data;
   struct Node *next;
   struct Node *prev;
};
void reverse(Node **head)
{
   Node *temp = NULL;
   Node *current = *head;
        /* swap next and prev for all nodes of a doubly linked list */
   while (current != NULL)
   {
      temp = current -> prev;

      current -> prev = current -> next;

      current -> next = temp;

       current  = current -> prev;
```

```
    }
    /* Before changing the head, check for the cases like empty list and list with only one node */
    if (temp != NULL)

            *head = temp -> prev;
}
```