CPE217 - Homework 5

Homework: Tree Traversals using Non-Recursive Technique

Homework Due Date: 16 August 2022

Patiwet Wuttisarnwattana, Ph.D.

Department of Computer Engineering

- คำชี้แจงการส่งงาน
- ให้ทุกคนเข้าโมดูล Assignment (Link สีแดง) อ่านคำอธิบายการบ้านใน PDF และโหลด Java Starter Code เพื่อทำความเข้าใจโจทย์
- หลังจากนั้นให้ทุกคนทำการบ้านพร้อมกันกับกลุ่มของตัวเอง พร้อมปรึกษากับ Core Person ในกลุ่มตัวเองว่าจะส่งคำตอบสุดท้ายว่าเป็นอะไร
- ทุกคนสามารถตรวจคำตอบของตัวเองได้ ว่าทำมาถูกต้องหรือไม่โดยใช้โมดูล Quiz (Link สีเหลือง) แต่อาจารย์จะตรวจคะแนนจาก Core Person เท่านั้น นศ ทุกคนจะได้คะแนนเท่ากันทั้งกลุ่ม แม้ว่าคุณจะส่งโค้ดใน Link สีเหลืองแตกต่างกันก็ตามที
- นศ ต้องเติมโค้ดในพื้นที่ที่กำหนดให้เท่านั้น ห้ามประกาศตัวแปรระดับคลาสเพิ่ม ห้ามสร้างฟังก์ชันอื่น ๆ หรือเรียกใช้ฟังก์ชันพิเศษเพิ่ม (ไม่แน่ใจสามารถสอบถามได้ที่อาจารย์) ให้เติมโค้ดใน Template ของอาจารย์เท่านั้น ฝ่าฝืนหักคะแนน 25% แม้ผลลัพธ์สุดท้ายจะทำงานได้ถูกต้อง
- เมื่อ Core Person ส่งคำตอบแล้ว ให้ Core Person เข้าโมดูล Assignment (Link สีแดง) และใส่รหัสของเพื่อนในกลุ่มลงในช่องคำตอบ
- TA จะตรวจคำตอบในโมดูล Quiz และนำคะแนนมาลงในโมดูล Assignment เพื่อให้ทุกคนในกลุ่มได้คะแนนเท่ากันครับ
- โค้ดของคุณต้องมี Comment เพื่ออธิบายว่าโค้ดดังที่เห็นอยู่นี้ทำงานอะไร หรือ if นี้ใช้เพื่อแยกกรณีใดออกมา กลุ่มไหนที่ไม่มีคอมเม้นต์จะถูกหักคะแนน 50% การเขียนคอมเม้นต์ไม่ต้องเขียนละเอียดยิบ เขียนเท่าที่คุณต้องการให้ผู้ตรวจทราบก็พอ

นศ ที่จะส่งคำตอบ ท่านต้องให้คำมั่นปฏิญาณต่อคำพูดดังต่อไปนี้ หากไม่สามารถทำได้ ท่านจะไม่มีสิทธิ์ส่งงาน

- *ข้าพเจ้าและเพื่อนในกลุ่มเข้าใจและตระหนักดีว่า ในการทำการบ้านนี้ ข้าพเจ้าและเพื่อนในกลุ่มจะช่วยกันทำงานนี้ให้เสร็จสิ้นเอง โดยไม่ปรึกษาหรือแบ่งปันข้อมูลกับกลุ่มอื่น ๆ หรือบุคคลภายนอก*
- *หากข้าพเจ้าเป็นรุ่นพี่ที่กลับมาเรียนวิชานี้อีกครั้ง ข้าพเจ้าตระหนักดีว่า ข้าพเจ้าจะทำงานให้เสร็จสิ้นเองอีกครั้ง โดยไม่ดูคำตอบของปีก่อน ๆ*
- *ข้าพเจ้าจะไม่เผยแพร่เนื้อหาโจทย์การบ้านนี้ออกสู่สาธารณะโดยเด็ดขาด*
- *หากข้าพเจ้าไม่สามารถปฏิบัติตามคำมั่นนี้ได้ ข้าพเจ้ายินดีที่จะยอมรับคะแนน ศูนย์คะแนน ในทุก ๆ การบ้านโดยไม่โต้แย้ง*

การบ้านนี้ให้นักศึกษา implement Tree Traversals using Non-recursion Technique โดยใช้ Java โดยให้มีคลาสดังต่อไปนี้

1. ให้สร้าง class ชื่อว่า Queue โดย class นี้ มีคุณสมบัติของ Queue ADT ตามที่ได้เรียนในห้องเรียน
2. ให้สร้าง class ชื่อว่า Stack โดย class นี้ มีคุณสมบัติของ Stack ADT ตามที่ได้เรียนในห้องเรียน
3. ให้ Stack และ Queue สามารถบรรจุ objects ของ class Node โดย class Node นี้มีคุณสมบัติของการเป็น Binary Tree
    a. ให้ Node แต่ละ Node สามารถบรรจุ data (key) ได้ค่า ๆ หนึ่ง โดยให้เป็นตัวแปรชนิด integer
    b. ให้ Node แต่ละ Node สามารถต่อกันเพื่อเป็นโครงสร้างข้อมูล Binary Tree ตามที่เรียนในห้องได้
    c. สมาชิกของ class Node ควรที่จะมี reference ชี้ไปยัง ลูกคนซ้าย (left child) และลูกคนขวา (right child)
    d. การบ้านนี้กำหนดให้ ไม่มี parent reference/pointer (ทราบลูก แต่ไม่ทราบแม่)
4. ให้ class Node มี 1 Constructor คือ Node(int data) ซึ่งทำหน้าที่ กำหนดค่าเริ่มต้นของ key จาก data
5. ในการบ้านนี้ กำหนดให้คุณใช้ Circular Array ในการ implement Queue ให้ Queue มีฟังก์ชันดังต่อไปนี้
    a. public void enqueue(Node node) ทำหน้าที่ enqueue Node ตามที่เรียนในห้อง
        - กำหนดให้ Circular Array สามารถบรรจุข้อมูลได้เต็ม capacity เช่น capacity = 10 ข้อมูลที่สามารถบรรจุได้คือ 10 พอดี หากข้อมูลที่ 11 เข้ามา ให้แจ้งว่า “Queue Overflow!!!”
    b. public Node dequeue() ทำหน้าที่ dequeue Node ตามที่เรียนในห้อง
        - หากทำการ dequeue ในขณะที่ Q ว่างอยู่ให้แจ้งว่า “Queue Underflow!!!”
    c. public void printQueue() ทำหน้าที่ แสดงว่าปัจจุบันนี้มีข้อมูลอะไรบรรจุอยู่ใน Queue บ้าง pattern การแสดงออกทาง Console ให้เป็นไปตามที่เห็นใน (ดังตัวอย่างด้านล่าง) เริ่มด้วย [Front] ลงท้ายด้วย [Back]
    d. public void printCircularIndices() ทำหน้าที่ แสดงว่าปัจจุบันนี้ front index กับ back index (ตามหลักการที่เรียนในห้อง)
6. ในการบ้านนี้ กำหนดให้คุณใช้ Array of Nodes ในการ implement Stack ให้ Stack มีฟังก์ชันดังต่อไปนี้
    a. public void push(Node node) ทำหน้าที่ push Node ตามที่เรียนในห้อง
        - หากข้อมูลที่ใส่เข้ามาใหม่ เกิน capacity ให้แจ้งว่า “Stack Overflow!!!”
    b. public Node pop() ทำหน้าที่ pop Node ตามที่เรียนในห้อง
        - หากทำการ pop ในขณะที่ Stack ว่างอยู่ให้แจ้งว่า “Stack Underflow!!!”
    c. public void printStack() ทำหน้าที่ แสดงว่าปัจจุบันนี้มีข้อมูลอะไรบรรจุอยู่ใน Stack บ้าง pattern การแสดงออกทาง Console ให้เป็นไปตามที่เห็นใน (ดังตัวอย่างด้านล่าง) เริ่มด้วย [Bottom] ลงท้ายด้วย [Top]
7. การบ้านนี้อาจารย์ได้เพิ่มคุณสมบัติพิเศษของ Node คือ คุณสามารถที่จะพิมพ์แผนภาพต้นไม้ออกมาทาง Console ได้ เพียงแค่เรียกใช้ฟังก์ชัน public void printTree() (ดังตัวอย่างด้านล่าง) คุณไม่ต้องเขียนฟังก์ชันนี้เอง แต่คุณต้องนำเข้า class พิเศษของอาจารย์เข้าไปด้วย โดยให้คุณทำตาม Step ดังต่อไปนี้
    a. ให้คุณนำไฟล์ BTreePrinter.java เข้าไปอยู่ในโปรเจคและ package ปัจจุบันของคุณ
    b. ให้คลาส Node ของคุณ สืบทอดคุณสมบัติ (OOP inheritance) ของคลาสที่มีชื่อว่า BTreePrinter (คุณควรที่จะรู้ว่าต้องใช้คำสั่งอะไรใน Java เพื่อ class หนึ่ง ๆ จะทำการสืบทอดคุณสมบัติของ class อีกอันหนึ่ง)

c. ให้คลาส Node ของคุณ มีฟังก์ชันที่ชื่อว่า public void printTree() โดยหน้าที่ของฟังก์ชันนี้คือการเรียกใช้ฟังก์ชัน protected void printTree(Node node) (ที่คุณสืบทอดมาจาก BTreePrinter) อีกทีหนึ่ง พารามิเตอร์ node ที่ส่งเข้าไป คือ root node ของแผนภาพต้นไม้

d. ให้คุณคิดว่า class BTreePrinter เป็นเครื่องมือในการแสดงผล คุณไม่จำเป็นต้องรู้ว่า class BTreePrinter ทำงานอย่างไร รู้แต่ว่าติดตั้งอย่างไรและใช้งานอย่างไรก็พอ

e. ให้คุณทำการสร้าง function contructTree1() และ constructTree2() เพื่อสร้าง trees 2 ต้น (โดย return ออกมาเป็น Node objects) เพื่อให้สามารถแสดงแผนภาพต้นไม้ได้ ผ่านการเรียกใช้ฟังก์ชัน printTree() ตามตัวอย่างการทำงาน 10.1 และ 10.2

8. ให้คุณทำการ implement Breadth-first Traversal โดยใช้ Queue ตามที่คุณเรียนในห้อง โดยทำเป็นฟังก์ชันที่เป็นหนึ่งของ class Node โดยมี prototype ดังต่อไปนี้

   a. public void printBFT()

   b. ให้ pattern การพิมพ์ออกทาง console ให้เป็นไปดังตัวอย่างด้านล่าง เริ่มต้นด้วยคำว่า “BFT node sequence [” ลงท้ายด้วย “]”

9. ให้คุณทำการ implement PreOrder Depth-first Traversal โดยใช้ Stack ตามที่คุณเรียนในห้อง โดยทำเป็นฟังก์ชันที่เป็นส่วนหนึ่งของ class Node โดยมี prototype ดังต่อไปนี้

   a. public void printDFT()

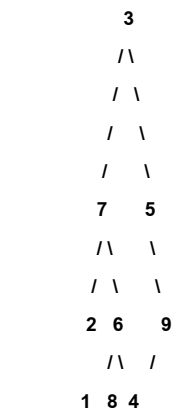
   b. ให้ pattern การพิมพ์ออกทาง console ให้เป็นไปดังตัวอย่างด้านล่าง เริ่มต้นด้วยคำว่า “DFT node sequence [” ลงท้ายด้วย “]”

10. ตัวอย่างการทำงาน

Java code

```
public static void main(String[] args) {
        Node tree = constructTree1(); // Create your own function in this Class
        tree.printTree();
}
```

Output (แผนภาพต้นไม้ด้านล่างอาจแตกต่างกับผลลัพธ์จริงเล็กน้อย)

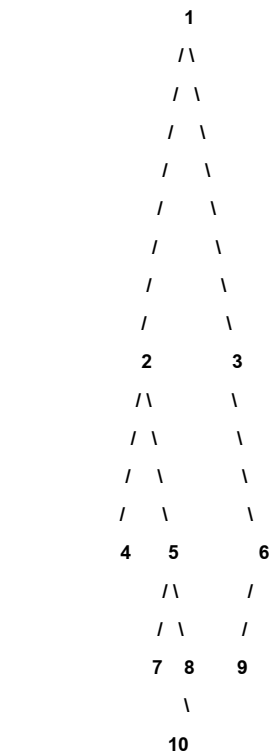


Java code

```
public static void main(String[] args) {
```

```
        Node tree = constructTree2(); // Create your own function in this Class
        tree.printTree();
}
```

Output (แผนภาพต้นไม้ด้านล่างอาจแตกต่างกับผลลัพธ์จริงเล็กน้อย)



Java code

```
public static void main(String[] args) {
        Stack s = new Stack(4);
        s.pop();
        s.push(new Node(5));
        s.push(new Node(6));
        s.push(new Node(7));
        s.push(new Node(8));
        s.printStack();
        s.push(new Node(9));
        System.out.println(s.pop().data);
        System.out.println(s.pop().data);
        System.out.println(s.pop().data);
        s.printStack();
}
```

Output

```
Stack Underflow!!!
[Bottom] 5 6 7 8 [Top]
Stack Overflow!!!
8
7
6
[Bottom] 5 [Top]
```

Java code

```
public static void main(String[] args) {
        Queue q = new Queue(4);
        q.dequeue();
        q.enqueue(new Node(5));
        q.enqueue(new Node(6));
        q.enqueue(new Node(7));
        q.enqueue(new Node(8));
        q.printQueue();
        q.enqueue(new Node(9));
        System.out.println(q.dequeue().data);
        System.out.println(q.dequeue().data);
        System.out.println(q.dequeue().data);
        q.printQueue();
}
```

Output

```
Queue Underflow!!!
[Front] 5 6 7 8 [Back]
Queue Overflow!!!
5
6
7
[Front] 8 [Back]
```

Java code

```
public static void main(String[] args) {
        Queue q = new Queue(4);
```

```
        q.printCircularIndices();
        q.enqueue(new Node(5));
        q.enqueue(new Node(6));
        q.printCircularIndices();
        q.enqueue(new Node(7));
        q.enqueue(new Node(8));
        q.printCircularIndices();
        q.printQueue();
        System.out.println(q.dequeue().data);
        q.printCircularIndices();
        System.out.println(q.dequeue().data);
        q.printCircularIndices();
        System.out.println(q.dequeue().data);
        q.printCircularIndices();
        q.enqueue(new Node(9));
        q.enqueue(new Node(10));
        q.enqueue(new Node(11));
        q.printQueue();
}
```

Output

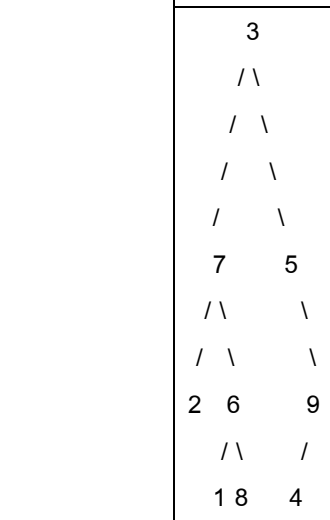
```
Front index = 0 Back index = 0
Front index = 0 Back index = 2
Front index = 0 Back index = 0
[Front] 5 6 7 8 [Back]
5
Front index = 1 Back index = 0
6
Front index = 2 Back index = 0
7
Front index = 3 Back index = 0
[Front] 8 9 10 11 [Back]
```

Java code

```
public static void main(String[] args) {
        Node tree = constructTree1();
```

```
        tree.printTree();
        tree.printBFT();
        tree.printDFT();
}
```

Output (แผนภาพต้นไม้ด้านล่างอาจแตกต่างกับผลลัพธ์จริงเล็กน้อย)



```
BFT node sequence [ 3 7 5 2 6 9 1 8 4 ]
DFT node sequence [ 3 7 2 6 1 8 5 9 4 ]
```

11. โปรดใช้ Starter code ที่อาจารย์แนบให้