First Quiz - Version A
CS 1102 Computer Science 2

Spring 2021

Tuesday March 2, 2021
Instructor Muller

KEY

Before reading further, please write your name on the top of all of your quiz answer sheets.

This is an open notes and open book quiz. But **collaboration is expressly prohibited**.

- Partial credit will be given so be sure to show your work.

Problem	Points	Out Of
1		4
2		6
Total		10

- Some variations of this quiz have problems for which you may want to use an ADT such as a `Stack<T>`, `Queue<T>` or `Deque<T>`. Feel free to assume that the `Deque<T>` ADT is available, implemented with a class `ArrayDeque<T>`.
- Some of the problems on the quiz variations involve linked-lists of nodes as shown on the right.
- We'll use the standard Java notation `{ 1, 2, 3 }` for an integer array and we'll use the notation `[ 1, 2, 3 ]` for a linked-list of integers.
- There is no need to be concerned with visibility attributes such as `public` or `private` on this quiz.
- Feel free to write helper functions if you need them.
- **Please write neatly.**

```
interface Deque<T> {
  void pushLeft(T item);
  T popLeft();
  void pushRight(T item);
  T popRight();
  int size();
  boolean isEmpty();
  }

class ArrayDeque<T>
  implements Deque<T> {
  ...
}
```

```
static class Node {
  int info;
  Node next;
  // assume any Node constructors that you want
}

+--------+--------+
|  info  | next o-+-->
+--------+--------+

// [ 1, 2, 3 ]
new Node(1, new Node(2, new Node(3, null)))

    +---+---+     +---+---+     +---+---+
o-->| 1 | o-+---> | 2 | o-+---> | 3 | o-+-+
    +---+---+     +---+---+     +---+---+ =
```

Problem 1: 4 Points

Write a function `static int[] treble(int[] a)` that returns a new array in which each item of `a` has been trebled, i.e., duplicated 3 times. For example, with `int[] a = {1, 2, 3}`, the call `treble(a)` would return the array `{1, 1, 1, 2, 2, 2, 3, 3, 3}`.

Answer:

```
// 1.3: treble each item in an integer array
private static int[] treble(int[] a) {
  int[] b = new int[a.length * 3];
  for (int i = 0; i < b.length; i++)
    b[i] = a[i / 3];
  return b;
}
```

Problem 2: 6 Points

Solve **only one** of the following two problems.

1. Write a function `static Node rotate(Node a, int n)` that accepts a linked list of integers `a` and an integer `n` and returns a new linked list just like `a` but with the integers of `a` rotated to the right by `n` places. For example, with

   ```
   Node a = new Node(4, new Node(8, new Node(16, null))); // [ 4, 8, 16 ]
   ```

   the call `rotate(a, 2)` would return the list `[ 8, 16, 4 ]`.

   Answer:

   ```
   // 2.3A: rotate a linked list to the right n places
   public static Node rotate(Node a, int n) {
     Deque<Integer> dq = new ArrayDeque<Integer>();
     while (a != null) {
       dq.addLast(a.info);
       a = a.next;
     }
     while (n > 0) {
       dq.addFirst(dq.pollLast());
       n--;
     }
     Node answer = null;
     while(!dq.isEmpty())
       answer = new Node(dq.pollLast(), answer);
     return answer;
   }
   ```

2. Write a function `static Node merge(Node a, Node b)`. You may assume that the integers in the linked lists `a` and `b` are both in strictly ascending order. The `merge` function should return a new list of nodes with the values of `a` and `b` merged into ascending order. For example, with

```
Node a = new Node(1, new Node(2, new Node(3, null))); // [ 1, 2, 3 ]
Node b = new Node(3, new Node(4, null));              // [ 3, 4 ]
```

the call `merge(a, b)` should return the linked list `[ 1, 2, 3, 3, 4 ]`. Note that either or both lists might be `null`.

Answer:

```
// 2.3B: merge two ascending sorted linked lists of ints into one linked list
public static Node merge(Node a, Node b) {
  Node answer = new Node();
  Node current = answer;
  while(a != null && b != null) {
    if (a.info <= b.info) {
      current.info = a.info;
      a = a.next;
    }
    else {
      current.info = b.info;
      b = b.next;
    }
    if (a == null)
      current.next = b;
    else if (b == null)
      current.next = a;
    else {
      current.next = new Node();
      current = current.next;
    }
  }
  return answer;
}
```