Rubin Peci
Data Structures
Chee Yap
HW 5

A.1
R-3.6
Formula - next = (a * cur + b) % n
next = (12 * 92 + 5) % 100 = 9
next = (12 * 9 + 5) % 100 = 13
next = (12 * 13 + 5) % 100 = 61
next = (12 * 61 + 5) % 100 = 37
next = (12 * 37 + 5) % 100 = 49

```
Node secondLast(Node u) {
        if(u == null || u.next == null)
                return null;
        Node tmp = u;
        while(tmp.next.next != null) {
                tmp = tmp.next;
        }
        return tmp;
}
```

A.2
To find the middle of a doubly linked list with sentinel nodes, you have two traversal nodes, one starting from the head and the other starting from the tail. The one starting from the head goes to the next one, and the one from the tail goes to the previous one. This keeps going until the traversals fall on the same node, or the head's next node is the tail one. Then, stop the loop and return the node that started from the header.

Pseudo code:

```
front = head sentinel.next
back = tail sentinel.prev
while (front != back && front.next != back)
        front = front.next
        back = back.prev

return front
```

A.3
algorithm -
make an array of the numbers n - 1, and then add them to a stack in different order
add the nth number after popping to every single possible position - beginning, in between each term, and at the end.
pseudo-code

```
permutation(int n)
        int tmp = n;
        Stack stk0;
        for(int i = 0; i < n; i++) {
                while(tmp > 0)
                        stk0.push(tmp)
                        tmp--;
                tmp = n;
        }
```

A.4
a.

```
Node genRhoList(int m, int n) {
        if (m < n) {
                Node head = new Node();
                Node tmp = head;

                int count = 0;
```

```
                while(count < m - 1) {
                        tmp.next = new Node();
                        tmp = tmp.next;
                        count++;
                }
                Node loopClose = tmp;

                //head.show("Tail:");
                //System.out.println("Value of count: " + count);

                while(count < n) {
                        tmp.next = new Node();
                        tmp = tmp.next;
                        count++;
                }

                //System.out.println("Value of count: " + count);
                //loopClose.show("Starting from LoopClose:");
                tmp.next = loopClose;
                return head;
        } else {
                Node head = new Node();
                Node tmp = head;
                for(int i = 0; i < n; i++) {
                        tmp.next = new Node();
                        tmp = tmp.next;
                }
                return head;
        }
}

b.
int sizeRhoList(Node u) {
        Node slow = u;
        Node fast = u;
        int count = 0;
        while(slow != null && fast.next != null){
                slow = slow.next;
                fast = fast.next.next;
                if (slow == fast)
                        break;
                count++;
        }
        return count;
}
```