

## 1 PandaList

Reference the class definitions below:

```
1 public class Panda {
2     static String food;
3     int age; //any int
4     boolean happy;
5
6     public Panda(int age) {
7         this.age = age;
8     }
9 }
10
11 public class PandaList {
12     public Panda item;
13     public PandaList rest;
14
15     public PandaList(Panda p) {
16         this.item = p;
17         this.rest = null;
18     }
19 }
```

What output will result from executing the following main method? Note, we **strongly** suggest you draw a box and pointer diagram.

```
1 public static void main(String[] args) {
2     Panda p = new Panda(1);
3     PandaList lst = new PandaList(p);
4     lst.rest = new PandaList(p);
5     lst.rest.rest = lst;
6     p = new Panda(-10);
7     PandaList temp = lst;
8     lst = new PandaList(p);
9     lst.rest = temp;
10    lst.rest.rest.rest.item.food = "Bamboo";
11    temp.rest.rest.item.happy = true;
12    temp.item.food = "shoots";
13    lst.rest.rest.rest = lst;
14    lst.rest.rest.rest.item.happy = false;
15    lst.rest.rest.item.happy = true;
16    System.out.println(temp.rest.item.age);           // Print statement 1
```

```
17         System.out.println(temp.rest.rest.rest.item.food); // Print statement 2
18         System.out.println(lst.rest.item.happy);           // Print statement 3
19         System.out.println(temp.rest.rest.item.happy);    // Print statement 4
20         System.out.println(lst.rest.rest.item.age);        // Print statement 5
21     }
```

For each of the following sub-questions, select what is printed out by the corresponding print statements.

**Print Statement One:**  -10  1  false

**Print Statement Two:**  Bamboo  shoots  null

**Print Statement Three:**  true  false

**Print Statement Four:**  true  false

**Print Statement Five:**  -10  1  false

## 2 Linked List Potpourri

Reminder: constant-time here means that regardless of the length of the list, this operation will take the same amount of time.

**Part 1** An encapsulated **singly** linked list with a pointer to **only** the first element allows for which of the following methods to be constant-time operations (assuming a reasonable implementation)?

- [ ] `addFirst(int x)`
- [ ] `addLast(int x)`
- [ ] `removeFirst()`
- [ ] `removeLast()`
- [ ] `remove(Node n)` (destructively removes Node `n` from the list)

**Part 2** An encapsulated **singly** linked list with a pointer to **both** the first element and the last element allows for which of the following methods to be constant-time operations (assuming a reasonable implementation)?

- [ ] `addFirst(int x)`
- [ ] `addLast(int x)`
- [ ] `removeFirst()`
- [ ] `removeLast()`
- [ ] `remove(Node n)` (destructively removes Node `n` from the list)

**Part 3** An encapsulated **doubly** linked list with a pointer to **both** the first element and the last element allows for which of the following methods to be constant-time operations (assuming a reasonable implementation)?

- [ ] `addFirst(int x)`
- [ ] `addLast(int x)`
- [ ] `removeFirst()`
- [ ] `removeLast()`
- [ ] `remove(Node n)` (destructively removes Node `n` from the list)