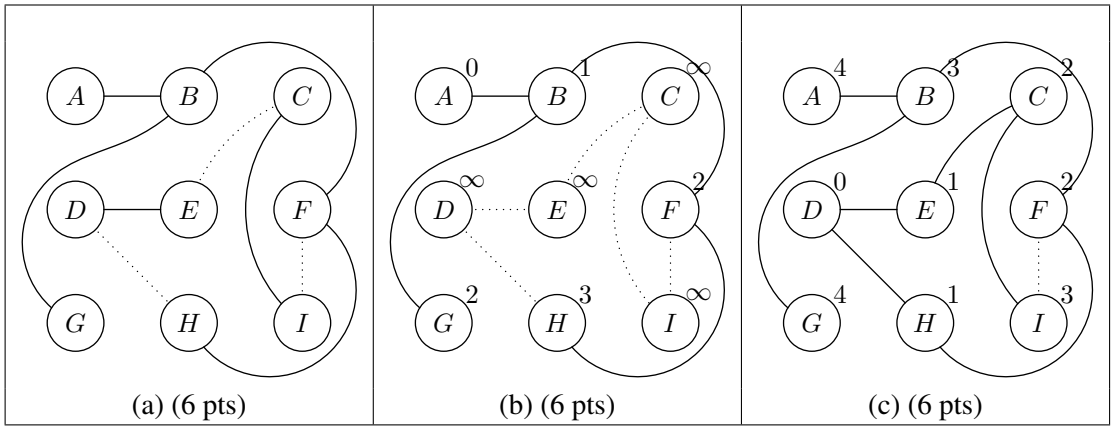
Com S 228 Fall 2014 Final Exam Key and Rubric

1. (18 pts)

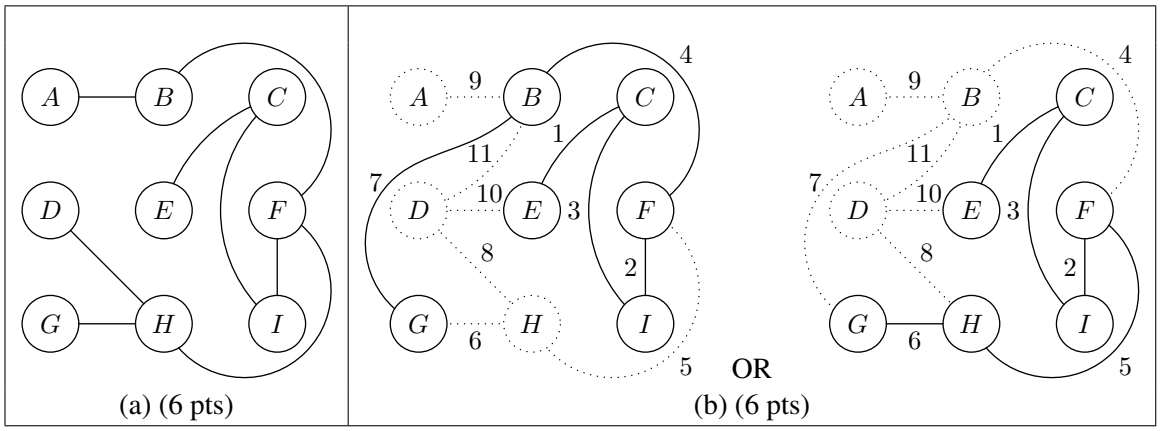
   - Students lose one point per error up to a max of 6 per part.
   - We forgive failure to circle vertices if:
     - the vertex has an incident edge;
     - the vertex is correctly marked with a depth; or
     - the vertex is incorrectly marked with a non-infinite depth.



(a) (6 pts) (b) (6 pts) (c) (6 pts)

2. (12 pts)

   - Students lose one point per error up to a max of 6 per part.
   - We forgive failure to circle vertices if the vertex has an incident edge.



(a) (6 pts) (b) (6 pts)

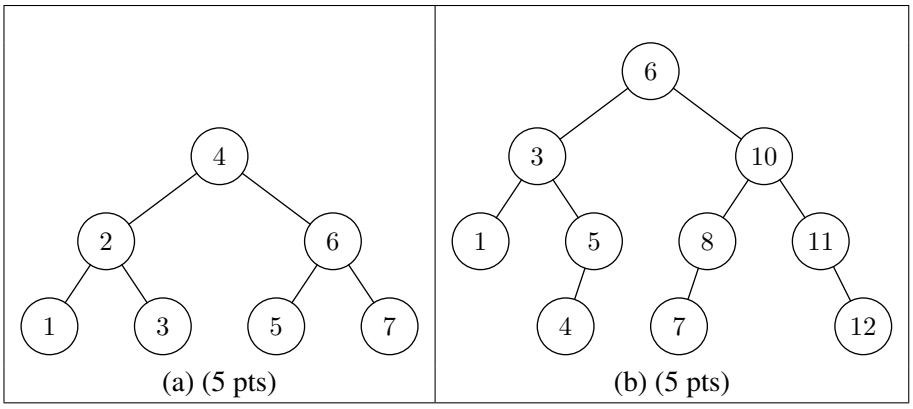
3. (12 pts)

	Row	Array						
	0	0	4	1	5	3	2	6
	1	0	4	6	5	3	2	1
*	2	0	5	6	4	3	2	1
	3	6	5	0	4	3	2	1
*	4	6	5	2	4	3	0	1
	5	1	5	2	4	3	0	6
*	6	5	1	2	4	3	0	6
	7	5	4	2	1	3	0	6
*	8	0	4	2	1	3	5	6
	9	4	0	2	1	3	5	6
*	10	4	3	2	1	0	5	6
	11	0	3	2	1	4	5	6
	12	3	0	2	1	4	5	6
	13	3	1	2	0	4	5	6
*	14	0	1	2	3	4	5	6
	15	2	1	0	3	4	5	6
	16	0	1	2	3	4	5	6
*	17	1	0	2	3	4	5	6
	18	0	1	2	3	4	5	6

- Each marked line and the demarcation are evaluated boolean right or wrong.
- Count total number of wrong answers (maximum possible is 8). Call that A_w.
- Student's score is: $12 - \lfloor 1.5 \times A_w \rfloor$.

4. (10 pts)

- (a) (5 pts)
  - -2 for first error.
  - -1 for each additional error.
  - Maximum penalty of 5 points.
  - If student makes an error, penalties should not compound. Grader must take care to ensure that subsequent points are awarded *as if* the preceding error had been correct. Depending on how badly this damages the tree, it may not be possible to salvage the student's score.
- (b) (5 pts)
  - -5 if the tree has 11 nodes (nothing removed).
  - -3 if the tree has other than 10 or 11 nodes.
  - -2 if the successor (10) wasn't chosen correctly.
  - -2 if 9 is still in the tree.
  - -2 for each additional incorrect node or value.
  - Maximum penalty of 5 points.



(a) (5 pts) (b) (5 pts)

5. (30 pts)

(a) (15 pts)

- Require explicit test and `NullPointerException`, 3 points.
- Require two recursive calls if child is not null, 3 points each.
- Require two comparisons of return value of recursive calls with `current.value`, 2 points each.
- Return the max value, 2 points.
- Do not use any other APIs, helper methods, etc., 7 points.
- Make no assumptions about the tree, `Node`, etc., 7 points.
- 1 to 2 points each for unforeseen errors, graders discretion.
- Maximum penalty of 15 points.

```
public Integer max(Node current)
{
  Integer tmp, val;

  if (current == NULL) {
    throw new NullPointerException();
  }

  val = current.value;

  if (current.left) {
    tmp = max(current.left);
    if (tmp > val) {
      val = tmp;
    }
  }
  if (current.right) {
    tmp = max(current.right);
    if (tmp > val) {
      val = tmp;
    }
  }

  return val;
}
```

(b) (15 pts)

- Require explicit test and `NullPointerException`, 3 points.
- Require two enqueue operations per iteration if child not null, 3 points each.
- Require queue is parameterized with `Node`, 2 points.
- Require loop recognize and terminate on empty queue, 2 points.
- Require one correctly-used dequeue operation per iteration, 2 points.
- Require at least one correctly used front operation per iteration, 2 points.
- Require comparison between `value` stored in node at front of queue with min once per iteration, 2 points.
- Require root to be correctly processed, 3 points.
- Require return of min value, 2 points.
- Do not use any other APIs, helper methods, etc., 7 points.
- Make no assumptions about the tree, `Node`, etc., 7 points.
- 1 to 2 points each for unforeseen errors, graders discretion.
- Maximum penalty of 15 points.

```
public Integer min()
{
  Queue<Node> q;
  Integer m;
  Node c;

  if (current == NULL) {
    throw new NullPointerException();
  }

  q = new Queue();

  c = root;
  m = root.value;
  while (c != null) {
    if (c.value < m) {
      m = c.value;
    }

    if (c.left) {
      q.enqueue(c.left);
    }
    if (c.right) {
      q.enqueue(c.right);
    }

    c = q.front();
    q.dequeue();
  }

  return m;
}
```

6. (18 pts)

- -2 points for incorrect answers
- -1 point for otherwise correct answers that fail to use correct notation (Big-O).
- For (i), accept $O(n)$ or anything clever or sufficiently self deprecating.

$O(\lg n)$ (a) (2 pts)	$O(1)$ (b) (2 pts)	$O(n \lg n)$ (c) (2 pts)
$O(V + E)$ (d) (2 pts)	$O(\lg n)$ (e) (2 pts)	$O(n)$ (f) (2 pts)
$O(E)$ (g) (2 pts)	$O(V^2)$ (h) (2 pts)	$O(1)$ (i) (2 pts)

Note on (e): This answer is incorrect, but we didn't catch it until grading. It's actually $\Omega(\lg n)$. We never taught that, and the problem worked out okay, regardless.