(a) Consider each grouping of code below. Identify any errors in the code and explain what is wrong.

```
ArrayList<int> numbers = new ArrayList<int>();
```

```
List<String> words = new ArrayList<String>();
```

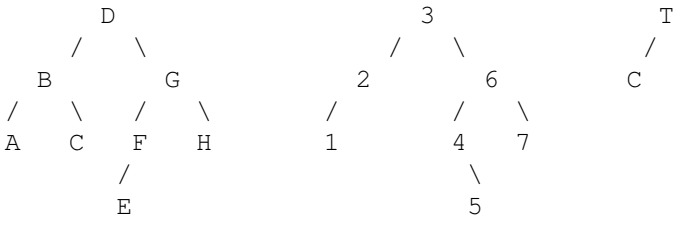
```
Integer[] integers = new Integer[3];
integers[0].toString();
```

(b) Circle the bullet in front of each statement that accurately describes your instructor's course policies:

- Having a copy of all or part of another student's source code, just to look at for a little help, is considered cheating.
- You must submit every lab assignment in order to pass this course.
- The final exam for this course will be comprehensive.
- If you miss more than three lectures, your grade will be dropped by half a letter grade.
- If you don't do your homework you will not be allowed to attend some classes.
- All students must submit a copy of their lecture notes before the end of the day.
- Lab attendance is required.
- Your instructor is such a nice guy that if you turn in an assignment 45 minutes late, he wouldn't consider it late.
- Your instructor will give you a free lunch if you invite him to lunch.

Do not circle the bullet in front of any statement that incorrectly describes your instructor's course policies.
Feel free to provide an explanation for your answer if you believe the statement to be ambiguous.

Consider the following three binary trees. For each tree that is a red-black tree, color the nodes (use circles to represent red and squares to represent black). For each tree that is not a red-black tree, indicate why it is not a red-black tree.



(a) Using the same assumptions as in lecture, implement the following method from the ArrayList class. You may assume that the size() method has been implemented.

```
public E remove(int index) {
```

(b) Circle the bullet in front of each statement that accurately describes your instructor's course policies:

- Having a copy of all or part of another student's source code, just to look at for a little help, is considered cheating.
- You must submit every lab assignment in order to pass this course.
- The final exam for this course will be comprehensive.
- If you miss more than three lectures, your grade will be dropped by half a letter grade.
- If you don't do your homework you will not be allowed to attend some classes.
- All students must submit a copy of their lecture notes before the end of the day.
- Lab attendance is required.
- Your instructor is such a nice guy that if you turn in an assignment 45 minutes late, he wouldn't consider it late.
- Your instructor will give you a free lunch if you invite him to lunch.

Do not circle the bullet in front of any statement that incorrectly describes your instructor's course policies.
Feel free to provide an explanation for your answer if you believe the statement to be ambiguous.

(a) Based on your memory, sketch the plot that showed time it took to execute the contains method on an ArrayList. Label the axes, but you do not need to show units.

(b) Implement the following ArrayList method using the same assumptions that we made in lecture.

```
public boolean contains(E target) {
```

(c) Give the worst-case big-oh time complexity for the method implemented in part (b). Be sure to justify your answers.

Recall that our LinkedList implementation had one attribute:

```
private Node head;
```

Implement the get(int index) method for the class we have been developing in lecture.

Recall the LinkedList class that we have been developing in lecture had one attribute: head. Implement the following method that will add an element to the front of the list:

```
public boolean addToFront(E element) {
```

}

Suppose that the LinkedList class that we have been developing in lecture was modified so that it only had one attribute: tail that pointed to the last node in the list. Further, suppose that the Node<E> class was modified so that the next attribute was replaced with a previous attribute that points to the previous node in the list instead of the next node in the list. Implement the contains method for the list

Consider the following partial implementation of a CircularQueue class. Identify all (if any) errors in the code. For each error identified, explain why it is an error.

```
public class CircularQueue<E> {

  private static final int DEFAULT_CAPACITY = 9;
  private E[] data;
  private boolean isEmpty;
  private int front;
  private int back;

  public CircularQueue() {
    this(DEFAULT_CAPACITY);
  }

  public CircularQueue(int capacity) {
    data = (E[])new Object[capacity];
    isEmpty = true;
    front = capacity-1;
    back = capacity-1;
  }

  public boolean offer(E element) {
    boolean added = false;
    if((isEmpty || front!=back) && element==null) {
      added = true;
      isEmpty = false;
      data[back++] = element;
    }
    return isEmpty;
  }
// ...









}
```

Write a recursive helper method for a BinaryTree<E> class that will return the height of a subtree and is compatible with the following method:

```
public int height() {
  return height(root);
}
```

Consider a class called Image that has the following methods:

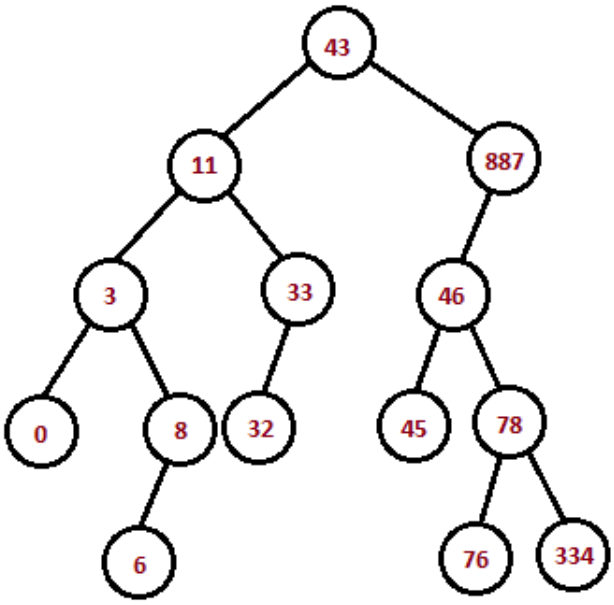
- int getHeight() – Returns the height of the image.
- int getWidth() – Returns the width of the image.
- boolean isFilled(int x, int y) – Returns true if the pixel at the specified location is filled.
- void setFilled(int x, int y) – Fills the pixel at the specified location.
- int areaFill(int x, int y) – Recursively fills the specified pixel and the area surrounding it up to bordering pixels at are filled or the edge of the image and returns the total number of pixels filled.

An IllegalArgumentException is thrown if an invalid pixel location is passed to any of the last three methods. Implement the areaFill method.

Complete the implementation of the remove() method in the following class uses a hash table to implement the Set interface.

```
public class HashQuiz<E> implements Set<E> {
  private ArrayList<E>[] table;

  public HashSet() {
    table = (ArrayList<E>[])new ArrayList[7919];
  }

  // ...

  public boolean remove(Object target) {









  }
}
```

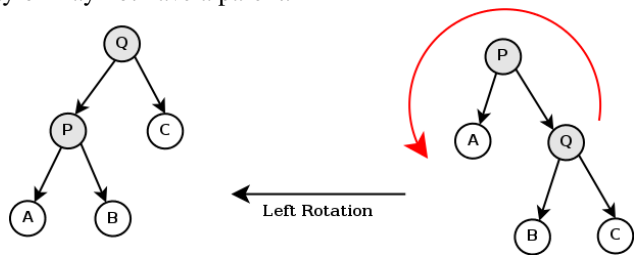
Draw the following binary search tree after the 11 is removed.



True/false. Cross out any of the statements below that are not true. Do not cross out the statement if it is true.

- It is not possible to have two elements in a Set if they are identical (o1.equals(o2)==true).
- It is not possible to have two entries in a Map if they have identical keys.
- It is not possible to have two entries in a Map if they have identical values.
- The contains() method of the TreeSet class is $O(n)$.
- The containsKey() method of the TreeMap class is $O(n)$.
- The containsValue() method of the TreeMap class is $O(n)$.
- It is possible to iterate over all elements in a TreeSet in $O(\log n)$ time.
- It is possible to implement the Set<E> interface using only one attribute: E[] data.
- The E in Set<E> can be any Java class name.
- The K in TreeMap<K, V> can be any Java class name.

Consider the following graphic that shows a similar rotation to the one discussed in lecture. Complete the leftRotate method shown below. The method is passed a reference to **P** and must return a reference to **Q**. The tree on the right shows the tree before leftRotate is called and the tree on the left shows the tree after the call. Keep in mind that **P** may or may not have a parent.




```
public Node<E> leftRotate(Node<E> p) {
  Node<E> q = p.right;
  if(q!=null) {
    Node<E> a = p.left;
    Node<E> b = q.left;
    Node<E> c = q.right;
    Node<E> parent = p.parent;
```

```
  }
  return q;
}
```

In lab 7 you used an ArrayList to store the Entrys in your LookupTable. Suppose that the Internal Revenue Service now wishes to use the LookupTable to store all tax returns (based on the filer's Social Security number). Assume your implementation made correct use of the Collections.binarySearch() method.

(a) Consider replacing the ArrayList with a LinkedList. Would this be better, equivalent, or worse? Explain your answer.

(b) Suppose the LookupTable class was modified to make use of a balance binary search tree instead of the ArrayList/Collections.binarySearch() implementation. Would this be better, equivalent, or worse? Explain your answer.

(c) Consider two possible ways of adding all of the Entrys to the LookupTable: 1) added in ascending order based on social security number and 2) added in random order.
Of the three implementations described above (ArrayList, LinkedList, balanced binary search tree) which implementation would have the greatest variation in execution time to add all of the tax returns to the LookupTable object? Justify your answer.