1. Test 1
   a. Average 84 (including zeros), 86 (excluding 0s)
   b. Return Test 1 and go over solutions.
   c. Remember that regrades will be handled by the head TA first and can then be appealed up to the professor.

2. Last minute Homework 3 Questions and Concerns

3. Pair Programming
   a. General introduction and overview – http://coweb.cc.gatech.edu/cs1316/188
   b. Advise students to start looking for a partner
      i. Pairs Page – http://coweb.cc.gatech.edu/cs1316/704
      ii. Pair Request Page – http://coweb.cc.gatech.edu/cs1316/705

4. Linked Lists rehashed
   a. PositionedSceneElement, SceneElementPositioned – Pictures placed one after the other.
   b. LayeredSceneElement, SceneElementLayered – Pictures placed one on top of the other.
   c. Introduction to inheritance with SceneElementPositioned and SceneElementLayered
      i. Notice all of the repeated code in PositionedSceneElement and LayeredSceneElement. So instead we create a super class called SceneElement that contains all of the repeated code and have SceneElementPositioned and SceneElementLayered inherit from it.

5. Homework 5
   a. Writing new linked list methods in PositionedSceneElement.
   b. Homework 5 description – http://coweb.cc.gatech.edu/cs1316/633#hw5
   c. Good methods to use
      i. PositionedSceneElement
         1. remove(PositionedSceneElement node) – removes specified node from list and fixes all of the links
         2. getPicture() – returns the node’s Picture
         3. insertAfter(PositionedSceneElement node) – inserts specified node after the node it is called on
         4. copy() – returns a copy of the node it is called on
         5. last() – returns the last node in the list
         6. drawFromMeOn(Picture bg) – draws the Picture from the node it is called on to the end of the list on a specified Picture.
      ii. SimplePicture/ Picture
         1. getFileName() – returns the Picture’s filename
   d. Handling all cases/ possible user error
      i. Remember that the student is the programmer in this course and he or she need to think about all of the possible inputs a user can put into the program.
ii. The more cases you consider the less likely your code will break due to user error.