Objective
In this programming assignment we will experience modeling objects with moving parts and design solutions to detecting object collisions.

Approach
Please refer to the mp4Solution program linked to the course web-page. You have to implement a similar program where the user can insert your hero objects into your scene. Your hero object must have a base with at least two (2) independently movable/scalable/rotate-able parts. Here are the technical specifications for the design of your hero object:

1. Your hero object must be movable/scalable/rotate-able.
2. The two independent parts must be physically connected to the base at all time.
3. When the entire hero object is transformed, the two independent parts must honor the transformation requests and change their geometric shapes/positions accordingly.
4. However, when the user transforms one of the independent parts, the base of the hero object and the other independent part must not be changed in anyway.

For example, in the FireTruck example we worked on in class, the truck body was the base, and the ladder was one (1) movable/scalable/rotate-able part. In this mp, you must have two such parts. You must also design a friendly graphical user interface to support all of the above transformation operations.

The following functionality must be supported on your mouse button:

<table>
<thead>
<tr>
<th>Mouse Events</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left / Down</td>
<td>Draws a new hero object at the mouse position</td>
</tr>
<tr>
<td>Left / Drag</td>
<td>Drags the object (this satisfies the movable requirement of the hero object)</td>
</tr>
<tr>
<td>Left / Up</td>
<td>If Collision, delete the newly created object.</td>
</tr>
</tbody>
</table>

When the left mouse button (LMB) comes up, you must determine if your hero object is colliding with any existing objects. Here we defined colliding as when the bounding boxes of the base of two different hero objects overlap. Note that the bounding box of a rotated rectangle is an axis-aligned rectangle. You do not have to worry about computing the exact tight bounding area of your hero object with the independent parts, just the bounding box on the base will be sufficient. If collision is detected, you must:

1. Highlight the existing object that collided with your new object.
2. Inform the user about the collision and remove the newly created object from your system.
3. The Highlighted object should return to normal the next time your user clicks on the LMB.
Credit Distribution

Here is how the credits are distributed in this assignment:

1. Hero Object 50%
   a. Move/Scale/Rotate on the entire object 10%
   b. Independent Move/Scale/Rotate part 1 15%
   c. Independent Move/Scale/Rotate part 2 25%

2. Proper GUI design/support of the above operations 10%

3. Proper Collision Detection 35%
   a. Correct collision detection 25%
   b. Proper highlight behavior of colliding object 10%

4. Others 5%
   a. General program efficiency/style
   b. Correct/usable zip submission

Extra Credit:

Here are a couple of ideas:

1. Compute the bounding box of your entire hero object with the two independent parts for collision detection (5%), or

2. Support a Run away mode where each object starts to move with a unique and random velocity. However, in this case, no objects can collide with each other. You must detect collision and bounce objects away from each other. Run away mode ends when all objects run off the screen area (10%).

Come talk to me if you have other ideas, we can figure out how much they worth.

WARNING: Both of the extra credit ideas are quite involved, please work on it only after you have completed the basic assignment. Collision detection can be quite confusing. You should reserve sometime for that. If your hero object looks even remotely similar to the FireTruck, you will receive zero credit for it. I have found two bugs in the T2dBound() class I gave you. Do not trust that class (in general do not trust my code, for that matters, do not trust me), make sure you understand what that class is doing and you should verify it is doing what you think it is doing.

This programming assignment will count 12% towards your final grade for this class.