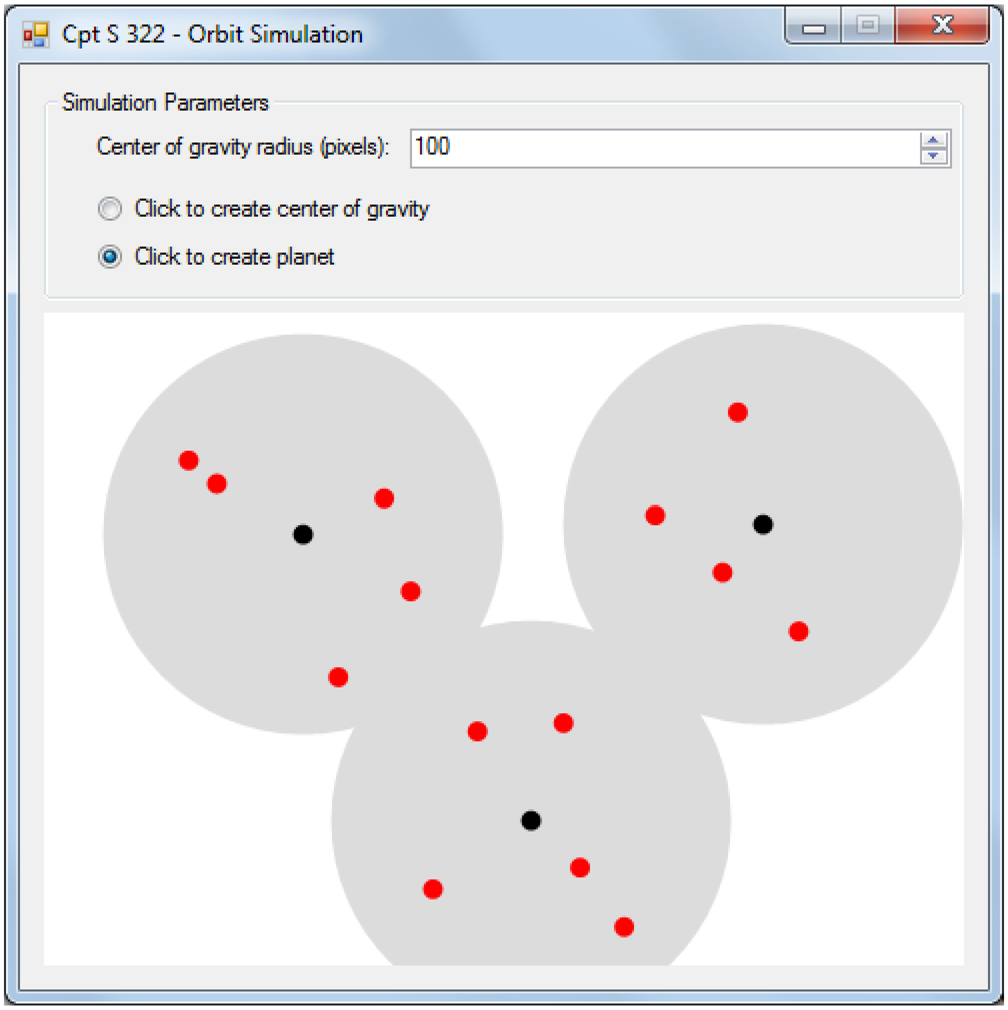
Orbiting Planets Animation
Cpt S 321 Homework Assignment
Washington State University

Submission Instructions: (see syllabus)

Assignment Instructions:

In this homework you'll create a simple animation of planets orbiting around centers of gravity. The physics in this simulation are far from realistic, although you can take the time to make more realistic simulations if you wish.



Simulation Parameters

Center of gravity radius (pixels): 100

Click to create center of gravity

[✓ Click to create planet]

Create a new WinForms application and put a picture box in the interface, taking up most of the space. Your simulation will allow you to create both centers of gravity and planets, so also leave room for radio buttons or some other way to choose what clicking in the picture box will create. Alternatively, you can refrain from creating interface elements for these options and just make right-clicks create planets and left-clicks create centers of gravity. Include a readme.txt file in your code submission and/or a help option in the interface that tells the user how to create the objects if you do this.

Requirements:

1. You must be able to create a center of gravity with a particular, finite radius. You can hard-code a constant radius value if you like, just make sure it's not so large that it fills up the entire simulation area. Note in the demo app in the screenshot above that a faint gray circle is being rendering to indicate the influence area of each center of gravity (black dots). The demo app has a **NumericUpDown** control that lets you configure the influence radius for each individual center of gravity, but this is not required.
2. You must be able to create planets. Planet size can also be hard coded and you don't have to simulate mass. All a planet really needs is an (x,y) location. This location will be updated every time you render a frame in your animation sequence based on which centers of gravity the planet is within range of.
3. You must render at least 30 times per second to produce a smooth animation. Use a **Timer** control and set its interval (which is in milliseconds) such that you get its "Tick" event to execute at least 30 times per second. In my example I'm using an interval of 20 for about 50 frames per second. Don't forget to set the "Enabled" property to true (either at design time or runtime) to make sure the timer calls the function at every interval.
4. In the timer-tick event do the following:
   - Update the position of each planet based on what center(s) of gravity they're within range of
   - Render each center of gravity (a small black dot will suffice, a light region showing the influence area is also required)
   - Render each planet (a small circle is fine, but use a color different from black)

Notes

- Render using the **System.Drawing.Graphics** class functionality (or if you *really* want you can do the pixel-processing on your own, it's just not recommended).
- You have to rotate the planets around centers of gravity. You may have to look online if you don't remember the basic rotation math to accomplish this.

- Like all homework assignments, you have 1 week to complete this assignment. If you get stuck on something, seek help. The assignment will take a long time if you don't have a good idea of what you're doing.
- You don't have to decouple the logic and UI nor do you need to create a separate logic engine for this assignment. This is just a basic demonstration of simple graphics so it can all be in the form code.
- A brief demo of the app will be shown in class to give you an overview of what it should look like. Attend lecture to make sure that you get to see this demo.

Point breakdown:

- **2 points for correct input processing and creation of objects at the appropriate locations when the user clicks in the picture box**
- **3 points for correct rendering of all planets and centers of gravity (both location and influence area)**
- **4 points for having planets correctly rotating around the center of gravity that they're within range of and having the general animation working correctly**
- **1 point for misc. items such as commenting and coding style**
- **----------**
- **10 points total**