COLBY COLLEGE
Department of Mathematics
and Computer Science

CS115 **Allen B. Downey**
Spring 1998

Assignment 4: Snowpeople!

Due: Wednesday 4 March

The purpose of this assignment is to practice using methods as a way of organizing and encapsulating complex tasks.

WARNING: It is very important that you take this assignment one step at a time, and get each step working correctly before you proceed. More importantly, make sure you understand each step before you proceed.

Make a new project

1. Create a new project named Snowperson.π, and make it an Applet. If you need more detailed instructions, refer to the Olympics lab handout.

2. If you are using your solution to the Olympics problem, you will want to take your code and wrap it up in a method named paintRings, as I did in my solution. The arguments to paintRings should be the Graphics object g, and four ints that specify the bounding box.

3. If you are using my solution, you can pick it up from the General Server.

   (a) Select Chooser from the Apple Menu. Click on the AppleShare icon. Select COLBY from the list of AppleTalk Zones. There should be three machines in that zone, listed under the words "Select a file server." Double-click on the General Sever.

   (b) A window will appear asking you to register. Click the Guest button and OK. An icon representing the General Server should appear on the desktop.

   (c) Close the chooser and double-click on the new icon. Open the folder named "Course materials." Then open the folders named "CS," "CS115," and "Java Examples." You should see all the sample code I have passed out so far, including Olympics.java.

4. Cut the paintRings method from Olympics.java and paste it into Snowperson.java. Test your program extensively before you proceed.

Make a snowperson

1. Create a new method named paintSnowperson that takes the same arguments as paintRings. Instead of drawing Olympic rings, though, it should draw three ovals, stacked on top of each other like a snowperson.

2. Modify the main method so that it determines the size of the window and calls the paintSnowperson method. The bounding box of the snowperson should be the same height as the window, but only one-third of the width, and it should be centered.

3. As you resize the screen, your program should redraw the snowperson so that it touches the top and bottom of the screen, and the three ovals touch, and the proportion of the three ovals should stay the same.

Draw the snowperson *American Gothic*

1. Modify your main method so that it draws two snowpeople side by side. One of them should be the same height as the window; the other should be 90% of the window height. (Snowpeople exhibit a sexual dimorphism in which females are roughly 10% smaller than males.)

   Their widths should be proportional to their heights, and their bounding boxes should be adjacent (which means that the drawn ovals probably will not touch). The pair should be centered, meaning that there is the same amount of space on each side.

Send the snowpeople to the Olympics

1. Modify the paintSnowperson method so that it invokes the paintRings method in order to emblazon the olympic rings on the chest (middle oval) of each snowperson. Try to make the rings fit inside the chest oval.

With a corn-cob pipe...

1. Write a method called paintFace that takes the same arguments as paintSnowperson, and that draws a simple face within the given bounding box. Two eyes will suffice, but you can get as elaborate as you like.

   As always, the proportions of the face should be the same, regardless of how big the bounding box is.

2. Modify the paintSnowperson method so that it invokes paintFace in order to fill in the top oval (the face).

Make Mrs. Snowperson pregnant

1. I realize that this is a little risqué, but there is a point. Modify the main method so that it draws a baby snowperson inside the abdomen oval of Mrs. Snowperson.

   Notice that adding something to main affects just one snowperson. Adding something to paintSnowperson affects all snowpeople.

Make all snowpeople pregnant

Ok, now let's imagine that instead of making a particular snowperson pregnant, we want to make all snowpeople pregnant. Instead of adding a line to main, we would add a line to paintSnowperson. What would that line do? It would invoke paintSnowperson! Can you do that? Can you invoke a method from within itself? Well, yes, you can, but you have to be careful.

SO DON'T DO THIS YET!!!

Think for a minute. What if we draw a big snowperson, and then put a small snowperson in the abdomen oval. Then we have to put an even smaller snowperson in the small abdomen oval, and so on and so on. The problem is, it would never stop. We would be drawing smaller and smaller snowpeople until doomsday!

One solution is to pick a minimum snowperson size, and say that below that size, we refuse to draw any more snowpeople.

1. Add a line at the beginning of paintSnowperson that checks the height of the bounding box and returns immediately if it is less than 10.

   ```
   if (height < 10) return;
   ```

2. Now that you have that line, it is safe to add code at the end of the method so that after drawing the ovals, and the face, and the Olympic rings on the chest, it invokes paintSnowperson to put a small snowperson in the abdomen.

What to turn in

As usual, print out the final version of your program, write your name in the upper right corner, and turn it in!