1. [SmileyFaceClient.java](file:///C:\\Users\\Doug\\Documents\\a-oprfhs\\10-11\\AP%20Computer%20Science\\Unit%206\\0809\\Assignment%206C\\SmileyFaceClient.java) is a GUI that will serve as a shell for you to do this program. Put all your code in the display() method. Watch the video to see what the final product should look like.
2. Create an array of SmileyFace object references. It should be the exact size to hold SmileyFace objects, each with a 20 pixel diameter, that are adjacent to each other and fill up the width of the screen. Leave a small buffer at each end. So the width you are trying to fill, in pixels, can be **getWidth()-20**. Divide that by the width of each SmileyFace in pixels (20), and you will have the right size for the array. Note: we don't have any SmileyFace objects at this point, all we have is an array of null object references.
3. Fill up each index in the array with a SmileyFace object. The specs for each SmileyFace are:
   1. Radius 10
   2. The x position for the first one is 20, and the x position for each one after that is 20 more than the one before it.
   3. The y position for each one is half the screen height: **getHeight()/2.**

Pause briefly after creating each SmileyFace object to give an animated effect (about 25 thousanths of a second):

**try {Thread.sleep(25);}**

**catch (InterruptedException e){}**

1. Pause for about a half-second after all the SmileyFace objects are created – sleep(500)
2. Pick a random SmileyFace from the array, and change it's color to a random color. Repeat this 1000 times.
   1. The SmileyFace class doesn't provide a way to have it's color changed. Revise the SmileyFace class to provide this service by adding a method. It should receive a Color object as a parameter, change the color instance variable for the SmileyFace, and then call the draw() method. Your client can now ask a SmileyFace object to change its color.
   2. All objects in the array should have a chance of being picked.
   3. Color components for red,green,blue should be in the full spectrum of 0-255.
   4. Don't sleep while changing the colors.
3. Pause for about a half-second after changing all the colors.
4. Clear the objects off the screen simultaneously from both ends to create a contracting accordion effect.
   1. I'll let you think about an algorithm to do that. If you get stuck, ask me.
   2. Sleep 25 milliseconds after clearing a pair of SmileyFaces from each end. (i.e. clear two, pause, clear two, pause, …)
   3. The SmileyFace class doesn't provide a way for a client to clear it off the screen. There is a clear() method, but it's private and your client can't call it. Make that public so your client will be able to call it.
5. Pause for about a half-second after clearing the objects. Note: they are still in the array. The objects still exist. They are just not displayed on the screen right now.
6. Change all the objects to a shade of green (or blue, or red if you like), starting from the middle and extending outward simultaneously in both directions. This will create an expanding accoridan effect.
   1. If you have an algorithm for the other one, this one will be easy.
   2. The green component (or red, or blue) should be a random in the range of 25-255 (i.e. don't pick colors between 0 and 24, they're too faint). The other two color components should be zero.
   3. Sleep 25 milliseconds after coloring one from each end. (i.e. color two, pause, color two, pause, …)
7. Visually pop out two SmileyFace objects, the one with the most green, and the one with the least green (or Red, or Blue if you went that route.)
   1. Scan through the entire array, find the SmileyFace with the most green (or Red, or Blue), and the SmileyFace with the least grean (or Red, or Blue).
   2. To evaluate how much green a SmileyFace has, you will have to ask it for its color object. There is no way for a client to ask a SmileyFace for its color. You will have to provide that service by writing a method in SmileyFace to return the color. The client will ask the SmileyFace for its color object, and then ask the the color object for its green component; **.getGreen()**
   3. Once you find these two objects, set their radii to 40 or so, so they pop out on the screen.