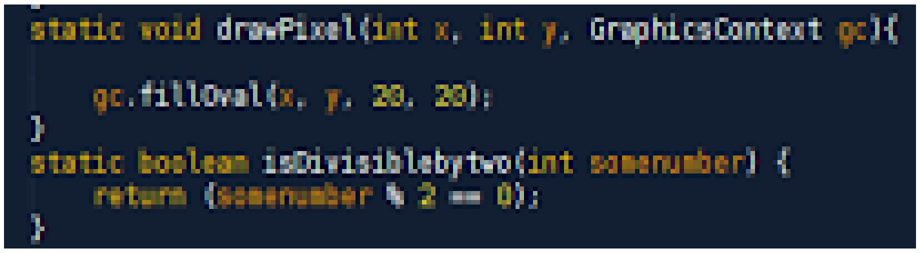
This is the essay for Carter Cook's (the 13373$7)(P. 7) code art. What makes this program different from my classmates' is that mine builds upon our last project. On our previous project, we had an output. I realized that instead of outputting a word or phrase, it outputs a circle. Just one diagonal column of multi-colored circles, until very recently when I finally got a good idea for where to go. I set it up so that all of the numbers from 0 to 1000 filter through the Challenge0 project and put different colored circles down a diagonal line according to whichever group it fit into (Team Building, Animate This!, etc) and formed a pattern. That was the point in which I was stuck for a while. What ended up happening to it was I made another diagonal line, the same as the one I made before, but from the upper right hand corner to the bottom left hand corner, rather than upper left to bottom right. Also, it has gradients going in a cross behind it, which is just a little rgb coloring trick. Finally, I added a black background to make it just look better overall. Basically, I expanded on ideas I got from our last project, which is unlike everyone else's.

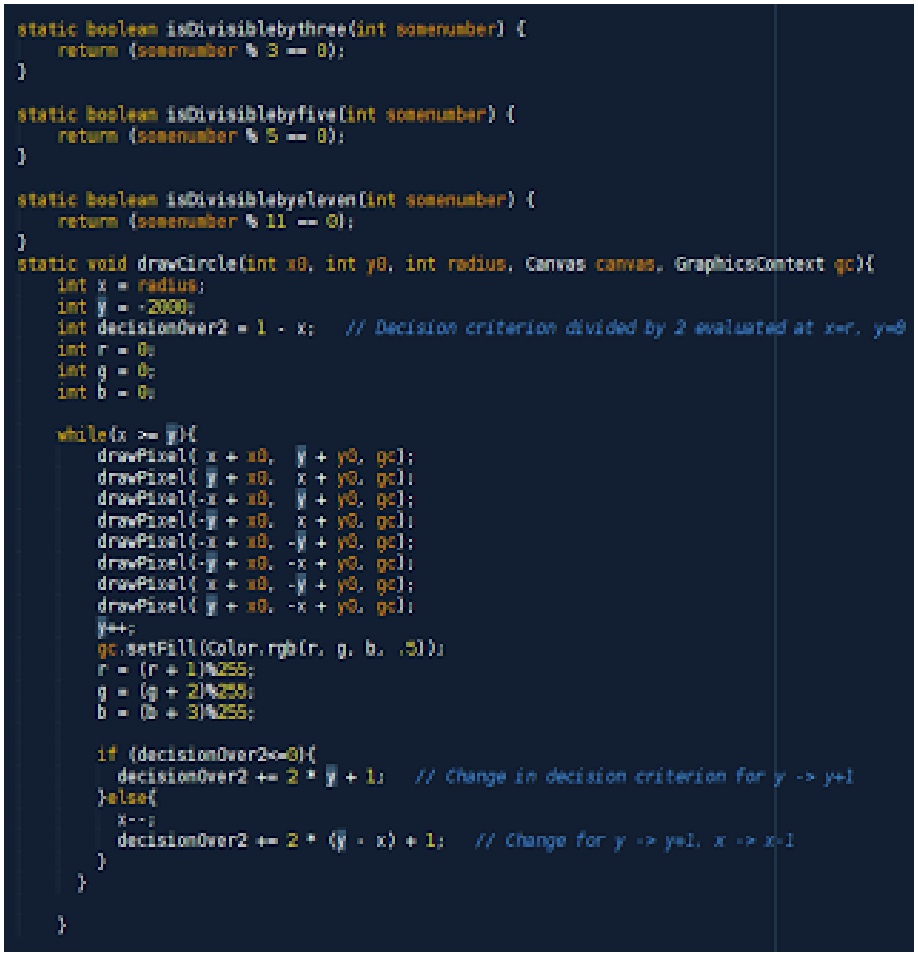
My project also incorporates/demonstrates encapsulation as well as other parts of object oriented programming. Basically, it keeps all the parts of my code in certain places and certain parts can only be accessed inside one of these groups. It is protected by this kind of capsule that it is contained by. It also has a private method, which is protected from other applications because it can only be read inside of this one file.

I have several methods, and here are 6 of the big ones:



```java
static void drawPixel(int x, int y, GraphicsContext gc){

    gc.fillOval(x, y, 20, 20);
}
static boolean isDivisiblebytwo(int somenumber) {
    return (somenumber % 2 == 0);
}
```



```java
static boolean isDivisiblebythree(int somenumber) {
    return (somenumber % 3 == 0);
}

static boolean isDivisiblebyfive(int somenumber) {
    return (somenumber % 5 == 0);
}

static boolean isDivisiblebyeleven(int somenumber) {
    return (somenumber % 11 == 0);
}
static void drawCircle(int x0, int y0, int radius, Canvas canvas, GraphicsContext gc){
    int x = radius;
    int y = -2000;
    int decisionOver2 = 1 - x;   // Decision criterion divided by 2 evaluated at x=r, y=0
    int r = 0;
    int g = 0;
    int b = 0;

    while(x >= y){
        drawPixel( x + x0,  y + y0, gc);
        drawPixel( y + x0,  x + y0, gc);
        drawPixel(-x + x0,  y + y0, gc);
        drawPixel(-y + x0,  x + y0, gc);
        drawPixel(-x + x0, -y + y0, gc);
        drawPixel(-y + x0, -x + y0, gc);
        drawPixel( x + x0, -y + y0, gc);
        drawPixel( y + x0, -x + y0, gc);
        y++;
        gc.setFill(Color.rgb(r, g, b, .5));
        r = (r + 1)%255;
        g = (g + 2)%255;
        b = (b + 3)%255;

        if (decisionOver2<=0){
          decisionOver2 += 2 * y + 1;   // Change in decision criterion for y -> y+1
        }else{
          x--;
          decisionOver2 += 2 * (y - x) + 1;   // Change for y -> y+1, x -> x-1
        }
      }

    }
```

The first one that I have in the list is *drawPixel()*. This is a method that I call later, but by itself it draws a dot, 20 pixels by 20 pixels. The next 4 come originally from Challenge0, and they are the ones used to decide what each number is divisible by. Those are important above in my main code segment that call those for each number that is tested to determine the output. The last one is pretty interesting. It uses math to do a variety of things. For my purposes, it creates a plus sign of the circles, which it uses *drawPixel()* for. One of the coolest things about it, though, is the colors that define how these circles look. I set up a simple additive to the color to add to the RGB values. This accidentally (but wonderfully) turned the lines into a nice series of gradients.

The values passed to my methods are, for the most part, commonly used. For example, the one used in the *isDivisibleby* methods is *somenumber*, allowing for different numbers to be passed to it, from 0 to 1000 defined by my for loop in the main part of my code. Another common one was *GraphicsContext gc*, for the obvious that we have graphics that are being created. A couple others are *x* and *y*, allowing a certain place on the canvas to be defined for whatever is being done.

Returning values are somewhat easier to explain than given values simply because there are less of them and they are less complex. One of the ones that is critical is a True/False return from my booleans that help my Challenge0 code function without having to redefine these terms every time, basically. Instead, they just run through *isDivisibleby* and get True/False to save time. That is mainly it, the one other thing being the output of a circle (*drawPixel*).

The methods that are called in *main()* are all of them. I only have one class that I needed, and that is my main class.

The access modifiers that I have are mostly public, as I don't have need for this specific project to protect my methods. However, I do have one that just happened to be there, which is the method that makes the background black.

Class constructors aren't called by my program, as I didn't set any default settings for classes. Also, I only trans siberian orchestrahave the one main class :P.

Overall, this project was a lot of fun. I was pretty lost at first, to be honest, and had no direction. Then I started messing around with my code a lot and started learning how things worked. I got out of my “coding block" as you could call it, and made something that was completely different from my plan, but really nice-looking. I hope we can continue having enough time to discover really awesome things, and just hope that this project ends up looking to other people just as good as it looks to me.