**LARSON—MATH 255–CLASSROOM WORKSHEET 27**
**More Interacts!**

1. (a) Start the Chrome browser.
   (b) Go to `http://cocalc.com`
   (c) You should see an existing Project for our class. Click on that.
   (d) Click "New", then "Sage Worksheet", then call it **c27**.
   (e) For each problem number, label it in theSAGE cell where the work is. So for Problem 1, the first line of the cell should be `#Problem 1`.

**The Birthday Problem**

2. How many students do we need in a classroom so that there is a better than 50% chance that at least one pair of them have the same birthday (Month & Day)?. What could you code to investigate this problem?

   There is a collection of examples of Sage INTERACTS at `http://wiki.sagemath.org/interact/`. Let's look at a few of these examples to see the kinds of things you can do with Sage.

**Posets**

3. A *poset* is a very common mathematical object. They consist of a set together with a *relation* that is reflexive, transitive, and anti-symmetric. Any collection of lists or sets with the subset relation form a poset. Try `P=Poset([[1,2],[],[1]])`. This makes a Poset $P$ consisting of 3 lists. You can get a nice picture (called a *Hasse diagram*) of this poset with the command `P.show()`.

4. Consider the list of integers `L=[5..10]`. Ordinary inequality defines a relation on $L$. So $(a, b)$ is in the relation if and only if $a \leq b$. Evaluate:
   `Q=Poset(([5..10], lambda x, y:  x<=y))`. Then `show` it.

5. Can you think of another relation on the positive integers? How about "$\geq$". Experiment that—and make a picture.

6. The positive integers together with the relation $R$ where a pair $(a, b)$ is in $R$ if and only if $a$ divides $b$ is a relation. So, for instance, $(1, 5)$ is in $R$ as 1 divides 5 and $(2, 4)$ is in $R$ as 2 divides 4. Here's a Sage INTERACT that makes a nice picture (called a *Hasse diagram*) of the positive integers with the divisibility relation.

```
@interact
def _(n=(5..100)):
    Poset(([1..n], lambda x, y: y%x == 0) ).show()
```

## Julia Set

You can read about the Julia set at `http://en.wikipedia.org/wiki/Julia_set`. It is a common example of *complex dynamics* that can be illustrated with cool pictures.

7. First recall that the complex numbers have the form $a + bi$ where $i = \sqrt{-1}$. Let `z=5+3*i`. You can think of this as the vector which points from the origin to $(5, 3)$.

8. $z$ has a length (or *magnitude*. Find it with `norm(z)`. Use `n(_)` to get a numerical approximation. Similarly, every point in the plane can be viewed as a complex number, and with an associated magnitude.

9. The secret to the Julia Set INTERACT is the function `complex_plot()` which associates colors to values of a complex function. Consider the function $f(z) = z$ where $z$ is a complex number. Make a pretty plot of this using `complex_plot(f, (-5, 5), (-5, 5))`. Remember to tell Sage that $z$ is a variable.

10. Here's the description you get if you evaluate `help(complex_plot)`: " 'complex_plot' takes a complex function of one variable, f(z). The magnitude of the output is indicated by the brightness (with zero being black and infinity being white) while the argument is represented by the hue (with red being positive real, and increasing through orange, yellow, ... as the argument increases)." Now try `complex_plot(z**2, (-5, 5), (-5, 5))`. Can you explain the lines in the picture?

11. Most of the following code is defining the input numbers and various tricks for fast display of the result. The main bits of code are the definition of $f(z)$ and the call to `complex_plot`.

```
@interact
def julia_plot(expo = slider(-10,10,0.1,2),
      iterations=slider(1,100,1,30),
      c_real = slider(-2,2,0.01,0.5),
      c_imag = slider(-2,2,0.01,0.5),
      zoom_x = range_slider(-2,2,0.01,(-1.5,1.5)),
      zoom_y = range_slider(-2,2,0.01,(-1.5,1.5))):
    var('z')
    I = CDF.gen()
    f(z) = z^expo + c_real + c_imag*I
    ff_j = fast_callable(f, vars=[z], domain=CDF)

    def julia(z):
      for i in range(iterations):
         z = ff_j(z)
         if abs(z) > 2:
            return z
      return z
    print("z <- z^{} + ({} + {})*I".format(expo, c_real, c_imag))

    complex_plot(julia, zoom_x,zoom_y, plot_points=200, dpi=150).show(frame=T
```

This last line should end with `.show(frame=True, aspect_ratio=1)` Try different input values for this set. If you think this is cool, maybe you might be interested in a course in *Discrete Dynamical Systems*!

**Random Walks Interact**

12. The following Sage INTERACT starts with the origin $(0,0,0)$ and "walks" $u$ ($u \in (-0.5, 0.5)$) units in each of the $x$, $y$ and $z$ directions, repeats this up to $1,000$ times, keeps track of all of the points that are visited and then draws all of the points, with lines from one point to the next.

```
@interact
def rwalk3d(n=slider(50,1000,step_size=1), frame=True):
    pnt = [0,0,0]
    v = [copy(pnt)]
    for i in range(n):
        pnt[0] += random()-0.5
        pnt[1] += random()-0.5
        pnt[2] += random()-0.5
        v.append(copy(pnt))
    show(line3d(v,color="black"),aspect_ratio=[1,1,1],frame=frame)
```

## Getting your classwork recorded

When you are done, before you leave class...

1. Click the "Make pdf" (Adobe symbol) icon and make a pdf of this worksheet. (If CoCalc hangs, click the printer icon, then "Open", then print or make a pdf using your browser).
2. Send me an email with an informative header like "Math 255 - c27 worksheet attached" (so that it will be properly recorded).
3. Remember to attach today's classroom worksheet!