1. Log in to your Sage/CoCalc account.
   
   (a) Start the Chrome browser.
   (b) Go to http://cocalc.com and sign in.
   (c) You should see an existing Project for our class. Click on that.
   (d) Click “New”, call it c24, then click “Sage Worksheet”.
   (e) For each problem number, label it in the Sage cell where the work is. So for Problem 1, the first line of the cell should be #Problem 1.
   (f) When you are finished with the worksheet, click “make pdf”, email me the pdf (at clarson@vcu.edu, with a header that says Math 255 c24 worksheet attached).

2. **Problem** (Ramanujan) 2, 9, 16, etc. can be written (uniquely) as the sum of 2 cubes (\(1^3 + 1^3, 1^3 + 2^3, 2^3 + 2^3\), etc.). Find the smallest integer which can be written as the sum of 2 cubes in 2 different ways.

**Our Own Class.** We started designing our own class to get a feeling for the main ideas.

```python
class Character:
    def __init__(self, name):
        self.name = name
        self.intelligence = randint(1,10)
        self.health = randint(1,10)
        self.strength = randint(1,10)
        self.fortitude = randint(1,10)

    def hello(self):
        print "Hello world! I am {}.".format(self.name)

    def status(self):
        print "My intelligence is {}".format(self.intelligence)
        print "My health is {}".format(self.health)
        print "My strength is {}".format(self.strength)
        print "My fortitude is {}".format(self.fortitude)

    def change_intelligence(self, amount):
        new = self.intelligence + amount
        if new < 1:
            self.intelligence = 1
        elif new > 10:
            self.intelligence = 10
        else:
            self.intelligence = new
```
We'll design a general class of Dungeons and Dragons character, sample concrete character objects, methods that can be accessed by any character objects, and functions that can be used on the characters. We'll build on the last worksheet.

3. Evaluate. Let \(c5=\text{Character("Gandalf")}\). This creates an object of the \text{Character} type. The name form the program environment’s point of view is “c5”. The \text{.name} built-in to the class is “Gandalf”—but that’s not useable for our programs—this is data that’s stored as part of the created object.

4. Things may happen to our characters. Gandalf may drink a potion that effects his intelligence. Let’s add a method so we can change a character’s initial intelligence. We must be careful never to leave the range of 1 to 10.

5. Now define the following function.

\[
\text{def drink_potion(character):
    if random() < 0.5:
        character.change_intelligence(3)
        print "I feel smarter!"
    else:
        character.change_intelligence(-3)
        print "Uh oh!"
}
\]

Try \(c5\.status()\), then \(\text{drink_potion(c5)}\), then \(c5\.status()\) again.

6. Perhaps we should award our characters “points” in certain situations? We can add a \text{points} value when we initialize the character. And also add it to our status reports. And there should be a way to change the number of points. So lets add a \text{change_points()} method to the \text{Character} class. We’ll have to update the \text{status} method at the same time.

\[
\text{def change_points(self, amount):
    self.points = self.points + amount}
\]

Evaluate. Let \(c6=\text{Character("LittleJohn")}\). Then try \(c6\.status()\).

7. Our characters may have to fight trolls. Define the following function.

\[
\text{def fight_troll(character):
    if character.health > 5 and character.strength > 5:
        character.change_points(5)
        print "I have defeated the troll!"
    elif character.health < 4 or character.strength < 4:
        character.change_points(-5)
        print "You have defeated me this time!"
    else:
        print "Run away!"
}
\]
8. Oh oh. LittleJohn has encountered a troll. Lets see what happens. Evaluate `fight_troll(c6)`. Then check his status with `c6.status()`.

You see these classes, objects and methods can get very interesting!

The Birthday Problem.

9. (Guess) 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)?

Here’s 3 basic and useful facts:

- **Multiplication Principle** (Burgers & Drinks). If there are \( m \) choices for one thing and \( n \) choices of another thing, there are \( m \cdot n \) ways to choose one from the first things and one from the second things.

- (Assuming all outcomes are equally likely) the probability of something happening is:

\[
\frac{\text{The number of good outcomes}}{\text{The total number of outcomes}}
\]

- **Opposite Probabilities.** The probability of an event happening = 1 - the probability it doesn’t happen.

10. Start by simplifying the problem. What is the probability that 2 students have the same birthday? This can be calculated directly: think of the students (or more clearly their birthdays) as \( S_1 \) and \( S_2 \). So the possible pairs of birthdays can be represented as \( (S_1, S_2) \). If we assume a year always has 365 days, how many “good” pairs are there (pairs where the birthdays are the same). How many total pairs are there? What’s the probability?

11. Now what’s what is the probability that if there are 3 students at least 2 will have the same birthday? (This is harder and needs a new idea).

12. Now what’s what is the probability that if there are 4 students at least 2 will have the same birthday? If you get this you may see the pattern and write code to figure out an answer for any number \( n \) of students.

13. Then keep repeating. What’s the smallest \( n \) where the probability is finally .5 or 50%. (Write a `while` loop with an appropriate stopping condition or a `for` loop. We know that we will have guaranteed certainty when \( k = 365 \)).

14. How many students do we need to have a 90% chance that at least 2 students share the same birthday?