LARSON—MATH 255–CLASSROOM WORKSHEET 31
Graph Theory—Classes—Objects—Methods

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 c31, then click “Sage Worksheet”.

Our Own Class

We started designing our own class to get a feeling for the main ideas. 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.

2. 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.

```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 %s." % (self.name)
    def status(self):
        print "My intelligence is %s" % (self.intelligence)
        print "My health is %s" % (self.health)
        print "My strength is %s" % (self.strength)
        print "My fortitude is %s" % (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
```

Evaluate. Let c5=Character("Gandalf").
3. Now define the following function.

```python
def drink_potion(character):
    if random() < .5:
        character.change_intelligence(3)
        print "I feel smarter!"
    else:
        character.change_intelligence(-3)
        print "Uh oh!"
```

Try `c5.status()`, then `drink_potion(c5)`, then `c5.status()` again.

4. Perhaps we should award our characters “points” in certain situations? We can add a `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 let’s add a `change_points()` method to the `Character` class.

```python
def change_points(self, amount):
    self.points = self.points + amount
```

Evaluate. Let `c6=Character("LittleJohn")`. Then try `c6.status()`.

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

```python
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!"
```

6. Oh oh. LittleJohn has encountered a troll. Let’s 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!

7. (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.