Practice Final Solution

1.

```
class IceCream:
     def __init__(self):
          self.cone = {}

     def add_scoops(self, flavor, num_of_scoops):
          if flavor in self.cone.keys():
               self.cone[flavor] += num_of_scoops
          else:
               self.cone[flavor] = num_of_scoops

     def get_flavor(self, flavor):
          if flavor in self.cone:
               return self.cone[flavor]
          else:
               return 0

     def to_string(self):
          scoops = self.cone.values()
          total = sum(scoops)
          return str(total) + ‘ scoops of ice cream with ‘ +
                 str(self.cone.keys())
```

2.

1.

```
new_ingredients = {}
for i in self.ingredients:
     new_ingredients[i] = self.ingredients[i] * num_servings
return new_ingredients
```

2.

```
breakfast = Recipe('Cereal')
breakfast.add_ingredient('milk', 1)
breakfast.add_ingredient('cereal', 2)
breakfast.add_next_step('Pour the cereal')
breakfast.add_next_step('Pour the milk')
```

3.

```
def swap_casing(phrase):
  result = ""
  for i in range(len(phrase)):
    if i % 2 == 0:
      result += phrase[i].upper()
    else:
      result += phrase[i].lower()
  return result
```

4.

```
def even_key(given_dict):
     ans_list = []
     for cur_key in given_dict:
          if(cur_key % 2 == 0):
               ans_list.append(given_dict[cur_key])
               given_dict[cur_key] = "even"
     return ans_list
```

5. Answer:

a. Global, do_stuff, recommend_by_influence
b. recommend_by_influence
c. Most likely this is due to a misspelling of the function name referred to as "read_result()" on line 107 of social_network.py. So a good start would be to search to see if there is a similarly named function in the file social_network.py. If that fails, maybe this function is defined in another namespace like we did before with Random or nx, requiring the function name to be prefaced with that module name.

6. sum : 9

7.
```
'''
Given a list of numbers, print the number of unique numbers
in the list and return a dictionary containing the numbers in
the list as keys, and values that are a set containing all of
the factors of that number.

'''
```

8.

a.

```
[sorted(x, reverse=True) for x in lst if len(x) % 2 != 0]
```

b.

```
{sum(x): x for x in lst}
```

9.
```
[Dog(name) for name in dog_names if len(name) > 0]
```