Phone Number Example - phonebook.py

```
class PhoneNumber:
     """
     A PhoneNumber represents a single phone number made up of the
     area code,exchange, and the line number.

     EX: (415) 552-7909
          ^    ^   ^
          |    |   |
          |    |   number
          |    exchange
          area code
     """

     def __init__(self, area_code, exchange, number):
          """
          Creates a new PhoneNumber from the provided area
          code,exchange and number.
          """
          self.area_code = area_code
          self.exchange = exchange
          self.number = number

     def call(self):
          """
          Calls this PhoneNumber.
          """
          print("Calling (" + str(self.area_code) + ") " +
          str(self.exchange) + "-" + str(self.number))
          print("ring... ring... Hello?")

     def print_number(self):
          """
          Prints a pretty version of this PhoneNumber.
          """
          print("(" + str(self.area_code) + ") " +
          str(self.exchange) \ + "-" + str(self.number))

class PhoneBook:
     """
     A PhoneBook is a collection of names and phone numbers.
     """
     def __init__(self):
          """
          Creates a new PhoneBook that is initially empty.
          """
          self.contacts = {}
```

```
    def add_number(self, name, phone_number):
        """
        Adds the provided name and PhoneNumber to this PhoneBook.
        Will replace the number if the name already exists in this
        PhoneBook.
        """
        self.contacts[name] = phone_number

    def delete_contact(self, name):
        """
        Removes the provided name and the associated PhoneNumber
        from this PhoneBook.
        """
        # This is how to remove from a dict. We might not have
        used this before.
        del self.contacts[name]

    def call(self, name):
        """
        Calls the phone number associated with the provided name.
        """
        self.contacts[name].call()
        print("Hi this is " + name + ".")

    def get_phone_number(self, name):
        """
        Returns the PhoneNumber associated with the provided name.
        """
        return self.contacts[name]

    def get_contacts_in_area_code(self, area_code):
        """
        Returns a list of all PhoneNumbers in this PhoneBook that
have
        the given area_code.
        """
        result = []
        for name in self.contacts:
            num = self.contacts[name]
            if num.area_code == area_code:
                result.append(num)

        return result
```

Phone Number Example - phonebook-client.py

```
from phonebook import *

# Make some new phone numbers
num1 = PhoneNumber(916, 272, 8010)
num2 = PhoneNumber(916, 274, 2805)
num3 = PhoneNumber(415, 552, 7909)

# Try printing them
num1.print_number()
num2.print_number()
# print(num1)
# print(num2)

# Try calling
num1.call()

# Make a new phone book
my_contacts = PhoneBook()

# add some contacts
my_contacts.add_number("Nick", num1)
my_contacts.add_number("Justin", num2)

# try calling the contacts
my_contacts.call("Nick")
my_contacts.call("Justin")

# experiment with getting the phone number from the phone book
num4 = my_contacts.get_phone_number("Justin")

print(num1 == num4)
print(num2 == num4)

numbers = my_contacts.get_contacts_in_area_code(916)
for num in numbers:
    num.print_number()
    # print(num)
```

CSE 160 Section 10 Problems

1. We want to modify the `PhoneBook` class so that a person can have more than one phone number.

   a) Currently, a `PhoneBook`'s contacts are in a dictionary that maps each person's name to a `PhoneNumber`. What data structure can we use to store multiple phone numbers per person? Describe the data structure, or combination of data structures, you would use.

   b) How would you rewrite the constructor method to implement the way of storing contacts you described in part a? You may not need to rewrite it at all, depending on your answer to part a.

   c) If we're going to make this change, we'll need to modify most of the other methods in `PhoneBook`. Start by rewriting the `add_number` method so that each person can have more than one `PhoneNumber`. (Hint: What two cases do you need to handle, now that the name might already be a contact?)

2. For the following code, write its output. If there is an error, describe the error and the cause, and include the output up until the error.

```
def histogram(words, stop_words=[]):
    """
    Return a dictionary mapping each word (separated by white
    space) in the string words to
    the number of times it occurs. Exclude words that appear
    in stop_words.
    """
    d = {}
    for w in words:
        if not w in stop_words:
            c = d.setdefault(w, 0)
            d[w] = c + 1
    return d

phrase = "I didn't ask for a dime"
d = histogram(phrase, ["for"])
print(d["a"])
print(d["dime"])
```

3. In homework 6, you will be using statistical tools to analyze datasets. One common measure for the difference/distance between two datasets is the mean squared error. MSE is computed as follows:
For each point in one dataset:

- compute the difference between it and the corresponding point in the other dataset square this difference
- Take the average of these squared differences.

Compute the MSE between f, g, and h. What can these numbers tell you?

x	f(x)	g(x)	h(x)
1	4	1	8
2	5	3	6
3	6	9	4