CS 61A Summer 2017 Structure and Interpretation of Computer Programs

Quiz 9 Solutions

INSTRUCTIONS

- You have 10 minutes to complete this quiz.
- The exam is closed book, closed notes, closed computer, closed calculator.
- The final score for this quiz will be assigned based on **effort** rather than correctness.
- Mark your answers **on the exam itself**. We will *not* grade answers written on scratch paper.
- For multiple choice questions,
  - ☐ means mark **all options** that apply
  - ◯ means mark a **single choice**

Last name	
First name	
Student ID number	
CalCentral email (`_@berkeley.edu`)	
Teaching Assistant	◯ Alex Stennet ◯ Kelly Chen ◯ Angela Kwon ◯ Michael Gibbes ◯ Ashley Chien ◯ Michelle Hwang ◯ Joyce Luong ◯ Mitas Ray ◯ Karthik Bharathala ◯ Rocky Duan ◯ Kavi Gupta ◯ Samantha Wong
Name of the person to your left	
Name of the person to your right	
All the work on this exam is my own. **(please sign)**	

1. (5 points) Oops! . . . I Did It Again

(a) (1 pt) Britney wants to define a `Person` class.

```
class Person:
    name = None
    def __init__(self, name):
        Person.name = name
    def greet(self):
        return 'Hello, my name is ' + self.name
```

Angela, however, sees a problem. Mark **all** appropriate criticisms of this implementation.

- [x] Every `Person`'s name will be equal to the most recently-created `Person`'s name.
- [ ] Instantiating a `Person` will cause an error.
- [ ] Every `Person`'s name will be `None`.
- [ ] Invoking `greet` on a person instance will cause an error.

(b) (2 pt) Consider the following simple class definition.

```
class Dog:
    def bark(self):
        print('woof!')
```

One day, while using this class, Britney decides she wants her dog, Lacey, to bark differently:

```
>>> lacey = Dog()
>>> lacey.bark = 'bow wow!'
```

Rocky quickly points out that this won't work. "`bark` is supposed to be a method, not a string!" So Britney attempts to reset the `bark` method to what it was before:

```
>>> lacey.bark = Dog.bark
```

Rocky isn't convinced this will fix it. Mark **all** appropriate statements about this assignment statement.

- [ ] Executing this assignment statement will cause an error.
- [x] After this assignment, invoking `fido.bark()` will cause an error.
- [ ] This assignment statement will have no effect at all.
- [ ] None of the above criticisms are valid.

(c) (2 pt) Mark **all** lines that should be removed so that the expression `N().r()` evaluates to `1`.

```
[ ] class M:
[x]     p = 2 # optional
[ ]     q = True
[ ]     def r(self):
[ ]         if self.q:
[ ]             return self.p
[x]         return self.r() - 1 # optional

[ ] class N(M):
[ ]     p = 1
[x]     q = False
[x]     def r(self):
[x]         return self.p + 1
```