Akmal Ataev

NYU Student ID aa44

LAB

HW 40

In [1]:

```
class Song(object):

    def __init__(self,lyrics):
        self.lyrics = lyrics

    def sing_me_a_song(self):
        for line in self.lyrics:
            print(line)

happy_bday = Song(["Happy birthday to you",
                   "I don't want to get sued",
                   "So I'll stop right there"])

bulls_on_parade = Song(["They rally around the family",
                        "With pockets full of shells"])

happy_bday.sing_me_a_song()

bulls_on_parade.sing_me_a_song()
```

```
Happy birthday to you
I don't want to get sued
So I'll stop right there
They rally around the family
With pockets full of shells
```

HW 41

In [2]:

```
import random
from urllib.request import urlopen
import sys

WORD_URL = "http://learncodethehardway.org/words.txt"
WORDS = []

PHRASES = {
    "class %%%(%%%):":
      "Make a class named %%% that is-a %%%.",
    "class %%%(object):\n\tdef __init__(self, ***)" :
      "class %%% has-a __init__ that takes self and *** parameters.",
    "class %%%(object):\n\tdef ***(self, @@@)":
      "class %%% has-a function named *** that takes self and @@@ parameters.",
    "*** = %%%()":
      "Set *** to an instance of class %%%.",
    "***.***(@@@)":
      "From *** get the *** function, and call it with parameters self, @@@.",
    "***.*** = '***'":
```

```
      "From *** get the *** attribute and set it to '***'."
}

# do they want to drill phrases first
PHRASE_FIRST = False
if len(sys.argv) == 2 and sys.argv[1] == "english":
    PHRASE_FIRST = True

# load up the words from the website
for word in urlopen(WORD_URL).readlines():
    WORDS.append(word.strip().decode("utf-8"))

def convert(snippet, phrase):
    class_names = [w.capitalize() for w in random.sample(WORDS, snippet.count("%%%"))]
    other_names = random.sample(WORDS, snippet.count("***"))
    results = []
    param_names = []

    for i in range(0, snippet.count("@@@")):
        param_count = random.randint(1,3)
        param_names.append(', '.join(random.sample(WORDS, param_count)))

    for sentence in snippet, phrase:
        result = sentence[:]

        # fake class names
        for word in class_names:
            result = result.replace("%%%", word, 1)

        # fake other names
        for word in other_names:
            result = result.replace("***", word, 1)

        # fake parameter lists
        for word in param_names:
            result = result.replace("@@@", word, 1)

        results.append(result)

    return results

# keep going until they hit CTRL-D
try:
    while True:
        snippets = list(PHRASES.keys())
        random.shuffle(snippets)

        for snippet in snippets:
            phrase = PHRASES[snippet]
            question, answer = convert(snippet, phrase)
            if PHRASE_FIRST:
                question, answer = answer, question

            print(question)

            input("> ")
            print("ANSWER:  %s\n\n" % answer)
except:
    print("\nBye")
```

```
class Bath(Argument):
> Bath
ANSWER:  Make a class named Bath that is-a Argument.


class Camp(object):
 def __init__(self, bottle)
>
ANSWER:  class Camp has-a __init__ that takes self and bottle parameters.


coil.bulb = 'base'
```

>
ANSWER: From coil get the bulb attribute and set it to 'base'.

```
competition.canvas(bell, cave, comfort)
```

>
ANSWER: From competition get the canvas function, and call it with parameters self, bell
, cave, comfort.

```
distance = Discussion()

Bye
```

In [ ]:

```
```