1. Turn in your code [20pts, HELP].
(If you are going for bonus points, write a sentence on how you accomplished it)
This ELIZA program is an expert on sports cars (5 different ones). ELIZA looks for keywords in the user's entry it infer what they might be asking about. Using keywords allows for a more general interpretation of the user's entry without having to handle different sentence structures or orders. For example, if the user input contains the word 'engine' then ELIZA assumes they want to know what type of engine a car has. The downside to this method is that there are many possible reasons the word engine could be in the input so this method is quite specific.

```
import re
import random
import string
import Rules

class ELIZA:
    """This chatbot will talk about sports cars"""

    def __init__(self):
        self.ruleDict = {}
        self.text = 'ELIZA: Hello, my name is ELIZA. Ask me anything
about sports cars!'

    def getInput(self, userInput):
        """Enter function"""
        return input(userInput)

    def randomRule(self, ruleList):
        """Takes in a list of rules and returns one at random"""
        number = random.randint(0, len(ruleList) - 2 ) #-2 to avoid
rule 15 for multiple

        return ruleList[number]

    def matchRule(self, input):
        """Enter function"""
        ruleList = []

        for key, rule in self.ruleDict.items():
            matches, answer = rule(self, input)
            if matches:
                ruleList += [answer]

        if len(ruleList) > 1:
            return self.randomRule(ruleList)
        elif len(ruleList) == 1:
            return ruleList[0]
```

```
        else:
            match, answer = self.ruleDict[14](self, input)
            return answer

    def generateRules(self):
        """Generates a dictionary of rules"""
        for i in range(1, 16):
            self.ruleDict[i] = eval('Rules.rule' + str(i))

    def main(self):
        self.generateRules()
        print(self.text)
        userInput = self.getInput("USER:
").lower().strip(string.punctuation)

        while userInput != 'q':

            elizaOutput = self.matchRule(userInput.lower())
            print('ELIZA: ' + elizaOutput)
            userInput = self.getInput('USER:
').lower().strip(string.punctuation)

if __name__ == "__main__":
    ELIZA().main()


def carInDict(car):
    return car in carDict

def rule1(self, input):
    """Rule 1 applies to questions about parts of a car"""
    one = re.search(r'\b(?:sort|type|kind |.*)\b \b(?:.*)\b(.+)
\b(?:.*)\b \b(?:a|an|the)\b (.+) \b(?:have|possess)', input)
    if one and carInDict(one[2]) and (carDict.get(one[2],
False).get(one[1], False)):
        car = one[2]
        part = carDict[car][one[1]]
        return True, 'The ' + car.capitalize() + ' has a ' + part + '
' + one[1] + '.'
    else:
        return False, None

def rule2(self, input): #
    """Rule 2 applies to questions about the horsepower of a car"""
    two = re.search(r'how much (.+) does \b(?:a|an|the)\b (.+) have?',
input)
    if two and carInDict(two[2]):
        car = two[2].capitalize()
        return True, 'A ' + car + ' has ' + carDict[two[2]]['power'] +
' ' + two[1] + '.'
```

```
    else:
        return False, None

def rule3(self,input): # 2 var
    """Rule 3 answers questions about the top speed of a car"""

    three = re.search(r'the (.+) (.+) of \b(?:a|an|the)\b (.+)',
input)

    if three and carInDict(three[3]):
        topSpeed = three[1] + ' ' + three[2]
        car = three[3]
        madeBy = carDict[car]['make']
        answer = carDict.get(three[3], None)[topSpeed]

        return True, 'The ' + topSpeed + ' of a ' + madeBy + ' ' +
car.capitalize() + ' is ' + answer
    else:
        return False, None

def rule4(self,input): # 2 var
    """Rule 4 applies to questions about the cost of a car. """
    four = re.search(r'\b(?:a|an|the)\b (.+) (cost+)', input)

    if four and carInDict(four[1]):
        car = four[1]
        cost = carDict.get(car, None)['price']
        make = carDict.get(car, None)['make']
        return True, 'A ' + make + ' ' + car.capitalize() + ' ' +
four[2] + 's ' + cost + '.'
    else:
        return False, None

def rule5(self,input): # 1 var
    """Rule 5 applies to questions about how fast a car is"""
    five = re.search(r'\b(?:fast|quick)\b is \b(?:a|an|the)\b (.+)',
input)
    if five and carInDict(five[1]):
        car = five[1]
        speed = carDict.get(car)['60']
        if carDict.get(car)['fastest']:
            return True, 'The ' + car.capitalize() + ' goes 0-60 mph
in ' + speed + ", it's the fastest car I know of!"
        return True, 'The ' + car.capitalize() + ' goes 0-60 mph in '
+ speed
    else:
        return False, None

def rule6(self,input): #2 var
    """Rule 6 applies to questions about who makes a car"""
```

```
    six = re.search(r'\b(.:|makes|manufactures|builds)\b \b(?:a|an|
the)\b (.+)', input)
    if six and carInDict(six[2]):
        car = six[2]
        make = carDict.get(car, None)['make']
        return True, make + ' ' + six[1] + ' the ' + car.capitalize()
    else:
        return False, None

def rule7(self,input): #2 var
    """Rule 7 applies to questions about convertible cars"""
    seven = re.search(r'\b(?:a|an|the)\b (.+) (?:.*) (?:.*)
(convertible+)', input)
    if seven and carInDict(seven[1]):
        if carDict.get(seven[1], False).get(seven[2]):
            return True, 'You can get a ' + seven[1] + ' in a ' +
seven[2]
        else:
            return True, 'You cannot get a ' + seven[1] + ' in a ' +
seven[2]
    else:
        return False, None

def rule8(self,input): #1 var
    """Rule 8 applies to questions about colors of cars"""
    eight = re.search(r'(?:what) (color|colors)+ (?:.*)', input)
    if eight:
        return True, 'It comes in the ' + eight[1] + ' ' + getColors()
    else:
        return False, None

def rule9(self,input): # 1 var
    """Rule 9 applies to definition of a sports car"""
    nine = re.search(r"(?:what|what is a|what's a) (sport car|sports
car)+", input)
    if nine:
        carString = ", "
        carList = list(carDict.keys())
        carList = [word.capitalize() for word in carList]
        carList.insert(4, 'and')
        carString = (carString.join(carList))

        return True, 'A ' + nine[1] + ' is really fast and handles
well.\nThe ' + carString + ' are all great sports cars.'
    else:
        return False, None

def rule10(self,input):
    """Rule 10 applies if the user asks what ELIZA's favorite car
is"""
```

```
    ten = re.search(r'(favorite+)', input)
    if ten:
        make, car = randomCar()
        return True, 'My ' + ten[1] +' sports car is the ' + make + '
' + car + '.' + ' What is your ' + ten[1] +'?'
    else:
        return False, None

def rule11(self,input):
    """""""
    eleven = re.search(r'i aborr (.+)', input)
    if eleven:
        return True, 'Why do you 11 ' + eleven[1] + '?'
    else:
        return False, None

def rule12(self,input):
    """Enter function"""
    twelve = re.search(r'i aborr (.+)', input)
    if twelve:
        return True, 'Why do you 12 ' + twelve[1] + '?'
    else:
        return False, None

def rule13(self,input):
    """Enter function"""
    thirteen = re.search(r'i aborr (.+)', input)
    if thirteen:
        return True, 'Why do you 13 ' + thirteen[1] + '?'
    else:
        return False, None

def rule14(self, input):
    """This rule returns a random input when ELIZA doesn't understand
the question"""
    partOne, partTwo = randomCar()

    return False, "Let's just talk about sports cars, okay? " + "Want
to hear about the " + partOne + " " + partTwo + "?"

def rule15(self,input):
    """Rule 15 applies if a car other than one ELIZA knows is asked
about"""
    fifteen = re.search(r'\b(?:a|an|the)\b (.+) (?:)', input)
    partOne, partTwo = randomCar()
    if fifteen and not carInDict(fifteen[1]):
        return True, "I don't think a " + fifteen[1].capitalize() + "
is a sports car." + " Want to hear about the " + partOne + " " +
partTwo + "?"
    else:
```

```
        return False, None

def randomCar():
    """Returns a random car from carDict"""
    modelList = list(carDict.keys())
    randomCar = modelList[random.randint(0, len(carDict) - 1)]
    partOne = carDict[randomCar]['make']
    partTwo = carDict[randomCar]['name']

    return partOne, partTwo

def getColors():
    number = random.randint(0, len(colors) - 1)
    return colors[number]

corvette = {'name': 'Corvette', 'transmission': '7spd manual','price':
'$123,000','power': '755','make': 'Chevrolet', 'engine': 'V8', 'top
speed': '200mph', '60': '3.0 seconds', 'fastest': False,
'convertible': False}
huracan = {'name': 'Huracan', 'transmission': '7spd automatic',
'price': '$261,000', 'power': '630', 'make': 'Lamborghini', 'engine':
'V10','top speed': '199mph', '60': '3.4 seconds', 'fastest': False,
'convertible': False}
nineEleven = {'name': '911', 'transmission': '7spd automatic',
'price': '$123,000', 'power': '540', 'make': 'Porsche', 'engine':
'Turbo Boxer 6','top speed': '191mph', '60': '2.8 seconds', 'fastest':
False , 'convertible': True}
roadster = {'name': 'Roadster', 'transmission': '1spd automatic',
'price': '$200,000', 'power': '800', 'make': 'Tesla', 'engine':
'Electric','top speed': '250mph', '60': '1.9 seconds', 'fastest':
True , 'convertible': False}
eightTwelveSuper = {'name': '812 Superfast', 'transmission': '7spd
automatic','price': '$335,000', 'power': '788', 'make': 'Ferrari',
'engine': 'V12','top speed': '211mph', '60': '2.9 seconds', 'fastest':
False, 'convertible': False}

colors = ['black', 'blue', 'red', 'silver', 'green', 'yellow']

carDict = {'corvette': corvette, 'huracan': huracan, '911':
nineEleven, 'roadster': roadster, '812 superfast': eightTwelveSuper}

#end code
```

2. *Intentionally left blank*

3. ELIZA would perform better in conversations if it looked for keywords and syntactical queues to infer more meaning about the user input.

4. "How many fins does a cat have?" Is a sentence that a pet expert

ELIZA would not get correct. It would likely be able to figure out that fins are an animal body part but not that cats don't have fins because ELIZA doesn't understand what a cat is or what fins are.