# [This document: http://tinyurl.com/qhs62ar]

# Instructions

Form a group of 4-5. Start on Question 0. Check off with a lab assistant when everyone in your group understands how to solve Question 0. Repeat for Question 1, 2, etc. **You're not allowed to move on from a question until you check off with a lab assistant.** You are allowed to use any and all resources at your disposal, including the interpreter, lecture notes and slides, discussion notes, and labs. You may consult the lab assistants, **but only after you have asked everyone else in your group.** **The purpose of this section is to have all the students working together to learn the material.**

### Question 0

**0a.** What are Python, Scheme, and Logic examples of? How are they the same? How are they different?

**0b.** What is a fact? What is a query?

**0c.** Are there function calls in Logic? How do we get anything done?

**0d.** Explain unification.

### Question 1

Grab the logic file from <http://inst.eecs.berkeley.edu/~cs61a-tz/guerrilla/hp.logic>

It contains various facts concerning the relationships between characters in the [Harry Potter](http://en.wikipedia.org/wiki/Harry_Potter) universe

**1a.** Write a query to list all the children of Molly (Weasley)

**1b.** Write a query to list all the people who are married and their spouses

**1c.** Finally, write a fact to determine if a person is a sibling of another person

(two people are siblings if they have the same parent)::

;;; Here are some tests

(query (sibling ron fred))

; expect Success!

(query (sibling fred ?who))

; expect Success! ; who: bill ; who: charlie ; who: percy ; who: fred ; who: george ; who: ron ; who: ginny ; who: bill ; who: charlie ; who: percy ; who: fred ; who: george ; who: ron ; who: ginny

**1d.** From above, Why is each child repeated twice? Think about how unification processes this

query.

### Question 2

**2a.** Write append.

(query (append (1 2 3) (4 5 6) (1 2 3 4 5 6)))

; expect Success!

(query (append ?x (7 4 6) (2 8 3 7 4 6)))

; expect Success! ; x: (2 8 3)

**2b.** Write reverse.

(query (reverse (1 2 3) (3 2 1))

; expect Success!

(query (reverse (1 2 3) ?y)

; expect Success! ; y: (3 2 1)

**2c.** Write member.

(query (member 3 (7 4 3)))

; expect Success!

**2d.** Write equal-lists.

(query (equal-lists (1 8 2) (1 8 2))

; expect Success!

### Question 3

A molecule of [deoxyribonucleic acid](http://en.wikipedia.org/wiki/DNA), or DNA, consists of two long strands of four nucleotides: adenine (A), thymine (T), cytosine (C), and guanine (G). The two strands are complementary to each other. This means that a nucleotide on one strand is bonded to its complement on the other: adenine will be bound to thymine, and cytosine will be bound to guanine.

The following are some facts about the complements of these nucleotides:

;;; Facts about DNA

(fact (complementary t a))

(fact (complementary a t))

(fact (complementary c g))

(fact (complementary g c))

Using these facts, write facts for rev\_comp\_strand, which relates a strand of DNA, represented as an arbitrarily long list of four symbols, with its reverse complement, which is another strand of DNA that has the complementary nucleotides, but *in reverse order*. You may use facts from Question 2. See the test below for an example.

;;; Test

(query (rev-comp-strand (t c t g a) ?what))

; expect Success! ; what: (t c a g a)

### Question 4

Write facts for unzip, which relates a larger list to two smaller

lists, the first of which has all the elements at even positions in the larger

list (starting at 0), and the second of which has all the elements at odd

positions. You may assume that the larger list has an even number of elements.

;;; Tests

(query (unzip (a b c d e f) ?odds ?evens))

; expect Success! ; odds: (a c e) evens: (b d f)

(query (unzip ?what (a b c) (d e f)))

; expect Success! ; what: (a d b e c f)