INSTRUCTIONS

- You have 10 minutes to complete this quiz.
- The exam is closed book, closed notes, closed computer, closed calculator.
- Mark your answers on the exam itself. We will not grade answers written on scratch paper.
- For multiple choice questions, fill in each option or choice completely.
  - □ means mark all options that apply
  - ○ means mark a single choice

<table>
<thead>
<tr>
<th>Last name</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>First name</td>
<td></td>
</tr>
<tr>
<td>Student ID number</td>
<td></td>
</tr>
<tr>
<td>CalCentral email (@berkeley.edu)</td>
<td></td>
</tr>
<tr>
<td>Discussion Section</td>
<td></td>
</tr>
<tr>
<td>All the work on this exam is my own. (please sign)</td>
<td></td>
</tr>
</tbody>
</table>

0. Your thoughts? Draw your favorite Halloween costume idea!
1. Bubba Gump

Write the output displayed by the interactive Python interpreter when each expression below is evaluated.

class Tree:
    def __init__(self, label, branches=[]):
        self.label = label
        self.branches = list(branches)

    def is_leaf(self):
        return not self.branches

    def __repr__(self):
        if self.is_leaf():
            return '{' + repr(self.label) + '}
        return '{' + repr(self.label) + ', ' + repr(self.branches) + '}'

forrest = Tree(1)
gump = Tree(1, [forrest, forrest])
forrest.label = 2
forrest = Tree(forrest)

>>> run = Tree(forrest, gump.branches)
>>> run
Tree(Tree(Tree(2)), [Tree(2), Tree(2)])

>>> forrest.label = 1
>>> run
Tree(Tree(1), [Tree(2), Tree(2)])

2. Seeing the Forest for the Trees

Implement all_paths which takes a Tree and returns a list of linked list paths from the root to each leaf.

def all_paths(t):
    class Link:
        empty = ()
        def __init__(self, first, rest=empty):
            self.first = first
            self.rest = rest

    ***
    >>> t = Tree(1, [Tree(2), Tree(3)])
    >>> all_paths(t)
    [Link(1, Link(2)), Link(1, Link(3))]
    ***

    if t.is_leaf():
        return [Link(t.label)]

    paths = []

    for b in t.branches:
        paths += [Link(t.label, path) for path in all_paths(b)]

    return paths