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

<table>
<thead>
<tr>
<th>Last name</th>
<th>First name</th>
<th>Student ID number</th>
<th>CalCentral email (@berkeley.edu)</th>
<th>Teaching Assistant</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Alex Stennet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Angela Kwon</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ashley Chien</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Joyce Luong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Karthik Bharathala</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Kavi Gupta</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Kelly Chen</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Michael Gibbes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Michelle Hwang</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mitas Ray</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rocky Duan</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Samantha Wong</td>
</tr>
</tbody>
</table>

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) Run, Forrest, Run!

(a) (2 pt) For each of the expressions below, write the output displayed by the interactive Python interpreter when the expression is evaluated. The output may have multiple lines. If an error occurs, write “Error”, but include all output displayed before the error. If a function value is displayed, write “Function”.

```python
class Tree:
    def __init__(self, root, branches=[]):
        self.root = root
        self.branches = list(branches)

    def __repr__(self):
        if self.branches:
            branches_str = ', ' + repr(self.branches)
        else:
            branches_str = ''
        return 'Tree(' + repr(self.root) + branches_str + ')

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

>>> run = Tree(forrest, gump.branches)

>>> run

>>> forrest.root = 1

>>> run
```

(b) (3 pt) Implement `all_paths` which takes in a `Tree` and returns a Python list containing all the paths (represented as linked lists) from the root to each leaf. The `Tree` class definition is given above.

```python
class Link:
    def __init__(self, first, rest=empty):
        self.first = first
        self.rest = rest

def all_paths(t):
    if ____________________________________________:
        return _________________________________

    result = []
    _____________________________________________
    _____________________________________________

    return result
```