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></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 (<a href="mailto:_@berkeley.edu">_@berkeley.edu</a>)</td>
<td></td>
</tr>
</tbody>
</table>

Teaching Assistant

| ○ Alex Stennet | ○ Kelly Chen |
| ○ Angela Kwon | ○ Michael Gibbes |
| ○ Ashley Chien | ○ Michelle Hwang |
| ○ Joyce Luong | ○ Mitas Ray |
| ○ Karthik Bharathala | ○ Rocky Duan |
| ○ Kavi Gupta | ○ Samantha Wong |

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) Tree Time

For each line in the implementation of the `IterableTree` class below, fill in the square to the left of the line if removing will help pass the doctests and **the implementation contains as few lines of code as possible.** Don’t cross out any docstrings or doctests.

The `__iter__` generator for this class should yield the values of the tree starting with the root, and yield all of the values of the left branch before any values of the right branch. The `Tree` class definition is shown to the right.

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

        def __iter__(self):
            yield self.root
            for branch in self.branches:
                if branch:
                    if self.branch:
                        branch = iter(branch)
                        for root in branch:
                            yield root
                            yield self.root
                yield root
                yield self.root
```

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

    def is_leaf(self):
        return not self.branches
```