CS 61A Structure and Interpretation of Computer Programs
Summer 2017

Redemption Quiz 1

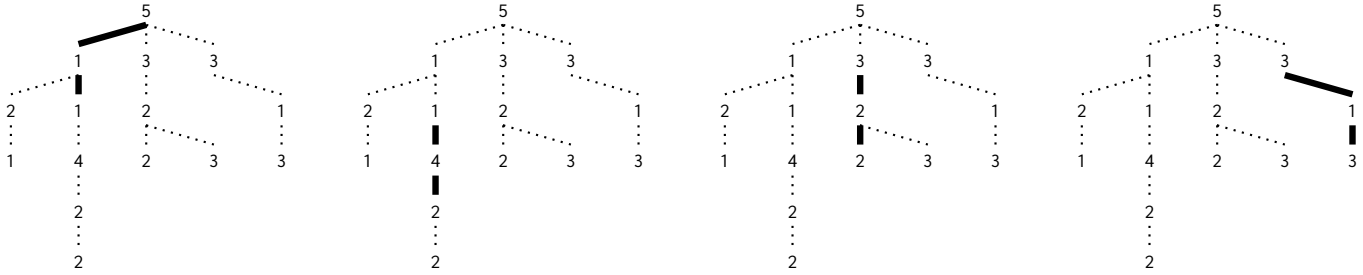
INSTRUCTIONS

- You have 10 minutes to complete this quiz.
- The exam is closed book, closed notes, closed computer, closed calculator.
- This redemption quiz is not worth any points; turn it in at lecture if you would like feedback.
- 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**

Last name	
First name	
Student ID number	
CalCentral email (`_@berkeley.edu`)	
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)**	

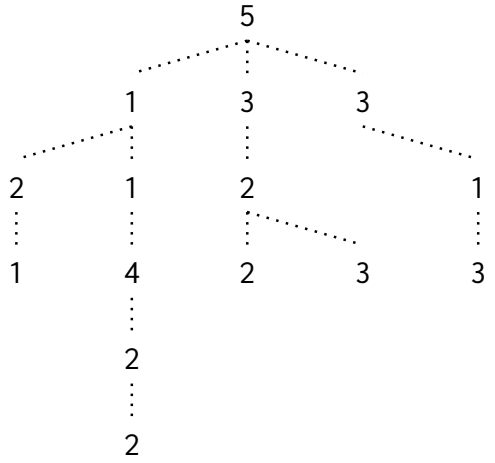
1. (5 points) Temmie Flakes (It's just torn up pieces of construction paper.)

Implement `enumerate_ways`, which takes a tree `t` and an integer `total` and returns a list of the ways any sequence of consecutive nodes can sum to `total`. Below are the four ways included in `enumerate_ways(t1, 7)`.



```
def enumerate_ways(t, total):
    """Return a list of the ways that any sequence of consecutive nodes can sum to total.

    >>> t1 = tree(5, [tree(1, [tree(2, [tree(1)]),
    ...                        tree(1, [tree(4, [tree(2, [tree(2)])])])]),
    ...               tree(3, [tree(2, [tree(2),
    ...                                 tree(3)])]),
    ...               tree(3, [tree(1, [tree(3)])])])
    >>> enumerate_ways(t1, 7)
    [[5, 1, 1], [1, 4, 2], [3, 2, 2], [3, 1, 3]]
    >>> enumerate_ways(t1, 4)
    [[1, 2, 1], [4], [2, 2], [2, 2], [3, 1], [1, 3]]
    >>> t2 = tree(2, [tree(-10, [tree(12)]),
    ...               tree(1, [tree(1),
    ...                        tree(-1, [tree(2)])])])
    >>> enumerate_ways(t2, 2)
    [[2], [-10, 12], [2, 1, -1], [1, 1], [1, -1, 2], [2]]
    >>> enumerate_ways(t2, 4)
    [[2, -10, 12], [2, 1, 1], [2, 1, -1, 2]]
    """
```



```
    def paths(_____________________________________):

        ways = _____________________________________

        if _____________________________________:

            _____________________________________

        for b in branches(t):

            ways += _____________________________________________________________________

            _____________________________________________________________________________

            _____________________________________________________________________________

        return ways

    return _____________________________________________________________________
```