

🌴 IG Problem 1A: Tree Climbing 🌴

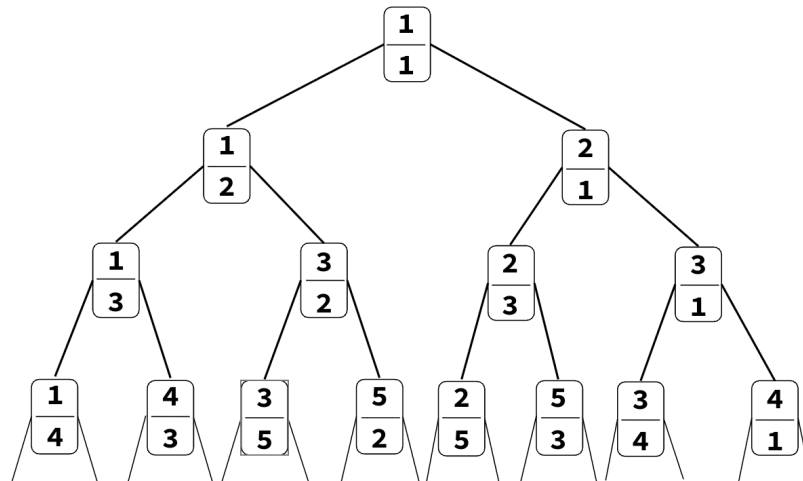
Written by Ben Yan

```

/* Returns a string of Ls and Rs corresponding to left and right moves
 * from the root 1 / 1 to the fraction num / denom on the Calkin-Wilf tree
 */
function cws(num, denom) {
    let seq = "";

    // ascends the tree until we reach the root fraction of 1 / 1
    while (num !== denom) {
        if (num < denom) { // traverse up left edge from child to parent
            seq = "L" + seq;
            denom -= num;
        } else { // traverse up right edge from child to parent
            seq = "R" + seq;
            num -= denom;
        }
    }
    return seq;
}

```



Note: From a vertex fraction **num** / **denom**, we can reach its parent in the Calkin-Wilf tree as follows:

- If **num** < **denom**, it's a left child, and its parent vertex is **num** / (**denom** - **num**) .
- If **num** > **denom**, it's a right child, and its parent vertex is (**num** - **denom**) / **denom** .

Also, we add each new direction ("L" or "R") to the front of the **seq** string, since we're traversing the tree bottom-up (from vertex to root), meaning the order of directions followed is in reverse order!

IG Problem 1B: Pastiche to Matisse

Written by Ben Yan

```
/* This program adds rectangular strips / boxes of varying size, color,
 * and orientation to a canvas, as a homage to Matisse's Snail painting.
 */
"use strict";
```

```
const GWINDOW_HEIGHT = 500; /* in pixels */
const GWINDOW_WIDTH = 500;
```

```
const NUM_BOXES = 10;
const BOX_MIN_WIDTH = 100; /* in pixels */
const BOX_MAX_WIDTH = 150;
const BOX_MIN_HEIGHT = 100;
const BOX_MAX_HEIGHT = 150;
```

```
function Matisse() {
    let gw = GWindow(GWINDOW_WIDTH, WINDOW_HEIGHT);

    for (let i = 0; i < NUM_BOXES; i++) {
        createColorBox();
    }
}
```

```
function createColorBox() {
    // create GRect object with random height and width
    let height = randomInteger(BOX_MIN_HEIGHT, BOX_MAX_HEIGHT);
    let width = randomInteger(BOX_MIN_WIDTH, BOX_MAX_WIDTH);
    let box = GRect(height, width);

    // based on its height and width, randomly position the
    // top-left corner so the GRect stays entirely on-screen
    let x = randomReal(0, GWINDOW_WIDTH - height);
    let y = randomReal(0, GWINDOW_HEIGHT - width);
    gw.add(box, x, y);

    // pigment the interior and boundary with solid color
    box.setColor(randomColor());
    box.setFilled(true);

    // randomly rotate the rectangle box about its center
    let angle = randomInteger(0, 359);
    box.rotate(angle);
}
```

