samePlane

This demonstrates the validator that tests for points to represent the same plane.

JavaScript

```
// NOTE: The below are intended to be used in a validator environment, rather than the valid

// testPlane function tests list a against list b element-wise to
// make sure that the ratio is the same value for each,
// ie a = k.*b; eg a = [1, 1, 2]
// b could be [3, 3, 6] but not [3, 3, 5].
// This can be tricky if one of the inputs is zero though.
function testPlane(a,b) {
        var ratioVec = [];
        var result = 1;
        for (i = 0; i < a.length; i++) {
                if (b[i] == 0) {
                        if (a[i].evaluate() != 0){
                                result = 0;
                        }
                }
                else {
                        var tempRat = a[i].evaluate()/b[i];
                        ratioVec.push(tempRat);
                        console.log(tempRat);
                }
        }
        ratioVec.sort();
        var endPoint = ratioVec.length-1;
        if (ratioVec[0] == ratioVec[endPoint]) {
                result = result*1;
        }
        else {
                result = result*0;
        }

    // Throw in a default check to make sure student doesn't just use all 0s
    // Going a bit overboard in case we want longer vectors in future.

    var isAllZero = 0;
    for (i = 0; i < a.length; i++) {
```

Learning outcomes:

```
37        if(a[i].evaluate() !== 0) {
38          isAllZero = 1;
39          break;
40        }
41      }
42
43      result = result*isAllZero;
44          return result;
45  };
46
47
48  // End of Validators intended to (only) be used in validator environments.
49
```

Problem 1 *This problem shows the 'samePlane' validator. It's intended to detect any equivalent expression that generates the same plane. Note that formatting is tricky for authoring the problem as it requires that you use the validator environment to test multiple answer boxes simultaneously, and you must enter the predicted answers and the ids of the supplied answers in as lists using brackets, and in the same order. Thus if your id for the x coefficient has 'id=x', the y coefficient has 'id=y' and the z coefficient has 'id=z' (in their respective* `\answer` *commands optional argument) and the expression you want for the plane is, say, $-3x, 5y+z = 0$, then you would use the following:*

```
Plug in any expression that generates the plane defined by $x + y + 2z = 0$.
\begin{validator}[{testPlane([x,y,z],[1,1,2])}]
    \[
        \answer[id=x]{1}x + \answer[id=y]{1}y + \answer[id=y]{2}z = 0
    \]
\end{validator}
```

The code is demonstrated below, try various equal and non-equal coefficients and see how it goes. Note that this also works correctly if you feel like making some of the coefficients zero for whatever reason.

Plug in any expression that generates the plane defined by $x + y + 2z = 0$.
`testPlane([AnsOne,AnsTwo,AnsThree],[1,1,2])`

$$\boxed{1}x + \boxed{1}y + \boxed{2}z = 0$$

As a footnote, due to peculiar coding, the value included the the answer command's mandatory argument is never used. So you could have used `\answer[id=x]{673}` *instead of* `\answer[id=x]{1}` *and as long as you entered the correct values into*

the validator environment, it would still only take correct answers for the x coefficient.