1 - quoi tester ?

- reprenons nos fonctions mathématiques

```
function add(a, b) { return a + b; }
function div(a, b) { return a / b; }
function mul(a, b) { return a * b; }
function sub(a, b) { return a - b; }

function testEqual(expected, result) {
    console.log(expected === result);
}

testEqual(3, add(1, 2));
testEqual(3, div(9, 3));
testEqual(3, mul(1.5, 2));
testEqual(5, div(10, 2));
```

- fonctions inutiles: autant utiliser + / * - natif, présents dans tous les langages

Quoi tester ?

- les fonctions de premier ordre qui apportent une vraie valeur
- les fonctions de second ordre ou composite
- les changements d'état d'objet

Exemples

Besoin d'une fonction exprimant le taux de compression d'un gaz en ectopascals selon son indice adiabatique.

```
function compressionRate(initial, final, gamma) {
    const rate = Math.pow(initial / final, 1 / gamma);
    return Number(rate.toFixed(4));
}

function testEqual(expected, result) {
    console.log(`${expected === result} (${result})`);
}

testEqual(0.4472, compressionRate(1000, 5000, 2));
testEqual(0.6687, compressionRate(1000, 5000, 4));
testEqual(0.9009, compressionRate(1000, 3500, 12));
```

- si ça devient compliqué à lire, fixer les valeurs et tester la fonction
- donne plusieurs exemple de résultats à titre de documentation

Un exemple de composite

```
// premier ordre
function first(a, b, c) {
    return (a * b) / c;
}

// second ordre
function div(a, b) {
    return a / b;
}

function mul(a, b) {
    return a * b;
}

function second(a, b, c) {
    return div(mul(a, b), c);
}

// fonction de test
function testEqual(expected, result) {
    console.log(expected == result);
}

// appel des tests
testEqual(3, first(2, 3, 2));
testEqual(3, second(2, 3, 2));
```