# 1 - Corrigé fonctions add, mul, sub, div

```
function add(a, b) {
    return a + b;
}

function addTest() {
    const expectedResult = 3;
    const testValue = add(1, 2);
    console.log(testValue === expectedResult);
}

addTest();
```

```
function sub(a, b) {
    return a - b;
}

function subTest() {
    const expectedResult = 1;
    const testValue = sub(3, 2);

    console.log(testValue === expectedResult);
}

subTest();
```

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

function divTest() {
    const expectedResult = 3;
    const testValue = div(9, 3);

    console.log(testValue === expectedResult);
}

divTest();
```

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

function mulTest() {
    const expectedResult = 10;
    const testValue = mul(5, 2);

    console.log(testValue === expectedResult);
}

mulTest();
```

## Simplifier notre code

- ces tests ne font qu'une comparaison entre deux valeurs
- l'opérateur === suffit en Javascript

```
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(expectedResult, testFunc) {
    console.log(expectedResult === testFunc);
}

testEqual(3, add(1, 2));
testEqual(4, div(8, 2));
testEqual(10, mul(5, 2));
testEqual(3, sub(6, 3));
```

Dans cet exemple:

- les seconds arguments d'appel de testEqual sont retournés à leur valeur calculée

- testEqual ne fait qu'une simple comparaison, sans assignation