07

Passagem por referência

Stefany vai fazer aniversário e, para isso, ela decidiu criar um programa em **JavaScript** que mexe com idades.

Primeiro, ela fez uma função `calculaProximaIdade()`, que recebe a idade que ela tem agora e imprime quantos anos ela terá depois do aniversário. Em seguida, ela criou a função `calculaProximasIdades()`, que recebe a lista de idades dela e de seus amigos e devolve quantos anos todos terão ao final do ano. Por fim, ela fez uma função `calculaIdadesDaqui5Anos`, que recebe a mesma lista de antes mas devolve as idades que todos terão daqui cinco anos.

O programa ficou da seguinte forma:

```
function calculaProximaIdade(idade) {
    idade += 1;
    console.log(idade);
}

function calculaProximasIdades(idades) {
    for (let i = 0; i < idades.length; i += 1) {
        idades[i] += 1;
    }
    console.log(idades);
}

function calculaIdadesDaqui5Anos(idades) {
     for (let i = 0; i < idades.length; i += 1) {
        idades[i] += 5;
```

```
    }
    console.log(idades);
}

const idadeStefany = 21;
calculaProximaIdade(idadeStefany);

const idadesAmigos = [idadeStefany, 20, 23, 18, 7];
calculaProximasIdades(idadesAmigos);

calculaIdadesDaqui5Anos(idadesAmigos);
```

COPIAR CÓDIGO

O programa de Stefany se comporta como o esperado?

Selecione uma alternativa

A Sim! Nenhuma das funções modifica os valores das variáveis `idadeStefany` e `idadesAmigos`. Assim, a última função vai mostrar `[ 26, 25, 28, 23, 12 ]`.

B Não. A segunda função modifica o valor da variável `idadesAmigos`. Assim, a última função vai mostrar `[ 27, 26, 29, 24, 13 ]`, todos com exatamente um ano a mais que o esperado.

C Não. Ambas as funções modificam os valores das variáveis `idadeStefany` e `idadesAmigos`. Assim, a última função vai mostrar `[ 28, 26, 29, 24, 13 ]`, todos com alguns anos a mais que o esperado.

D Não. A primeira função modifica o valor da variável `idadeStefany`. Assim, a última função vai mostrar `[ 27, 25, 28,`

`23, 12 ]`, ou seja, a primeira idade será um ano a mais que o esperado.