**Fonctions**

Table des matières

[Fonctions JS 3](#_Toc69222731)

[Déclaration 3](#_Toc69222732)

[Scope 6](#_Toc69222733)

[Notion de bloc de code 6](#_Toc69222734)

[Retourner une valeur 9](#_Toc69222735)

[Exercice : Calculer une moyenne 12](#_Toc69222736)

[Générer un nombre aléatoire 14](#_Toc69222737)

[Mélanger un tableau 15](#_Toc69222738)

[Méthode map 16](#_Toc69222739)

# Fonctions JS

## Déclaration

Une fonction c’est un bout de code réutilisable

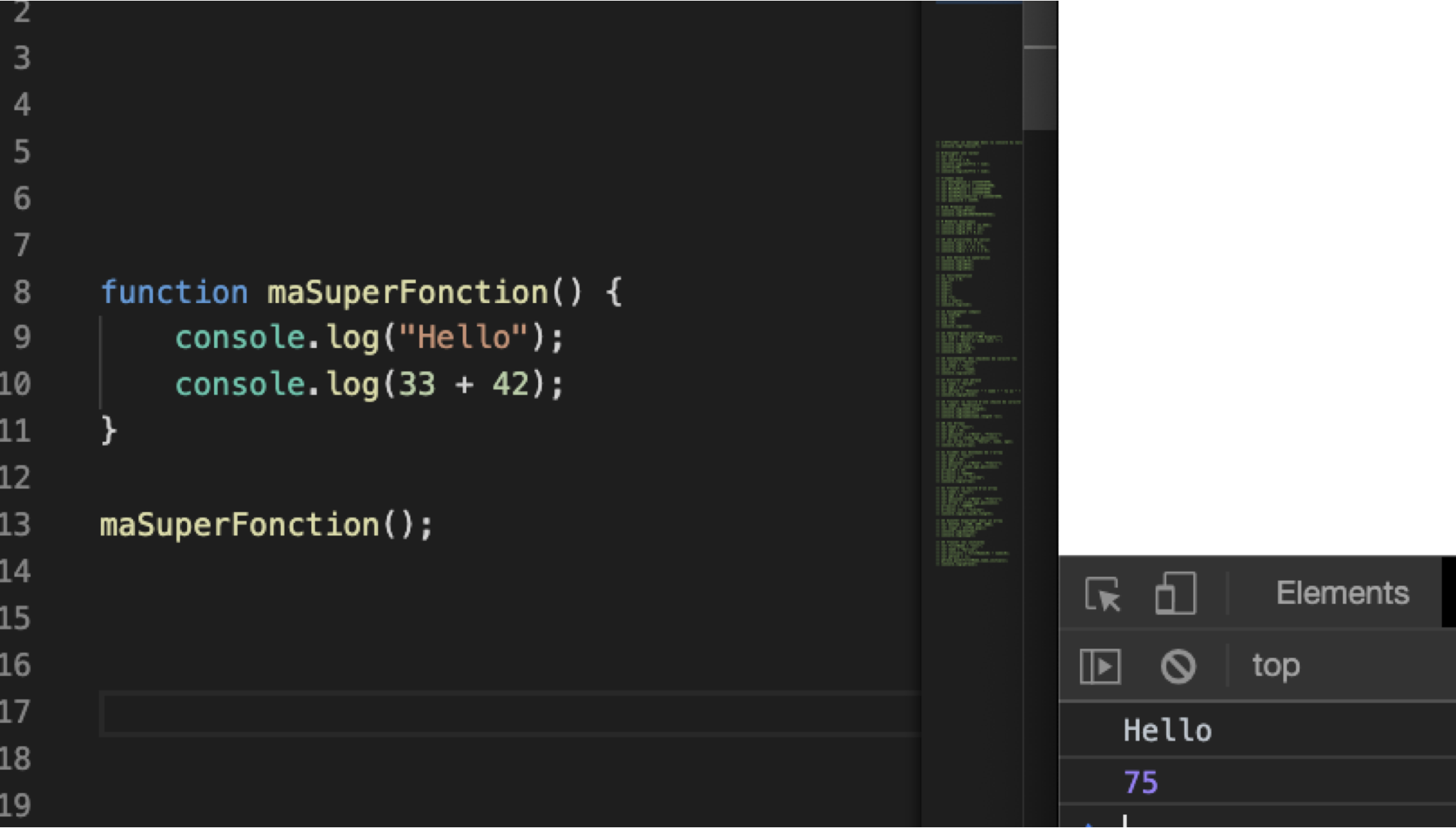
C’est utile car en programmation on a souvent besoin de répéter certaines actions.

(Exemple : Enregistrer un nouvel utilisateur)

On va utiliser function

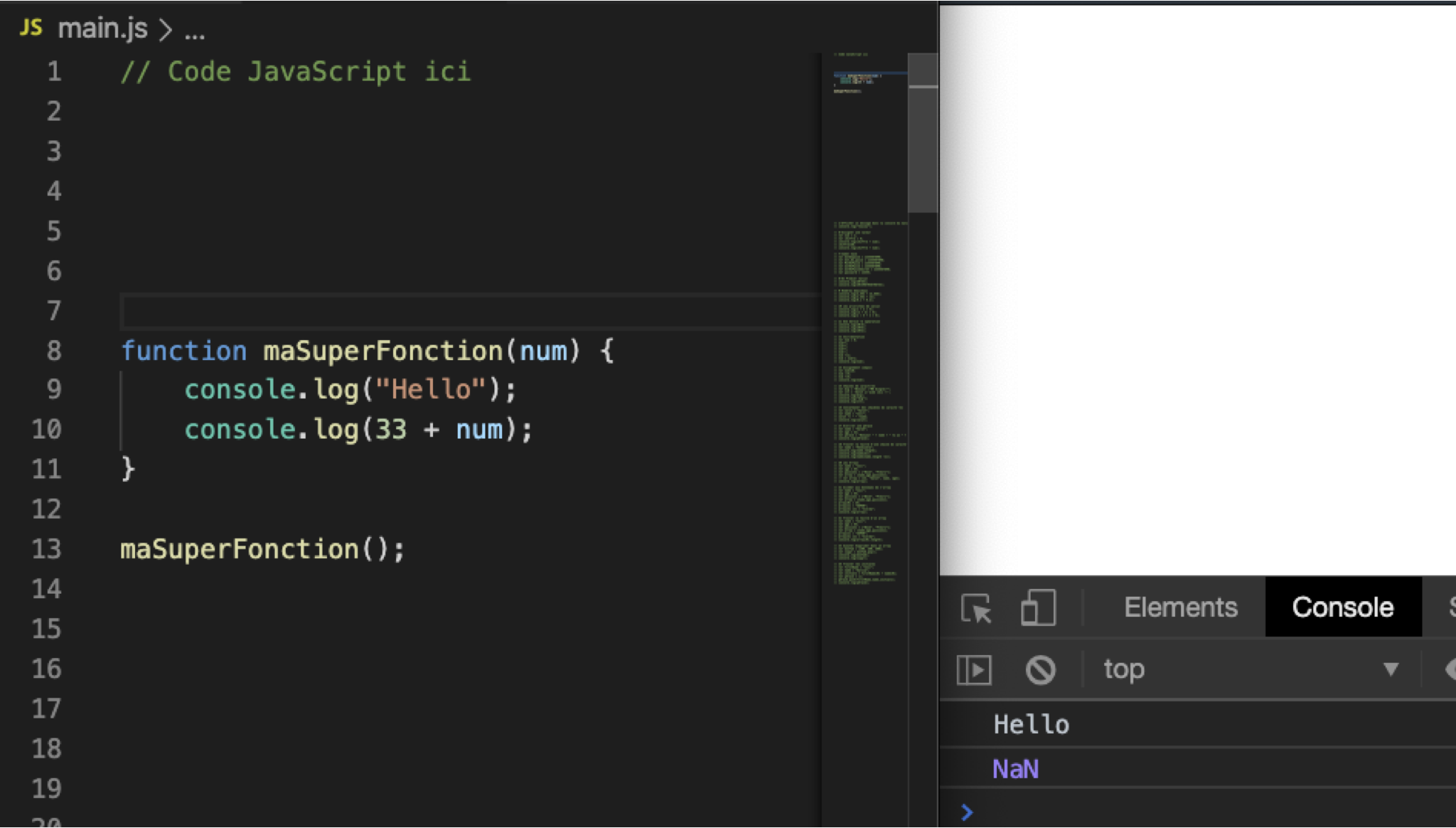
Une fonction va être interprétée comme un sous-programme dans notre fichier.

Quand on a défini une fonction, il faut ensuite l’appeler quelque part dans le programme.



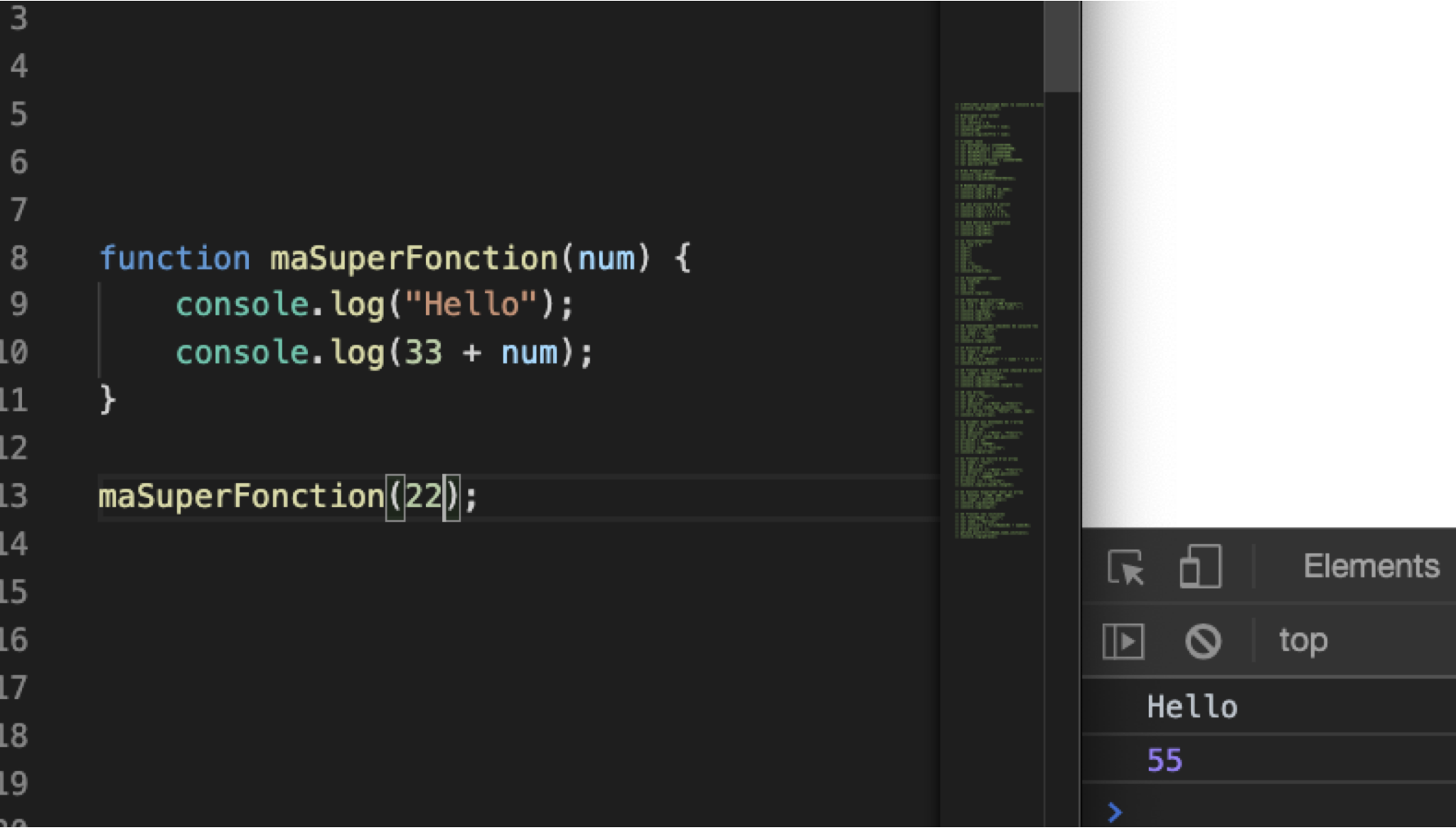
L 13 Il faut appeler au moins 1 fois la fonction

Selon la tâche que doit accomplir la fonction il peut être nécessaire dans certains cas de lui passer une variable en paramètre (Pour faire des calculs par exemple)



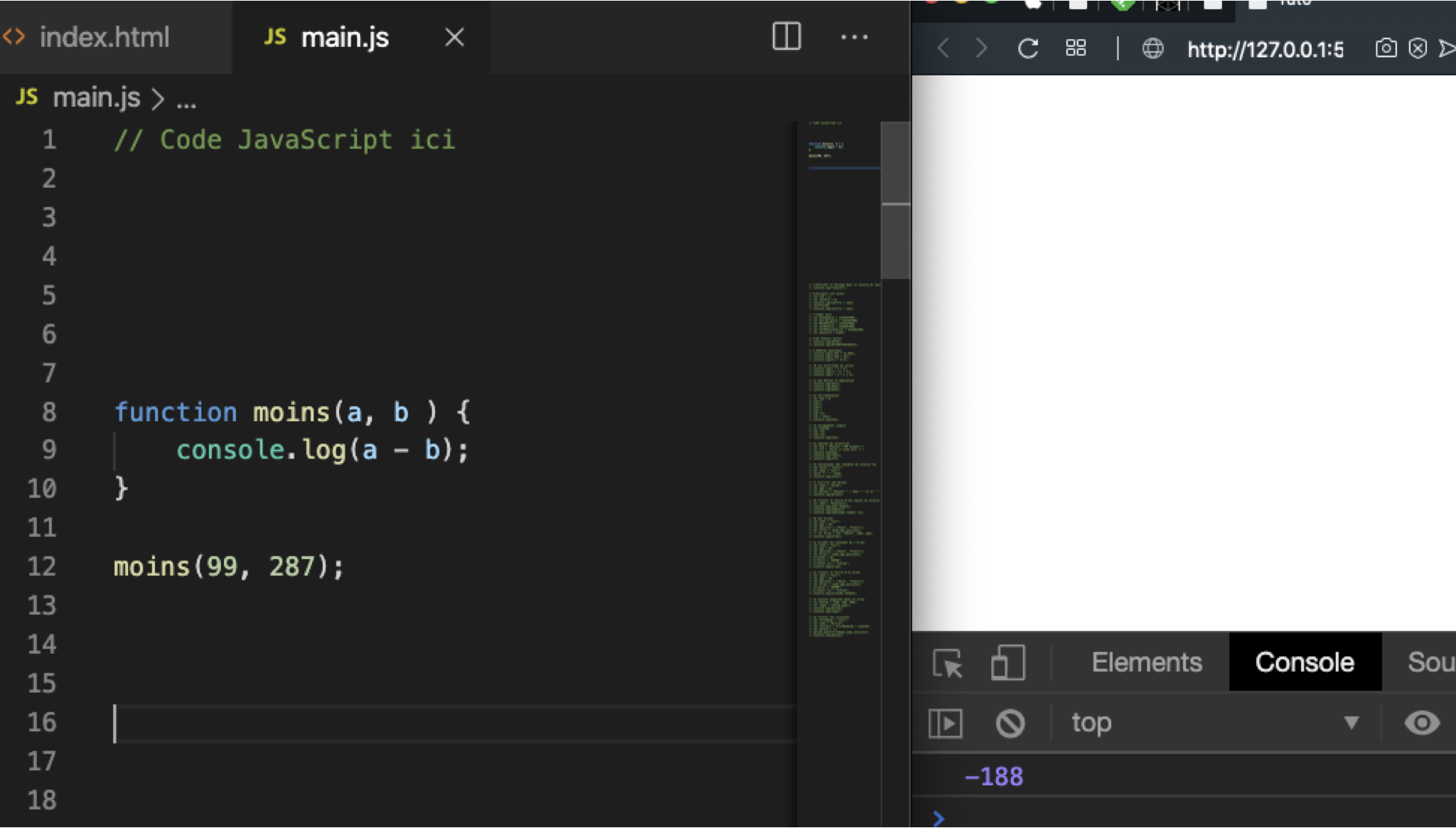
Ici on a une erreur la console renvoi NAN.

Quand on définit la fonction elle reçoit une variable num en paramètre or quand on l’utilise on ne lui passe aucun paramètre



Ici la fonction attend qu’on lui passe quelque chose

On lui passe 22



Dans cette fonction pour faire son calcul on doit lui passer 2 variables en paramètre

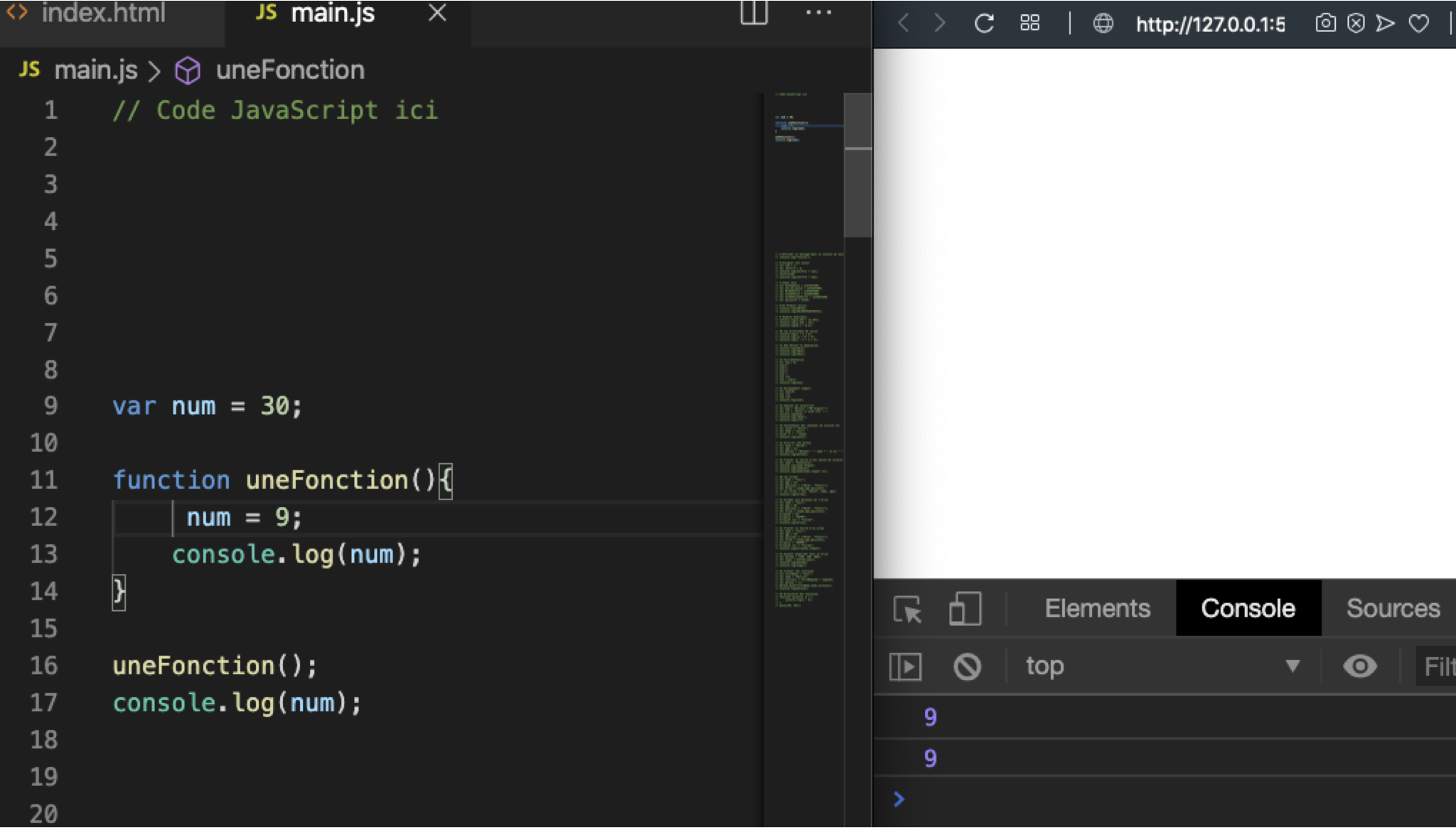
## Scope

En JS, le scope va définir la portée d’une variable, est-ce que ma variable est disponible pour tout le programme ou est-elle disponible uniquement pour ma fonction ?

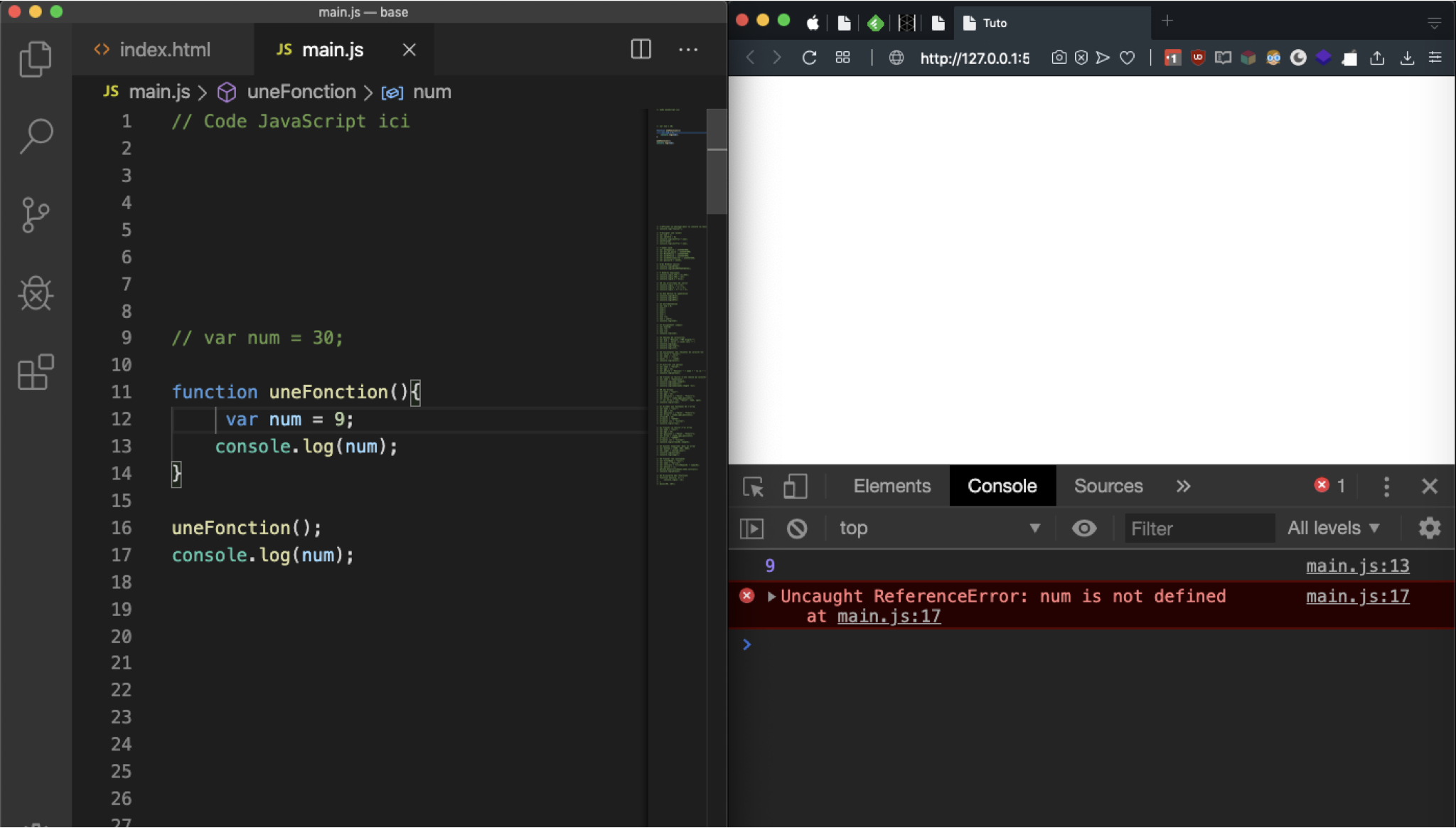
### Notion de bloc de code

On l’a vu les fonctions agissent comme un sous-programme à l’intérieur de notre programme.

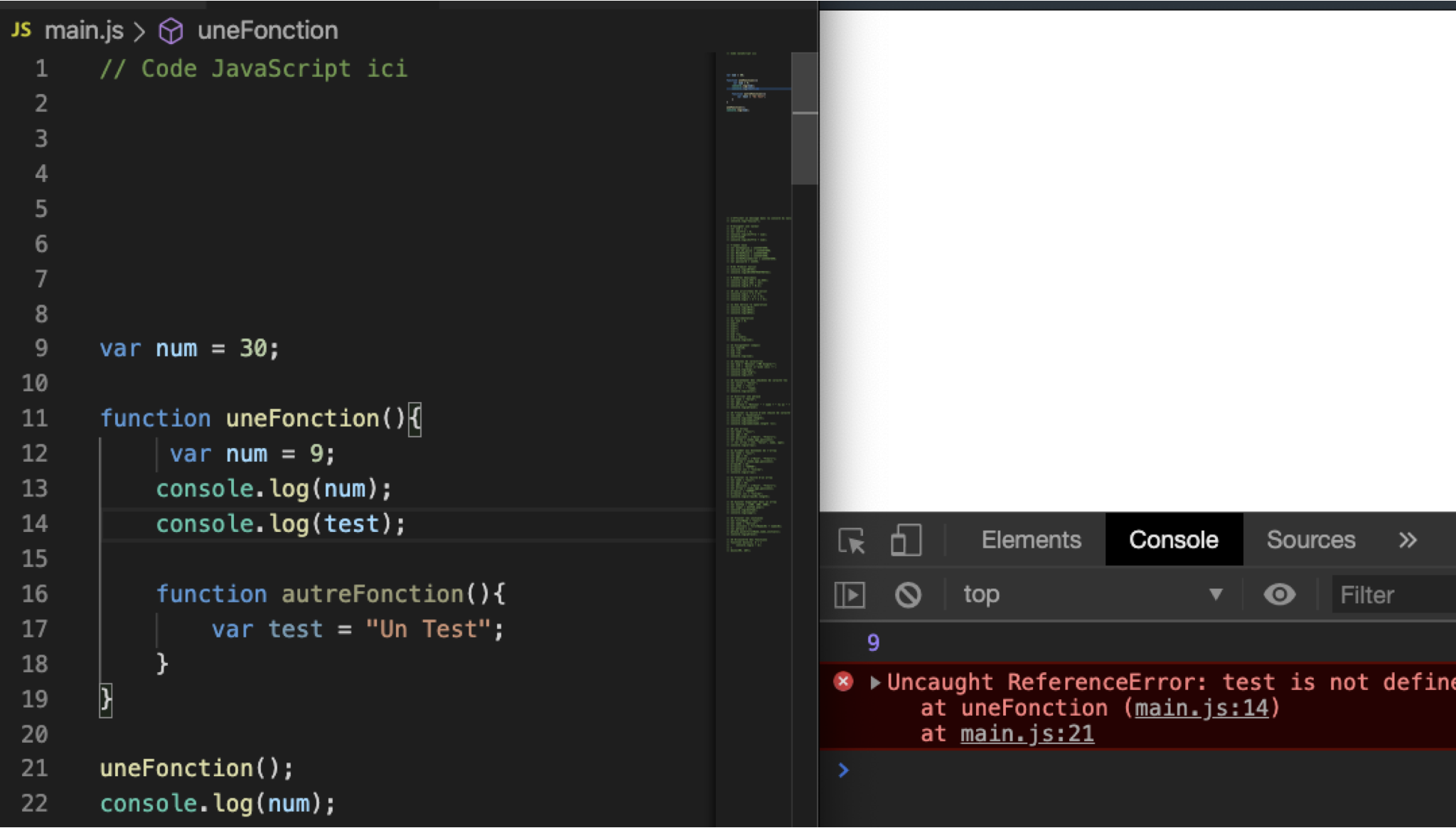
La notion de scope permet de se rendre compte JUSQU’OU une variable peut être accessible dans le code.



Ici je déclare une variable num =30, ensuite dans ma fonction je mets à jour num

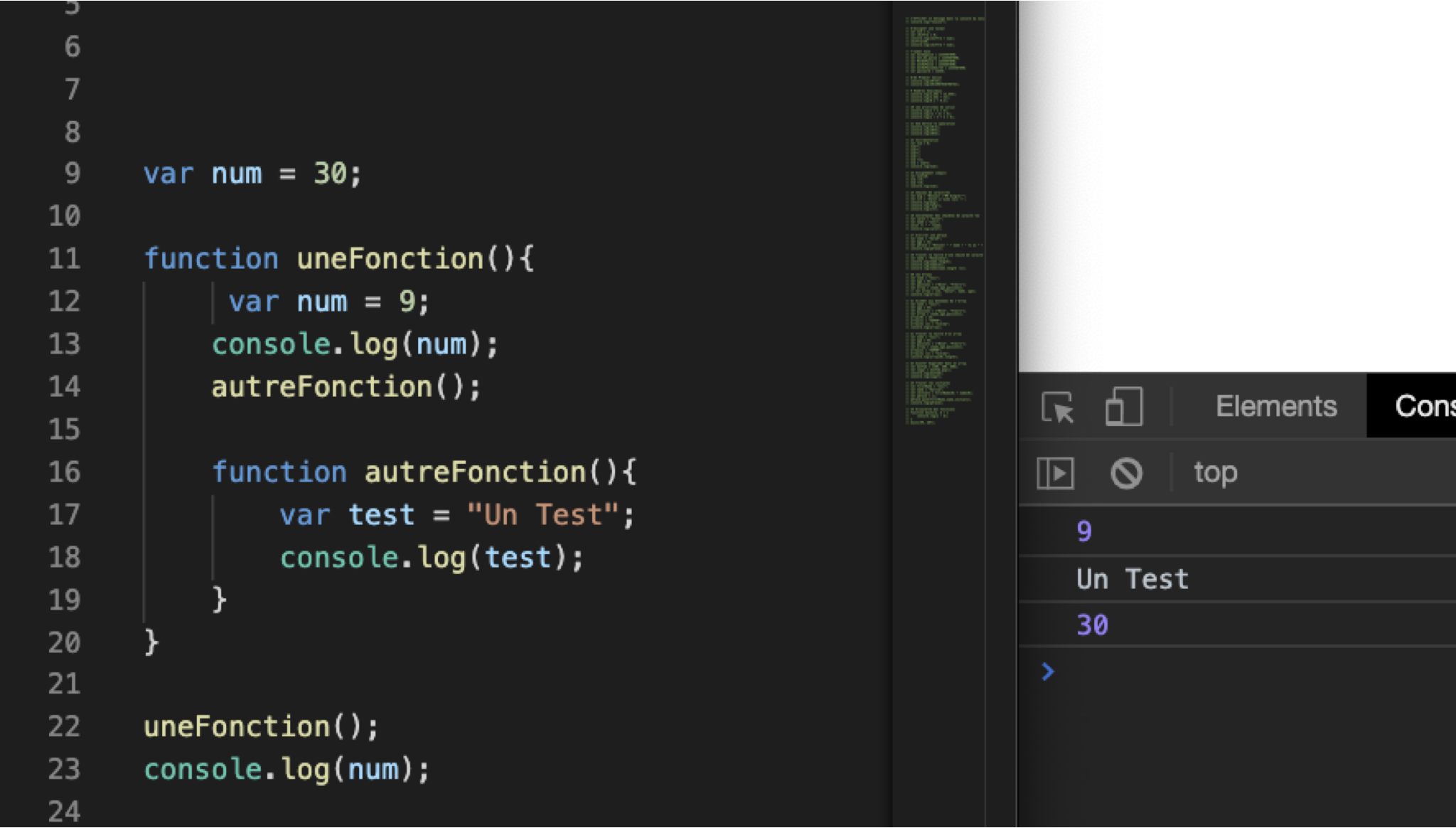


Ici on a une erreur, je déclare une variable num dans ma fonction, et je l’utilise dans mon programme principal, de fait pour le programme principal num n’est pas définit



Ici j’ai une fonction dans une fonction

Erreur car la variable test n’est accessible qu’au sein de la fonction autreFonction()



Quand on a des imbrications : enfant - parent

L’enfant peut accéder aux variables du parent

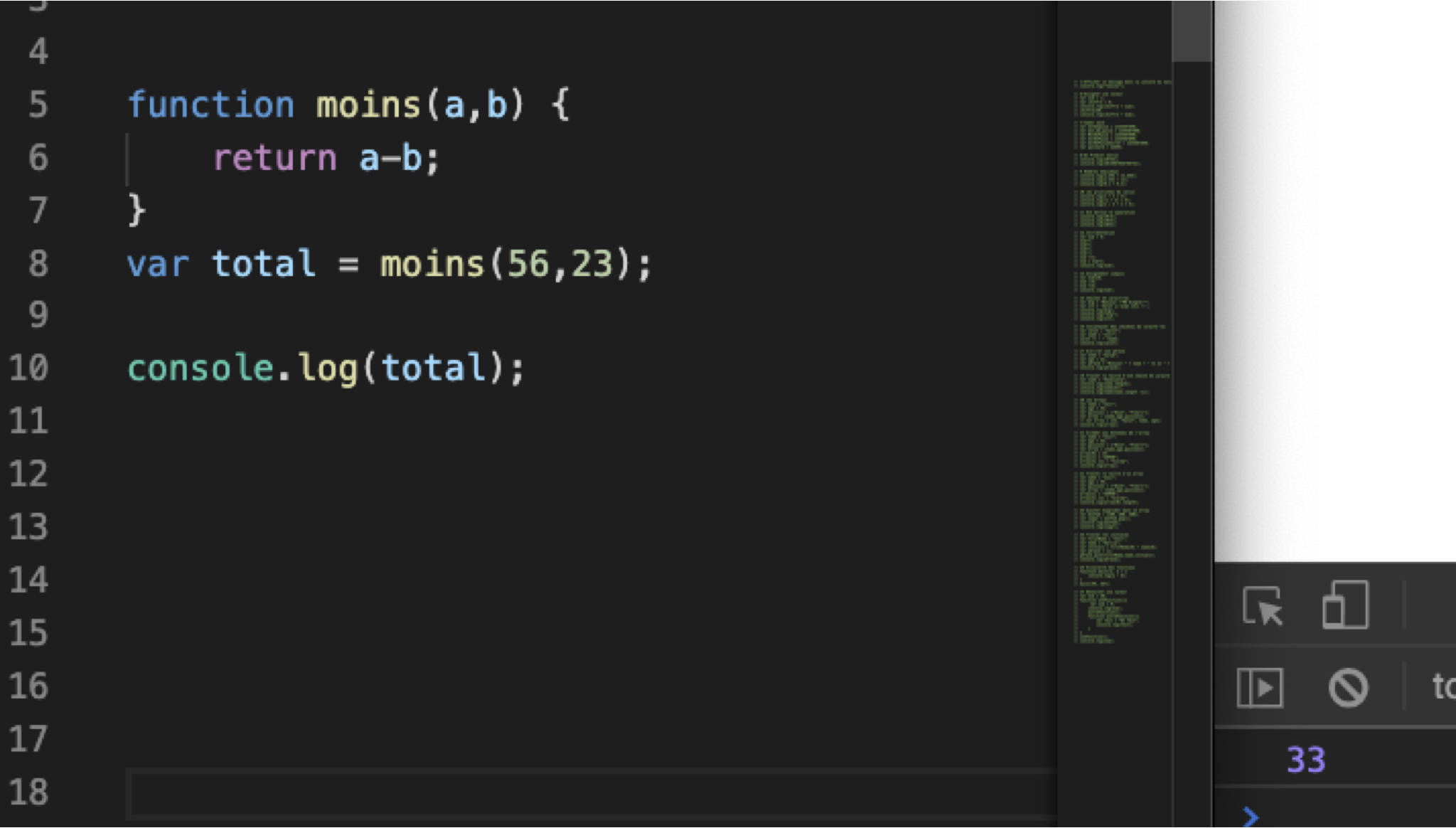
Mais le parent ne peut pas accéder à une variable contenue dans un objet enfant.

## Retourner une valeur

En JS et dans la plupart des langages de programmation, il est possible d’effectuer un return,

on indique à une fonction qu’elle doit nous retourner ou renvoyer quelque chose.

C’est utile si on veut stocker le résultat d’une fonction directement dans une variable.



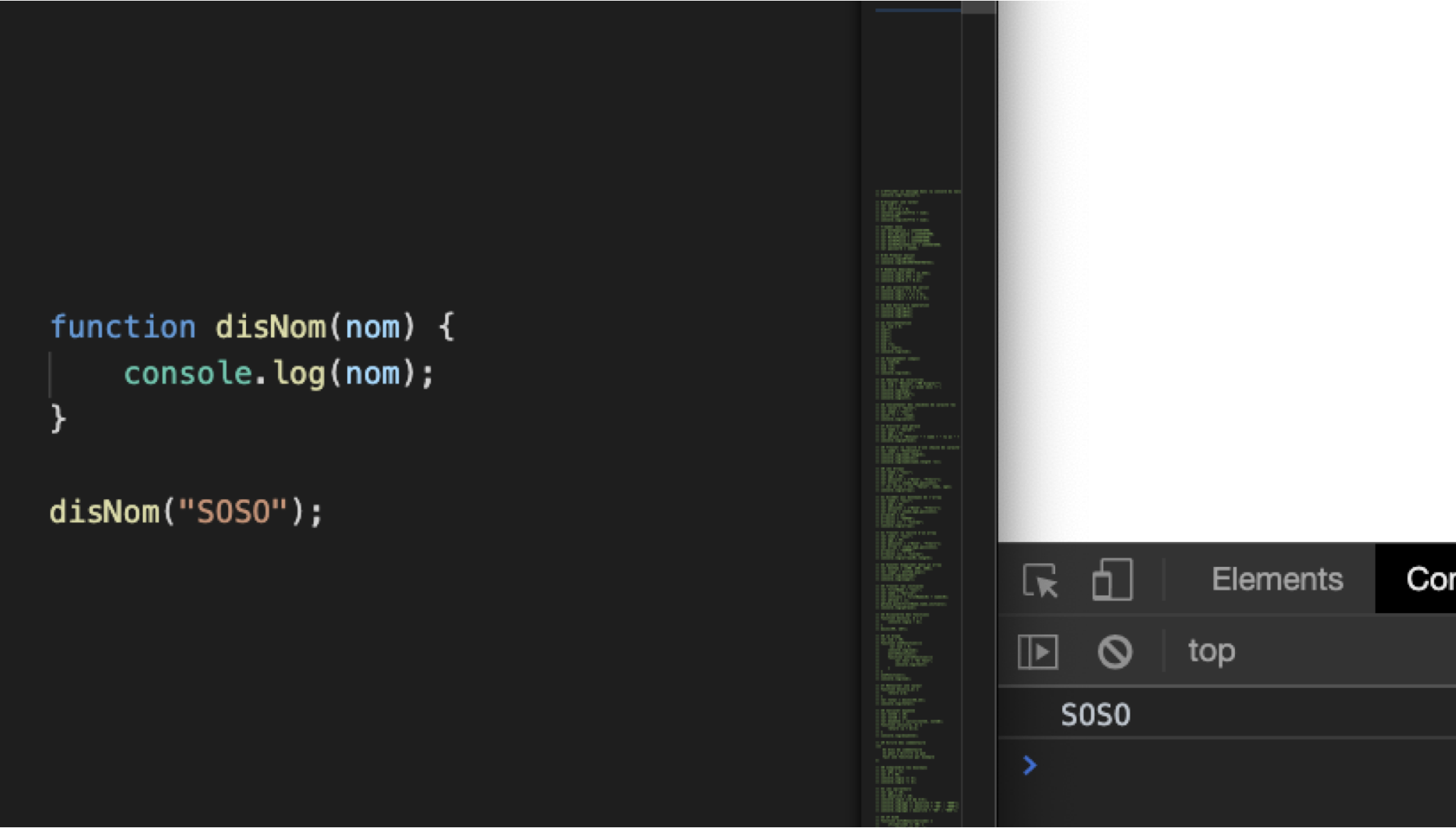
Ici la fonction moins retournera le résultat de la soustraction calculée.

Depuis ES6 le js passe un type par défaut aux paramètres non remplis « undefined »

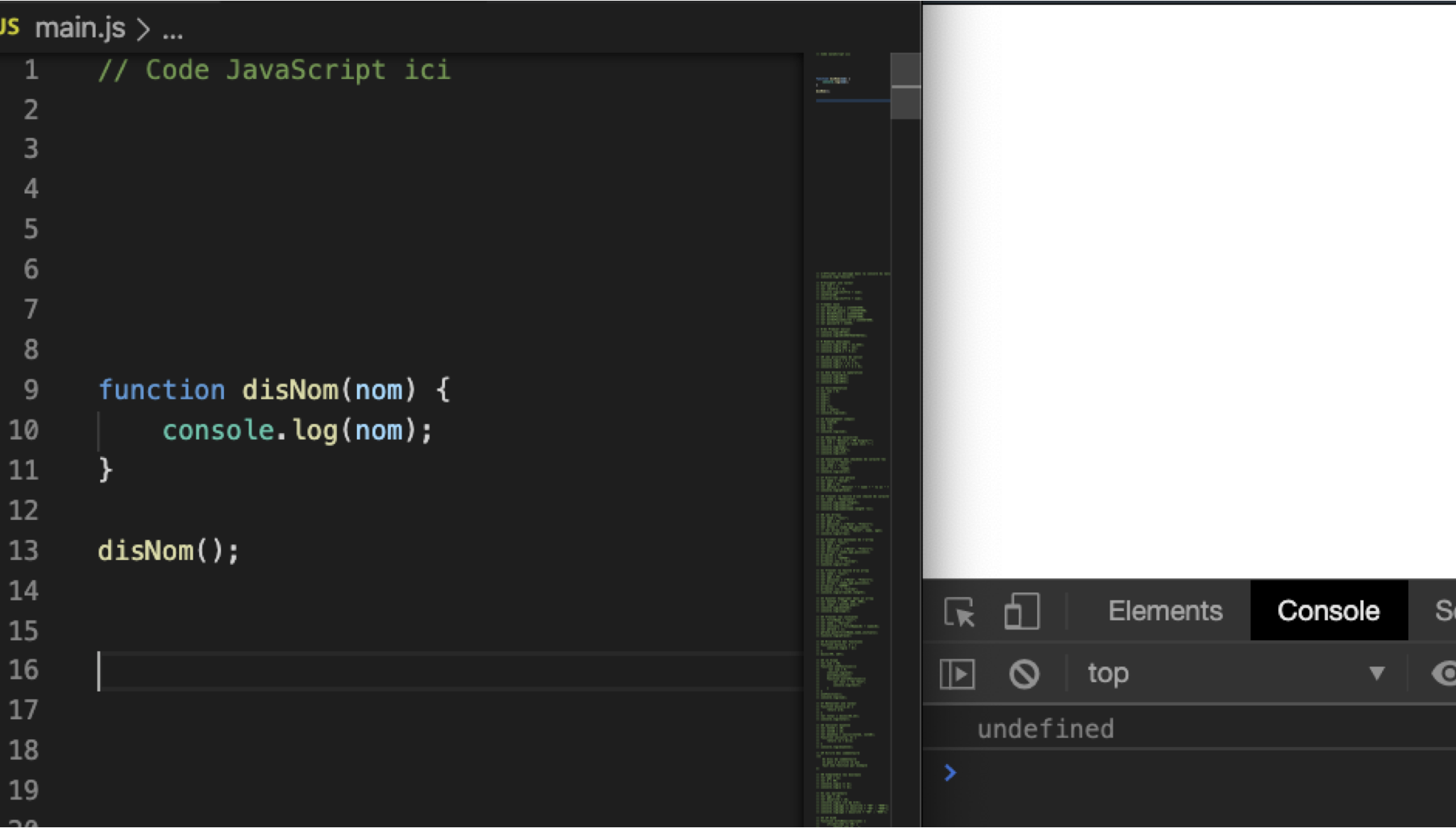
On a vu comment se servir des fonctions.

Quand une fonction attend qu’on lui passe un paramètre pour nous retourner un résultat.

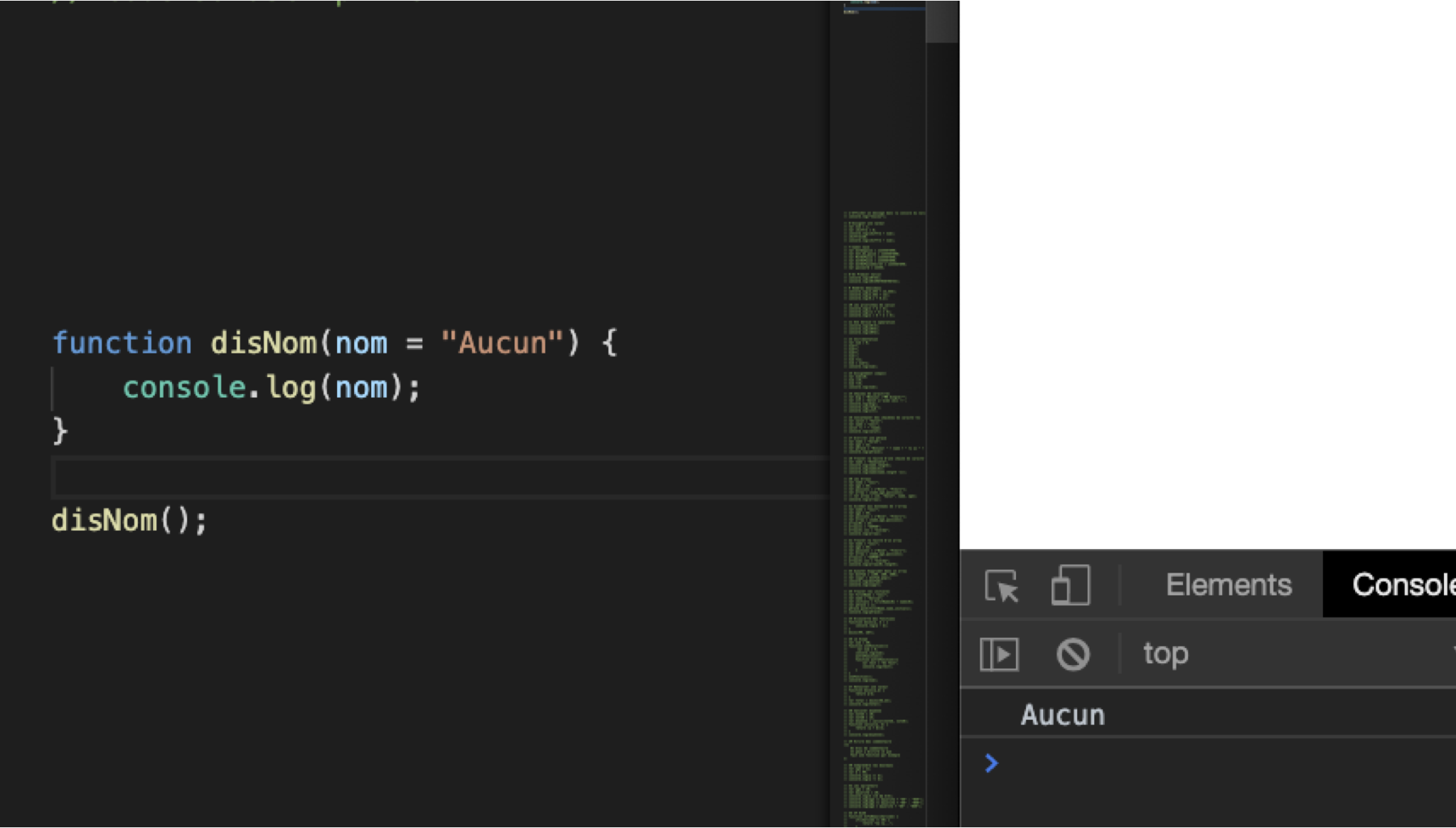
Que faire si elle ne reçoit pas de paramètre ?



Utilisation d’une fonction avec un paramètre



Si on ne lui passe pas de paramètre.



Ici quand on définit notre fonction on lui passe un paramètre par défaut.

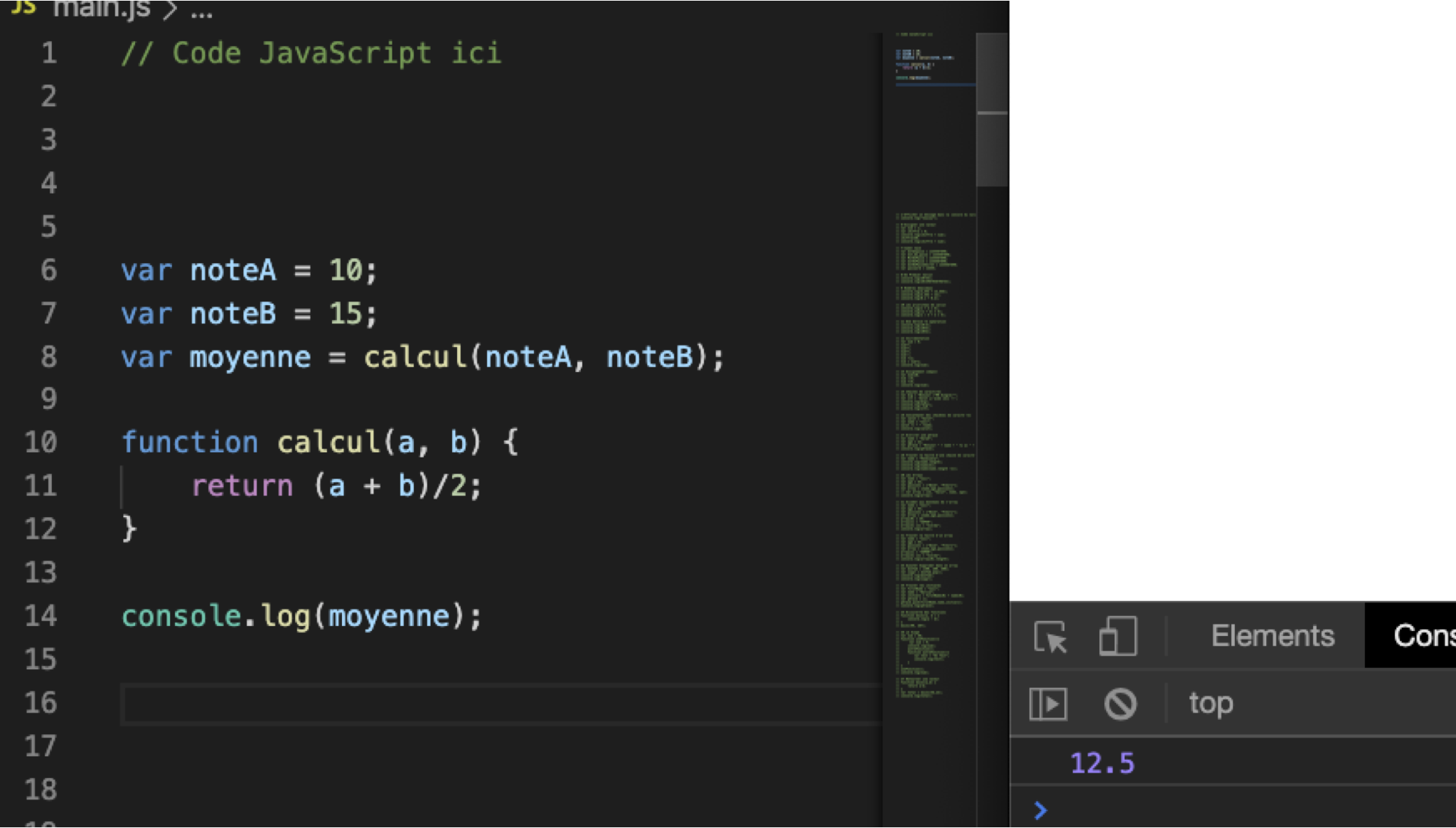
## Exercice : Calculer une moyenne

Créer 2 variables

* + 1. noteA
    2. noteB

Créer une fonction qui va calculer la moyenne et qui retourne le résultat

Solution : Exercice : Calculer une moyenne

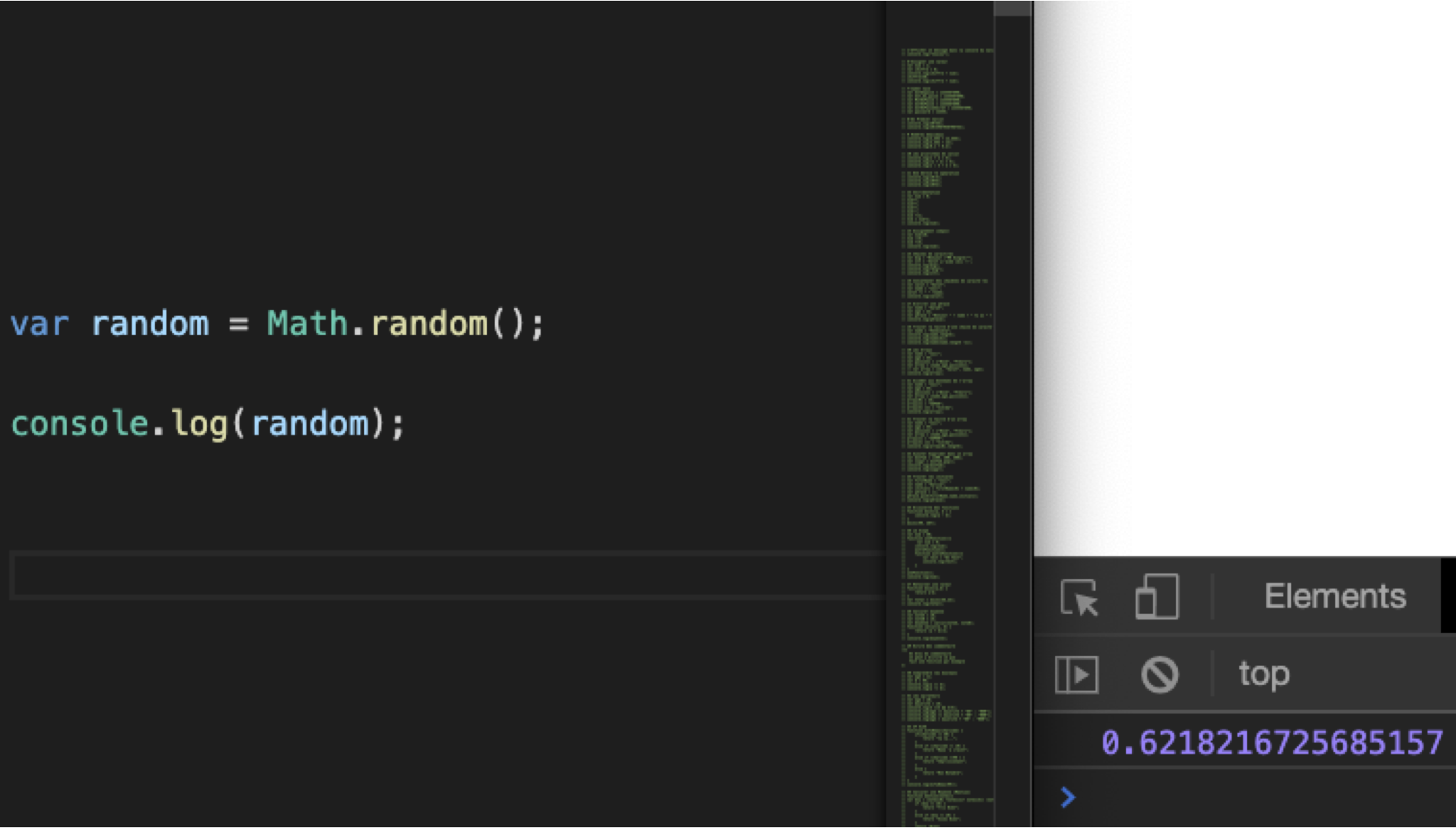


## Générer un nombre aléatoire

En JS on va utiliser l’objet Math (C’est un objet Natif de JS du coup on l’appelle **M**ath)

Gérer l’aléatoire peut être utile dans certains cas  (Jeux de hasard, loterie, exécuter des tests …)

On peut arrondir au-dessous le résultat d’un chiffre aléatoire avec Math.floor()



Exemple d’utilisation de la fonction random

## Méthode map

Pour manipuler des tableaux on a besoin de créer une boucle pour parcourir ce dernier et ainsi manipuler ses éléments.

Pour se faciliter la vie avec les tableaux, on a la méthode map()

Map facilite le parcours d’un tableau avec une syntaxe plus allégée.

Exemple : Je veux ajouter +1 à tous les éléments de mon tableau.

