

JavaScript



JavaScript

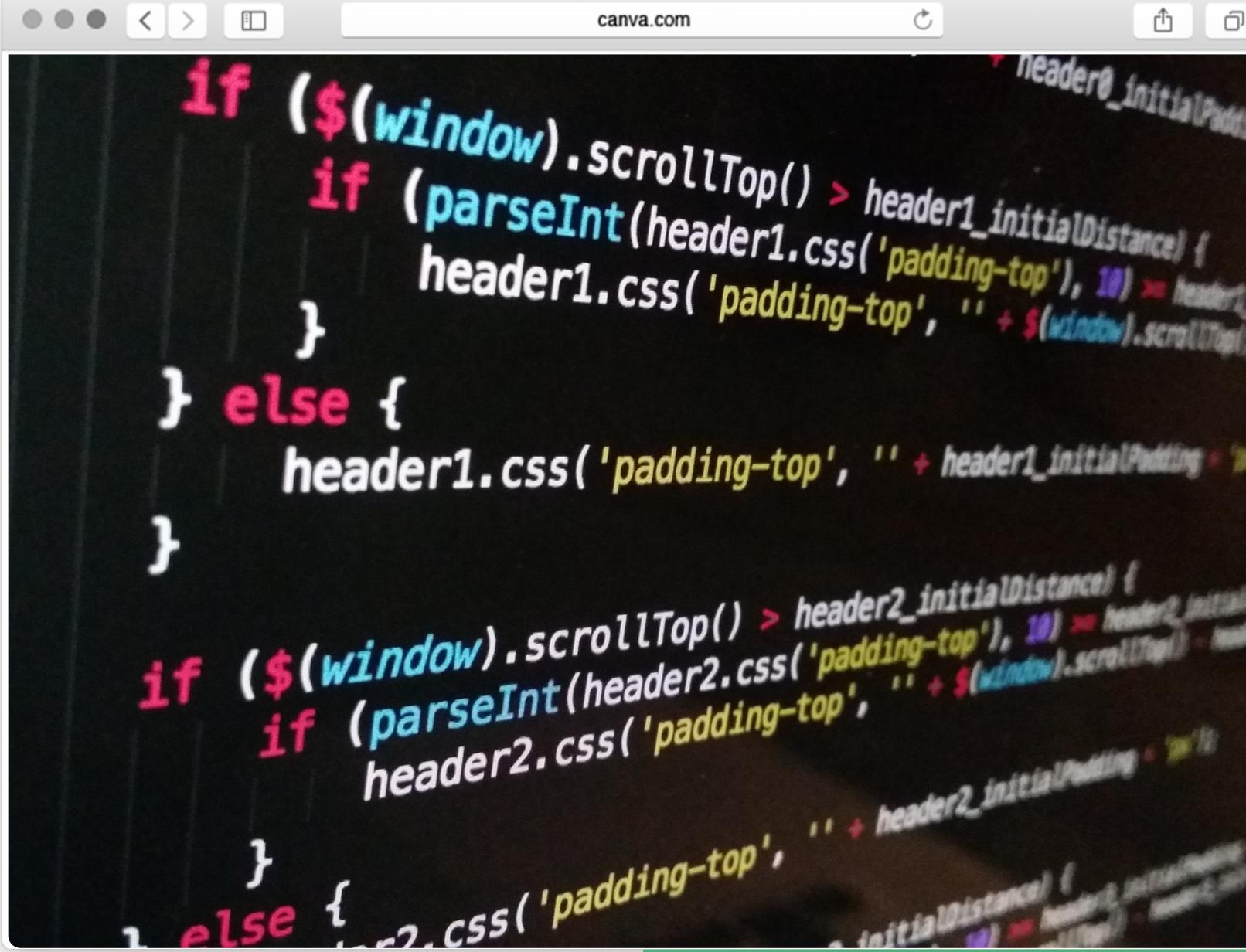
Les bases

LE JAVASCRIPT, C'EST QUOI ?

Langage de programmation dynamique

Il permet d'ajouter de l'interactivité à votre site web en offrant la possibilité de coder des jeux ou des réponses quand on clique sur certains éléments.

JAVASCRIPT



A screenshot of a web browser window titled "canva.com". The page content is a large block of JavaScript code. The code uses jQuery's '\$(\$window)' selector to check the scroll position of the window. It has two main sections: one for 'header1' and one for 'header2'. Each section contains an 'if' statement that checks if the scroll top value is greater than an initial distance. If true, it parses the current padding-top value from the CSS of the header element, adds the scroll top value to it, and then sets the new padding-top value back to the header element. If false, it simply sets the padding-top to the initial value plus the scroll top value. The code is written in a style that uses color-coded keywords and values.

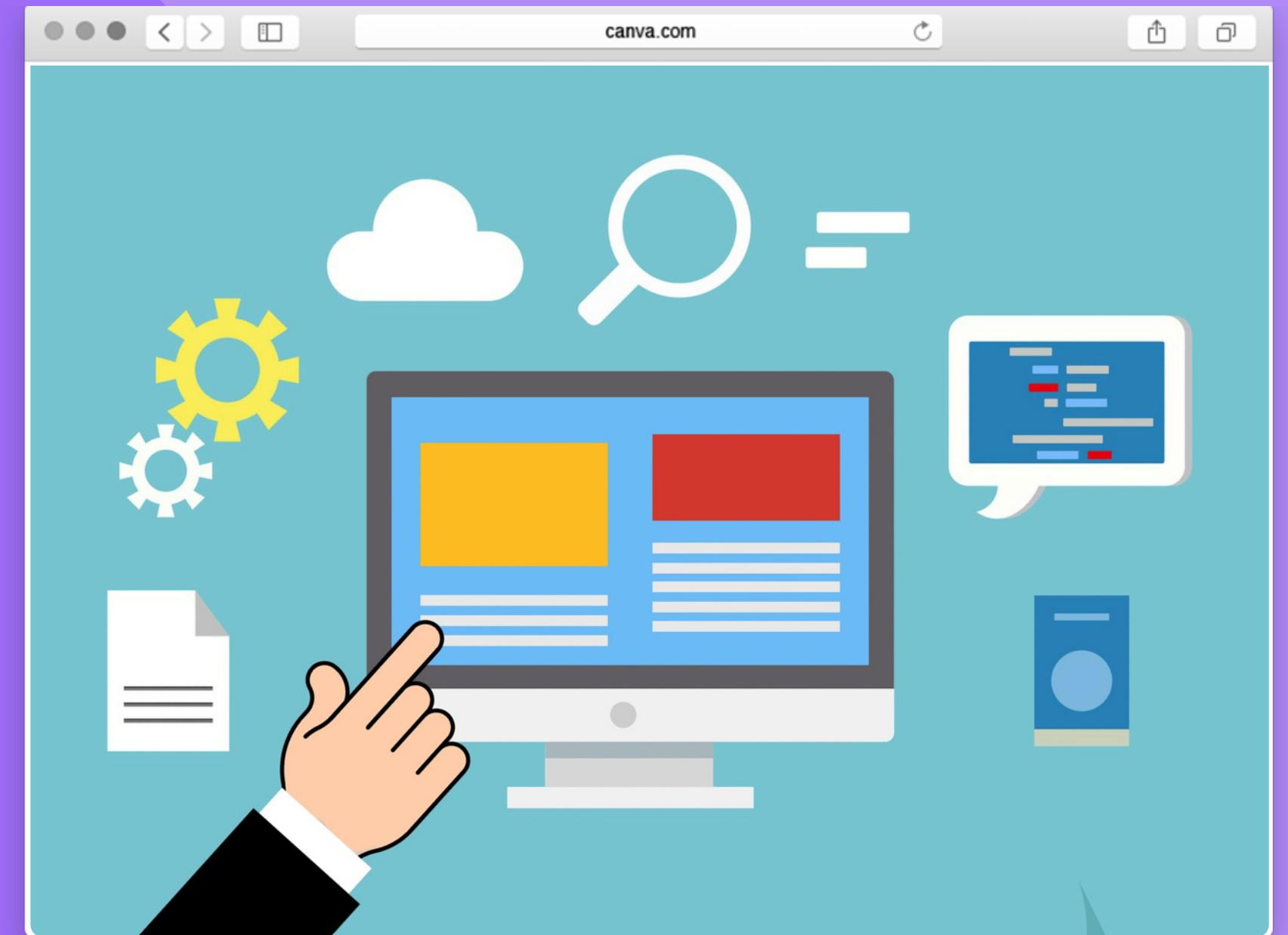
```
if ($($window).scrollTop() > header1_initialDistance) {
    if (parseInt(header1.css('padding-top'), 10) > header1_initialPadding) {
        header1.css('padding-top', '' + $($window).scrollTop() +
    }
} else {
    header1.css('padding-top', '' + header1_initialPadding + 'px')
}

if ($($window).scrollTop() > header2_initialDistance) {
    if (parseInt(header2.css('padding-top'), 10) > header2_initialPadding) {
        header2.css('padding-top', '' + $($window).scrollTop() +
    }
} else {
    header2.css('padding-top', '' + header2_initialPadding + 'px')
}
```

Abrégé JS et non Java, ce langage de programmation s'exécute côté client

JAVASCRIPT ET HTML

JavaScript vous permet d'aller très loin dans le développement de votre site et même de vos applications mobile.



01

Créer un fichier .js, par exemple script.js

02

Ajouter le fichier en bas de page avec :

```
<script src="script.js"></script>
```

FAIRE DU
JAVASCRIPT

LES COMMENTAIRES

Intégrer des commentaires dans du code JavaScript

Tout comme en HTML ou en CSS, il est possible d'intégrer des commentaires dans du code JavaScript. Pour cela, deux possibilités:

```
1 // Commentaire sur une seule ligne
2
3 /* 
4     Ce commentaire est
5     sur plusieurs
6     lignes
7 */
```

Les variables JavaScript

Une variable est une instruction qui permet de stocker des données

Une variable en JavaScript se déclare avec le mot clé LET devant. Celui-ci doit être en minuscule.

Les variables peuvent contenir différents types de données

Chaîne de caractères, nombre, booléen, tableau et objet

```
1 let maVariable = 12;  
2 let maChaine = "Chaîne de caractères";
```

Règles de nommage

Règles à respecter pour le nommage des variables :

1. Commencer par une lettre ou un underscore _
2. Peut contenir des chiffres, lettres, underscores, pas de tirets ou points
3. Pas de mots clés (new, for, while)
4. Sensible à la casse : maVar différent de mavar
5. Si plusieurs mots : majuscule sur première lettre de chaque mot sauf premier mot. Ex : monAdresse
6. Le nom doit décrire la variable
7. Prononçable
8. Aussi court que possible, mais aussi long que nécessaire

Règles de nommage

La "portée" ou le "scope"

Selon l'endroit où vous allez utiliser votre variable, son nommage sera différent.

Plus la variable a de chance d'être utilisée à divers endroits dans votre programme (sa portée), plus le nommage de celle-ci sera important.

Choisissez un nom de variable qui soit le plus explicite possible.

Une variable définie dans une condition a une portée uniquement limitée à celle-ci.

Une variable définie en dehors de toute condition ou boucle, voit sa portée étendue à l'ensemble du document.

Apprendre en pratiquant

Définir des variables et attribuer différentes valeurs

Vous devez définir :

- Une variable booléenne
- Une variable qui ne peut pas changer de valeur une fois initialisée
- Une chaîne de caractère
- Une variable numérique avec un entier
- Une variable numérique à virgule

Apprendre en pratiquant

Ecrivez un programme qui permet de prendre deux valeurs différentes en entrée renseignée par l'utilisateur (simulée : 12 et 18)

Le programme doit inverser les valeurs : la première valeur doit devenir la deuxième et inversement.

Le programme doit afficher à l'utilisateur le résultat des deux variables en indiquant clairement les valeurs.

Par exemple :

- L'utilisateur entre 6 en première valeur (simulée)
- L'utilisateur entre 3 en deuxième valeur (simulée)

Le programme inverse les deux valeurs(variables), soit :

- La première variable doit valoir 3
- La deuxième variable doit valoir 6

Apprendre en pratiquant

Calculer le prix TTC grâce à un prix HT

On souhaite calculer et afficher, à partir d'un prix hors taxe saisi, la TVA ainsi que le prix TTC.

Le montant TTC dépend :

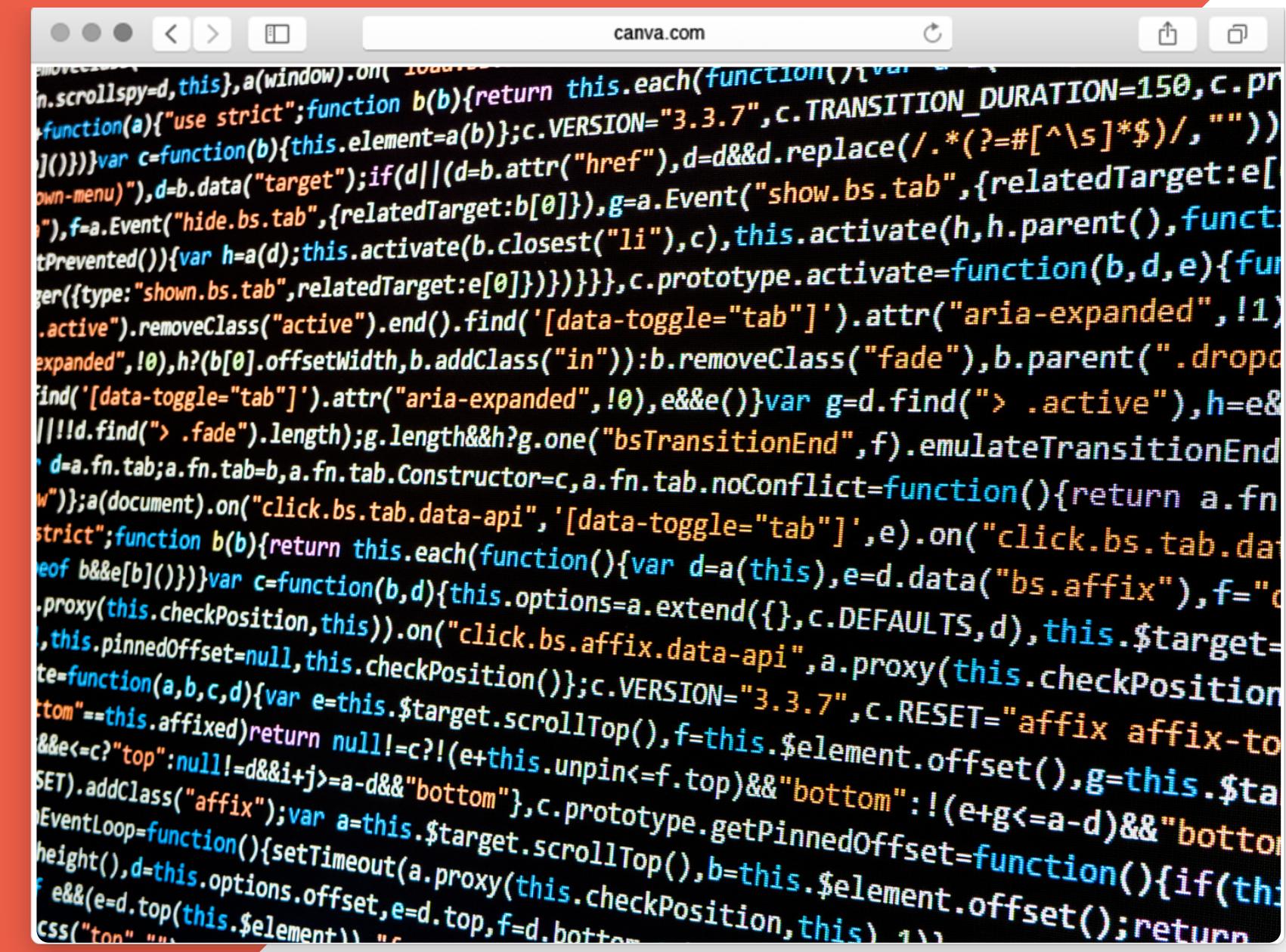
- Du prix HT
- Du taux de TVA qui est de 20%

Calcul prix TTC

```
1 const tva = 20;
2 const titre = "Résultat : ";
3
4 let prixHT, prixTTC, montantTVA;
5
6 prixHT = 130;
7
8 prixTTC = prixHT * (1 + tva/100);
9 montantTVA = prixTTC - prixHT;
10
11 console.log(titre + " " + prixHT + "€ HT + TVA " + montantTVA + "€ devient " + prixTTC + "€ TTC");
```

LES CONDITIONS

Permettent d'effectuer des actions seulement si les conditions nécessaires sont remplies



La syntaxe des conditions

Vous pouvez effectuer un ou plusieurs tests au sein d'une même condition. Pour cela, vous pouvez les inclure grâce aux mots clés ET et OU.

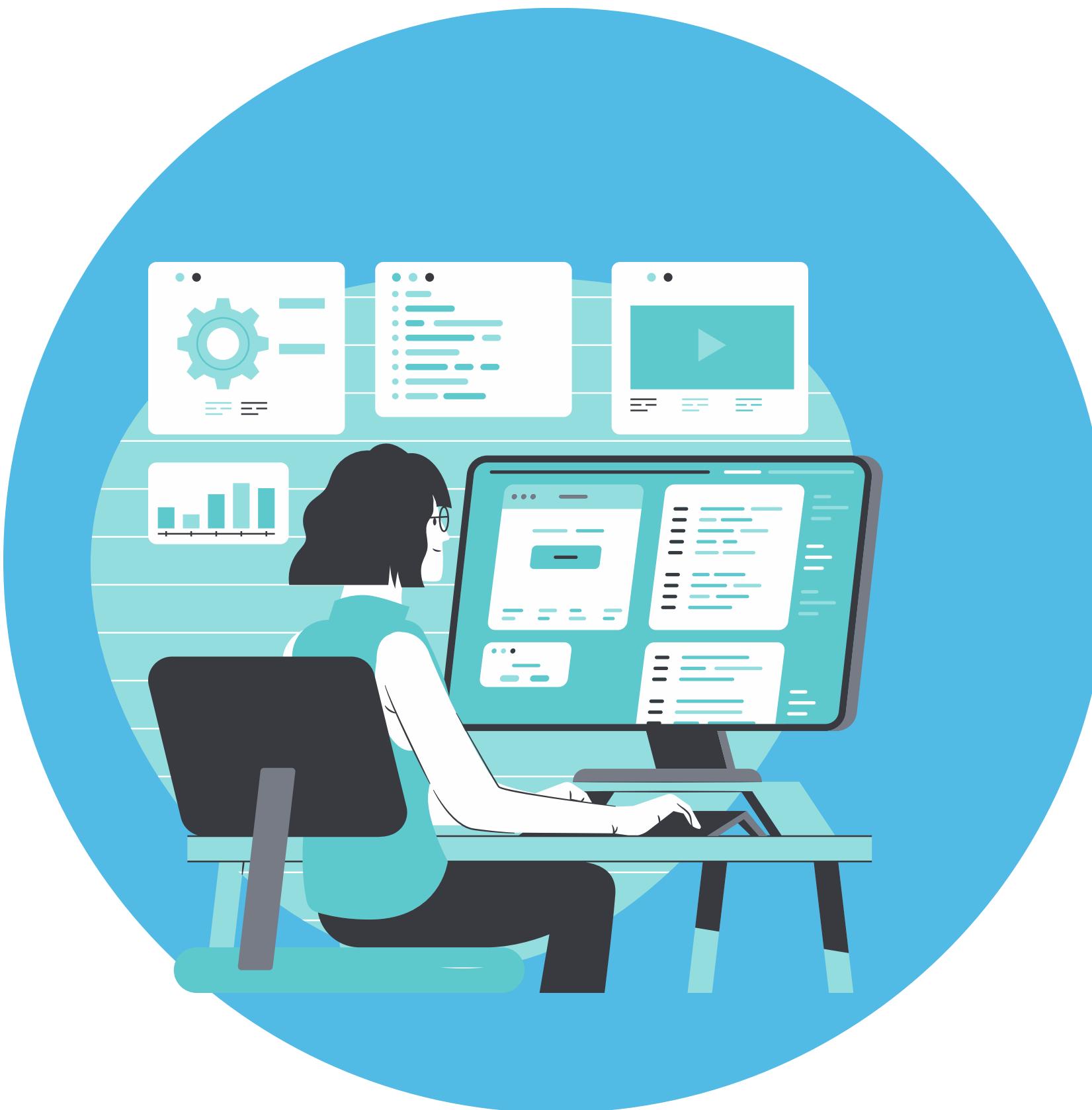
En code cela est représenté par `&&` pour ET et `||` pour OU

```
1 if(condition){  
2     // faire quelque chose  
3 } else {  
4     // Faire quelque chose d'autre  
5 }
```

Les conditions

Les conditions que vous pouvez rencontrer :

- SI... (if)
- SI.... SINON (if, else)
- Si... SINON SI... SINON (if, else if, else)



Les opérateurs de comparaison

1 <	PLUS PETIT QUE
2 >	PLUS GRAND QUE
3 <=	PLUS PETIT OU EGAL A
4 >=	PLUS GRAND OU EGAL A
5 ==	EGAL A
6 !=	DIFFERENT DE

Les conditions : exemple

```
1 // Affiche le double d'une valeur si celle-ci est inférieure
2 // au seuil donné
3 const seuil = 10;
4 let valeur = 2;
5
6 if(valeur < seuil)
7 {
8     valeur = valeur * 2;
9 }
10
11 console.log(`Voici la valeur finale : ${valeur}`)
```

LES CONDITIONS IMBRIQUÉES

Imbrication de conditions possible

Exemple d'un code qui affiche :

- "Reçu avec mention Assez Bien" si la note est supérieur ou égale à 12
- "Reçu avec mention Passable" si la note est supérieure à 10 et inférieure à 12
- "Insuffisant" dans tous les autres cas

```
1 let note = 15;  
2  
3 if(note >= 12){  
4     console.log("Reçu avec mention Assez Bien");  
5 } else if(note >= 10){  
6     console.log("Reçu mention Passable");  
7 } else {  
8     console.log("Insuffisant");  
9 }
```

Apprendre en pratiquant

Écrivez un programme qui permet de saluer l'utilisateur en fonction de l'heure

- S'il est avant 12h, afficher "Salut !"
- S'il est entre 12h et 17h, afficher "Bonjour !"
- S'il est après 17h mais avant 23h, afficher "Bonsoir"
- S'il est après 23h et avant 7h30, afficher "Vas te coucher !"

L'heure ne sera pas réel pour cet exercice, vous devez créer une variable qui simulera l'heure qu'il est. Gardez à l'esprit que pour indiquer une demi-heure il faut écrire .5. Par exemple, pour 5h30, vous écrirez 5.5.

Apprendre en pratiquant

Écrivez un programme qui permet d'entrer une valeur (simulée = 12)

Une fois la valeur entrée, le programme doit :

- Multiplier la valeur par 6 si elle est inférieure à un seuil donné (seuil = 15)
- Ne rien faire sur la valeur si celle-ci est supérieure ou égale au seuil
- Dans tous les cas, afficher à l'utilisateur le résultat de la valeur

Apprendre en pratiquant

```
1  /*
2   * Modifiez le code pour qu'il fonctionne correctement
3   */
4 let prixPanier = 325;
5 let remiseMinRequis = 150;
6 let prixFinal;
7
8 if( prixPanier >= remiseMinRequis )
9 {
10    let remise;
11
12    if( prixPanier > 150)
13    {
14        remise = 40;
15    } else if( prixPanier > 250)
16    {
17        remise = 50;
18    } else if(prixPanier > 300)
19    {
20        remise = 60;
21    }
22
23    prixFinal = prixPanier - remise;
24    console.log(`Prix final : ${prixFinal}`);
25 }
```

EXERCICE

Calculer le coût du carburant d'un voyage

En fonction des données fournies par l'utilisateur, vous devez afficher le coût du carburant pour la totalité de son voyage.

Pour cela vous avez besoin de connaître :

- La distance qu'il souhaite parcourir avec son véhicule (km)
- La consommation moyenne en carburant (L/100km)
- Le prix du litre de carburant

SÉLECTION À CHOIX MULTIPLES

Tester une variable pour effectuer une instruction en fonction de la valeur de celle-ci

Utilisation de la condition SWITCH

Ecrire simplement les conditions sans imbriques

Ex. On souhaite afficher un message selon le sexe, on peut utiliser SWITCH

```
1 let genre;
2
3 switch(genre){
4     case "M":
5         console.log("Monsieur");
6         break;
7
8     case "Mme":
9         console.log("Madame");
10        break;
11
12    case "Mlle":
13        console.log("Mademoiselle");
14        break;
15
16    default:
17        console.log("Non genre/Autre")
18 }
```

Apprendre en pratiquant

Utilisez le SWITCH pour

- Tester si la variable nb1 est égale à 8
 - Si c'est le cas, afficher : Le nombre 1 vaut 8
- Tester si la variable nb1 est égale à 10
 - Si c'est le cas, afficher : Le nombre 1 vaut 10
- Tester si la variable nb1 est égale à 12
 - Si c'est le cas, afficher : Le nombre 1 vaut 12
- Si on n'entre dans aucun des cas afficher : On n'entre pas dans les autres conditions

Apprendre en pratiquant

Vous allez utiliser le SWITCH afin de tester différents niveaux de compte d'un utilisateur

Pour cela, n'oubliez pas de définir la variable que vous allez tester.

Dans le cas où le compte est :

- Abonné
 - Afficher : Vous êtes un abonné.
- Contributeur
 - Afficher : Vous êtes contributeur.
- Administrateur
 - Afficher : Vous êtes administrateur.
- Aucune des conditions ci-dessus :
 - Afficher : Ce type de compte n'est pas connu

LA BOUCLE FOR

S'utilise lorsque vous avez besoin de répéter une itération plusieurs fois.

La boucle FOR évite d'utiliser des conditions SI par centaine, voir plus en fonction du nombre de répétitions.

La boucle FOR permet d'effectuer le travail de test à votre place.

```
1  /*
2   * Afficher tous les nombres impairs
3   * de 1(inclu) jusqu'à 100
4  */
5  for(let i = 1; i < 100; i = i + 2){
6    console.log(`Nombre impair : ${i}`);
7 }
```

LA BOUCLE FOR

```
1 for(définition compteur; condition; incrémentation){  
2     // faire quelque chose  
3 }  
4  
5 // Exemple  
6 for(let i = 0; i < 10; i++){  
7     console.log(`La valeur de i est ${i}`);  
8 }
```

Apprendre en pratiquant

Avec une boucle pour (for), vous allez :

- Afficher tous les nombres pairs de 0 à 100
- 0 inclus (c'est compté comme un nombre pair)
- 100 inclus (le 100 doit également être affiché)

Apprendre en pratiquant

Avec la boucle pour (for), écrivez un programme qui affiche la table de multiplication d'un nombre (jusqu'à 10)

- Le nombre doit absolument être renseigné dans une variable qui pourra changer facilement

Exemple de ce que doit afficher le programme :

$$1 \times 5 = 5$$

$$2 \times 5 = 10$$

$$3 \times 5 = 15$$

$$4 \times 5 = 20$$

...

$$10 \times 5 = 50$$

Apprendre en pratiquant

Avec la boucle pour (for), écrivez un programme qui affiche la table de multiplication d'un nombre jusqu'à x

- Le nombre doit absolument être renseigné dans une variable qui pourra changer facilement
- X c'est une valeur qui pourra être choisie par l'utilisateur
- Pour l'instant, X doit être une variable simulée

Exemple de ce que doit afficher le programme :

Variable x = 25

$$1 \times 5 = 5$$

$$2 \times 5 = 10$$

$$3 \times 5 = 15$$

$$4 \times 5 = 20$$

...

$$25 \times 5 = 125$$

La boucle tant que

S'utilise lorsque vous avez besoin de répéter une itération plusieurs fois.

La boucle WHILE évite d'utiliser des conditions SI par centaine, voir plus en fonction du nombre de répétitions.

La boucle WHILE permet d'effectuer le travail de test à votre place.

```
1 let i = 0; // amorce
2
3 while(condition){
4
5     i++; // incrémentation/modification variable de condition
6 }
```

LA BOUCLE WHILE

```
1 let i = 0;  
2  
3 while(i < 10){  
4     console.log(`La valeur de i est ${i}`);  
5     i++;  
6 }
```

Apprendre en pratiquant

Avec une boucle tant que (while), vous allez :

- Afficher tous les nombres pairs de 0 à 100
- 0 inclus (c'est compté comme un nombre pair)
- 100 inclus (le 100 doit également être affiché)

Aidez vous de la boucle for pour construire votre boucle while

Apprendre en pratiquant

Ecrivez un programme qui affiche la table de multiplication d'un nombre (jusqu'à 10)

- Le nombre doit être une variable qui pourra changer facilement
- Utilisez la boucle while

Apprendre en pratiquant

Écrivez un programme qui affiche la table de multiplication d'un nombre jusqu'à X

- X = valeur choisie par utilisateur
- Pour l'instant X doit être une variable simulée
- Utilisez la boucle While

Apprendre en pratiquant

Écrivez un programme qui affiche la figure ci-dessous

```
A  
A A  
A A A  
A A A A  
A A A A A  
A A A A A A  
A A A A A A A  
A A A A A A A A
```

- Utilisez la boucle WHILE



Les tableaux

Un tableau est une variable qui permet de stocker plusieurs données à la fois.

L'indice commence à 0.

CRÉER UN TABLEAU

```
1 // Création du tableau
2 let monTableau = [15, 18, 33, "Ma chaine"];
3
4 // Connaitre la quantité de données du tableau
5 console.log(monTableau.length); // 4
6
7 // Afficher le 3e valeurs
8 console.log(monTableau[2]);
```

Apprendre en pratiquant

Créer un tableau qui contient 5 nombres

Apprendre en pratiquant

Créer un tableau qui contient 5 données :

- Vous devez avoir au moins 1 booléen
- Vous devez avoir au moins 1 chaîne de caractère
- Vous devez avoir au moins 1 nombre décimal



Les objets

Un objet fonctionne comme un tableau, à la différence qu'il ne possède pas d'indice.

Chaque case du tableau est identifiée par un nom

CRÉER UN OBJET

```
1 // Cr ation de l'objet
2 let monObjet = {
3     marque: "Renault",
4     modele: "Clio",
5     annee: 2008,
6     bonEtat: true
7 };
8
9 // Afficher votre objet
10 console.log(monObjet);
11
12 // Afficher le mod le
13 console.log(monObjet.modele);
14
15 // Ajouter une donn e
16 monObjet.vente = "En cours";
```

Apprendre en pratiquant

Créer un objet pour représenter un étudiant. L'objet devra contenir les informations suivantes :

- Nom : string
- Prénom : string
- Age : integer
- Formation : string
- Notes : table

la boucle pour chaque

Si vous possédez un tableau ou un objet pour lequel vous souhaitez parcourir tous les éléments, vous pouvez utiliser la boucle pour chaque : forEach().

```
1 let monTableau = [10, 15, 32, "Tutu"];
2
3 monTableau.forEach((element) => {
4     console.log(`Donnée du tableau : ${element}`);
5 })
```

Apprendre en pratiquant

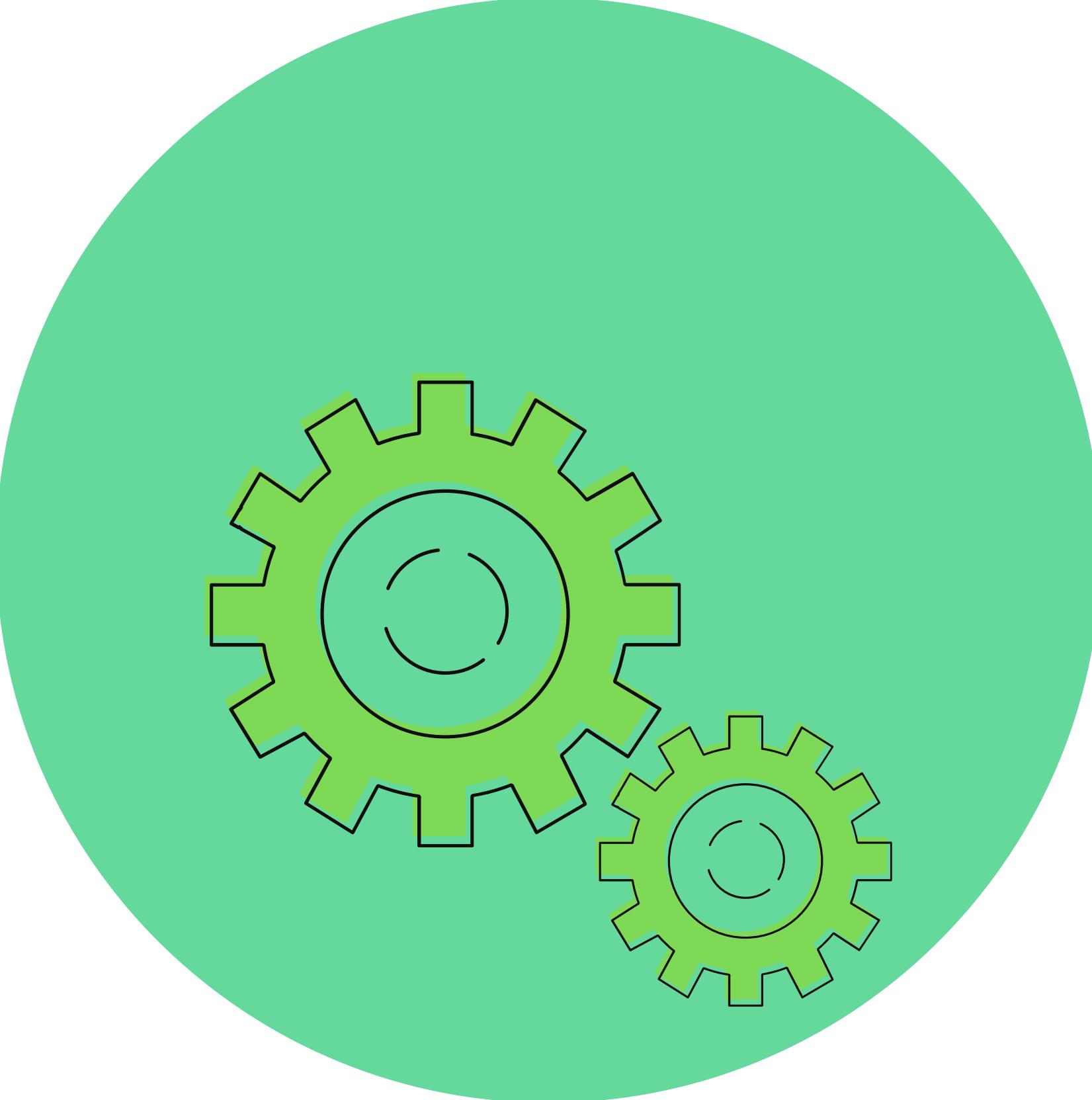
Avec une boucle pour chaque, vous allez :

- Lire et afficher tous les éléments qui se trouvent dans un tableau
- Compter le nombre d'élément dans le tableau

Apprendre en pratiquant

Avec une boucle pour chaque, vous allez :

- Compter le nombre de notes supérieures ou égales à 10
- Faire la moyenne de toutes les notes
- Afficher le nombre de notes supérieures ou égales à 10
- Afficher la moyenne



Les fonctions

Un "sous-programme" qui permet d'effectuer des tâches répétitives.

Évite de créer plusieurs fois le même code.

Exemple de fonctions

Quelques exemples de fonctions :

1. `length` // la longueur d'un tableau ou d'une chaîne de caractère
2. `Math.floor()` // Permet de renvoyer un entier
3. `Math.random()` // Générer un nombre aléatoire (nombre à virgule)
4. `toFixed()` // Arrondir à X chiffres après la virgule
5. `push()` // permet d'ajouter une valeur à un tableau

Apprendre en pratiquant

Créer une fonction qui permet d'additionner 3 nombres.

Apprendre en pratiquant

Créer une fonction qui permet de calculer une moyenne.

INFORMATIONS :

- Utilisez un tableau afin de faire passer les notes en argument

Apprendre en pratiquant

Créer une fonction qui permet de créer un étudiant avec les données suivantes :

- Nom : string
- Prénom : string
- Age : integer
- Formation : string
- Notes : table

Apprendre en pratiquant

Créer une fonction qui permet de créer un étudiant et de l'ajouter à la liste des étudiants.

Un étudiant doit avoir les données suivantes :

- Nom : string
- Prénom : string
- Age : integer
- Formation : string
- Notes : table

INFORMATIONS :

- La liste des étudiants doit être enregistrée dans un objet "ListeEtudiants"



Manipulation du DOM

Avec JavaScript vous pouvez manipuler le DOM simplement.

Modifier du contenu, supprimer du contenu, ajouter du contenu.

MANIPULER LE DOM

Pour manipuler le DOM vous devez sélectionner l'élément souhaité :

- `document.querySelector('selecteur CSS')`
- `document.querySelectorAll('selecteur CSS')`
- `document.getElementById('id sans le #')`
- `document.getElementsByClassName('nom de classe sans .')`
- `document.getElementsByTagName('balise')`

MANIPULER LE DOM

Vous pouvez modifier directement le contenu d'un élément :

- `element.textContent = 'Texte'`
- `element.innerText = "Texte"`
 - Modification TEXTUELLE du contenu d'un élément
- `element.innerHTML = '<p>Texte et HTML</p>'`
 - Modification HTML du contenu d'un élément

MANIPULER LE DOM

Vous pouvez créer et supprimer des éléments :

- `document.createElement()`
 - Permet de créer un nouvel objet html (p, div, span, etc.)
- `element.append()`
 - Ajoute un nouvel élément à la fin de la sélection
- `element.before()`
 - Ajoute un élément avant la sélection
- `element.remove()`
 - Supprime un élément

ÉCOUTER DES ÉVÈNEMENTS

Vous pouvez réagir à un évènement avec JS représenté par :

- Un nom (click, mousemove, etc.)
- Une fonction appelée callback
 - Exécutée au déclenchement de l'évènement

Écouter un évènement :

- addEventListener()
 - Écouter tous types d'évènements
 - Prend en paramètres le type de l'évènement et le callback

ÉCOUTER DES ÉVÈNEMENTS

```
1 // Sélectionner l'élément sur lequel on veut détecter le click
2 const el = document.querySelector('button');
3
4 // Ecouter le click
5 el.addEventListener("click", () => {
6   console.log(`L'utilisateur à cliqué sur le bouton`);
7 })
```

ÉCOUTER DES ÉVÈNEMENTS

Vous pouvez éviter le comportement par défaut d'un élément (ne pas prendre en compte le clique sur un lien ou l'envoi d'un formulaire).

```
1 // Sélectionner l'élément sur lequel on veut détecter le click
2 const el = document.querySelector('button');
3
4 // Ecouter le click
5 el.addEventListener("click", (e) => {
6   console.log(`L'utilisateur à cliqué sur le bouton`);
7   e.preventDefault();
8 })
```

ÉCOUTER DES ÉVÈNEMENTS

Récupérer du texte saisi dans un formulaire grâce à l'écouteur d'évènements.

```
1 // Sélectionner le champ texte
2 const el = document.querySelector('input');
3
4 // Ecouter les modification sur le champ
5 el.addEventListener("input", (e) => {
6     // récupérer la valeur du champ
7     let texte = e.target.value;
8     console.log(`le texte renseigné : ${texte}`)
9 })
```