



ES3/5

Bases et techniques avancées

Séries d'exercices didactiques

Précède : ES3/5 - « Design Patterns »



Table des matières

1. Syntaxe et Algorithmie en ES3/5	3
a. Bases niveau 1.....	3
b. Bases niveau 2.....	13
c. Intermédiaire	17
d. Perfectionnement	17
2. Fondamentaux et techniques avancées en ES3/5	17
a. Les Objets	17
b. Le Prototypage	17
c. Les Objets fondamentaux.....	17
d. Les Modèle Objet du Document	17
e. Comprendre certaines techniques avancées	18

1. Syntaxe et Algorithmie en ES3/5



Dans cette partie, nous ignorerons volontairement la notion d'objet en ES3/5 pour nous concentrer sur la syntaxe et l'algorithmie en ES3/5 (ainsi le terme « fonction » sera préféré au terme « méthode », le terme « tableau » sera préféré au terme « objet de type tableau », etc.)

a. Bases niveau 1 :

Objectif : Savoir utiliser différents types de données

Exercice 1/11 : Utiliser des variables et des « fonctions »

- Préparation :
 - Créez un document `HTML` valide contenant un paragraphe de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - Une variable permet de stocker en mémoire une chaîne de caractère ou un nombre. Une variable est un mot employé pour désigner un emplacement mémoire. Pour créer une variable on utilise le mot-clé `var`. Par exemple :
`var emplacement;`
 - Une « fonction » est un bloc de code stocké en mémoire qu'on peut appeler autant de fois que nécessaire. Le développeur peut créer une ou plusieurs « fonctions ». Le moteur du langage propose de nombreuses « fonctions » prêtes à l'emploi. Nous utiliserons une « fonction » fournie de base. La « fonction » stockée sous le nom de variable `alert`
 - Lorsque le code contenu dans cette « fonction » est exécuté, le navigateur Internet affiche une boite de dialogue.
 - Pour demander l'exécution du code contenu dans cette « fonction » on peut écrire le code suivant :
`alert();`
 - Si on fournit une valeur à la « fonction » `alert` lors de son exécution, alors le code de la « fonction » reçoit cette valeur et la boite de dialogue affiche la valeur fournie sous forme d'une chaîne de caractères. Par exemple :
`alert("soleil");`
 - Si on fournit une variable à la « fonction » `alert` lors de son exécution, alors le code de la « fonction » reçoit la valeur correspondant à la variable et la boite de dialogue affiche la valeur fournie sous forme d'une chaîne de caractères. Par exemple :
`alert(emplacement);`
- Enoncé :
 - Affichez dans une boite de dialogue le mot *bonjour* au moyen de la « fonction » `alert()` ;

- Créez une 1^{ère} variable à laquelle vous assignerez une chaîne de caractère. Affichez la valeur vers laquelle pointe cette variable dans une boîte de dialogue.
- Créez une 2^{ème} variable à laquelle vous assignerez une chaîne de caractère *L'activité de ce début de journée, c'est l'apprentissage des variables en JavaScript*. Affichez la valeur vers laquelle pointe cette variable dans une boîte de dialogue.

Exercice 2/11 : Utiliser des opérateurs arithmétiques

- Préparation :
 - Créez un document `HTML` valide contenant un paragraphe de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - Les opérateurs arithmétiques disponibles en ES3/5 sont :
 1. `+` pour l'addition, attention à la confusion avec la concaténation.
 2. `-` pour la soustraction
 3. `/` pour la division, attention aux divisions par 0.
 4. `+` pour l'addition.
 5. `%` pour le reste entier de la division, attention prend toujours le signe du dividende (nombre divisé).
 6. `++` pour l'incrémentation.
 7. `--` pour la décrémentation.
 - 8. D'autres opérateurs, que nous étudierons plus tard, sont disponibles dans des versions plus récentes de l'ECMAScript.
- Enoncé :
 - Créez 3 variables avec 3 noms différents. Chacune de ces variables pointe vers un nombre différent de votre choix. Assignez la somme de ces 3 variables dans une 4^{ème} variable. Affichez la valeur vers laquelle pointe la 4^{ème} variable dans une boîte de dialogue.
 - Multipliez la valeur vers laquelle pointe la 4^{ème} variable par le nombre 5. Assignez le résultat de cette opération à une 5^{ème} variable. Affichez la valeur vers laquelle pointe la 5^{ème} variable dans une boîte de dialogue.
 - Divisez la valeur vers laquelle pointe cette 5^{ème} variable par 3. Assignez le résultat de cette opération à une 6^{ème} variable. Affichez la valeur vers laquelle pointe la 6^{ème} variable dans une boîte de dialogue.

Exercice 3/11 : Utiliser des opérateurs arithmétiques, suite : le reste

- Préparation :

- Créez un document `HTML` valide contenant un paragraphe de texte ;
- Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
- Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - L'opérateur `%` calcule le reste entier d'une division, attention il prend toujours le signe du dividende (nombre divisé). Par exemple :
 1. `7 % 2`; donne 1
 2. `29 % 25`; donne 4
 3. `80 % 20`; donne 0
 - Les développeurs appellent habituellement l'opérateur `%` « modulo » en référence aux mathématiques. C'est incorrect, un « modulo » ne prend pas toujours le signe du dividende (nombre divisé). C'est pourquoi l'opérateur `%` est qualifié de « reste » et non « modulo » dans la documentation officielle du langage.
- Enoncé :
 - Créez 4 variables.
 - Assignez à la 1^{ère} variable la valeur 90 ;
 - Assignez à la 2^{ème} variable la valeur 11 ;
 - Assignez à la 3^{ème} variable la valeur 20 ;
 - Assignez à la 4^{ème} variable la valeur 9 ;
 - Le diviseur est toujours la 1^{ère} variable (le diviseur est la variable à droite de l'opérateur).
 - Calculez le reste de la division de la 2^{ème} variable par la 1^{ère} et affichez le résultat dans une boîte de dialogue.
 - Calculez le reste de la division de la 3^{ème} variable par la 1^{ère} et affichez le résultat dans une boîte de dialogue.
 - Calculez le reste de la division de la 4^{ème} variable par la 1^{ère} et affichez le résultat dans une boîte de dialogue.
 - Que remarquez-vous ?

Exercice 4/11 : Utiliser l'opérateur de concaténation

- Préparation :
 - Créez un document `HTML` valide contenant un paragraphe de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - On peut assembler une chaîne de caractères à partir de plusieurs chaînes de caractères. On appelle ce type d'opération la concaténation.

- Pour effectuer une concaténation on utiliser l'opérateur `+`. Par exemple :
`"de" + "main"; donne "demain"`
- Enoncé :
 - Créez 4 variables.
 - Assignez à la 1^{ère} variable le texte : *La nuit*
 - Assignez à la 2^{ème} variable le texte : *tous les*
 - Assignez à la 3^{ème} variable le texte : *chats*
 - Assignez à la 4^{ème} variable le texte : *sont gris*
 - Concaténez les 4 variables et assignez le résultat à une 5^{ème} variable.
 - Affichez la valeur vers laquelle pointe la 5^{ème} variable dans une boite de dialogue.

Exercice 5/11 : Utiliser l'opérateur de concaténation - suite

- Préparation :
 - Idem
- Présentation :
 - Idem
- Enoncé :
 - Créez 3 variables.
 - La 1^{ère} pointe vers une chaîne de caractères.
 - La 2^{ème} pointe vers un nombre.
 - La 3^{ème} pointe vers un nombre.
 - Concaténez la 1^{ère} variable avec la 2^{ème} et assignez le résultat à une 4^{ème} variable.
 - Concaténez la 4^{ème} variable avec la 3^{ème} et assignez le résultat à une 5^{ème} variable.
 - Affichez la valeur vers laquelle pointe la 5^{ème} variable dans une boite de dialogue.
 - Quel résultat obtenez-vous et pourquoi à votre avis ?

Exercice 6/11 : Utiliser des « tableaux »

- Préparation :
 - Idem
- Présentation :
 - On peut assigner à une variable des valeurs dites primitives (nombres, chaînes de caractères ou booléens) mais également des valeurs plus complexes.
 - Nous allons nous intéresser à la notion de « tableau ». Un « tableau » peut contenir un ensemble de valeurs qui vont être indexées de 0 à la valeur souhaitée.
 - En ECMAScript les « tableaux » n'existent pas, il s'agit en réalité d'objets construits à l'aide de la fonction constructeur fondamentale `Array`. Mais

pour l'instant, et dans le but de simplifier l'apprentissage, nous appellerons ce type d'objets « tableau ».

- Un « tableau » peut se déclarer à l'aide des caractères [et]. Par exemple :

```
["lunette", 15, 3.7, "soleil"];
```
- On peut assigner un « tableau » à une variable comme n'importe quelle valeur. Par exemple :
 1.

```
var monTableau;
```
 2.

```
monTableau = ["lunette", 15, 3.7, "soleil"];
```
- Pour pointer sur une valeur dans un « tableau », on utilise son indice en comptant à partir de 0. Par exemple : `monTableau[3]`; pointe sur "soleil".
- La valeur entre crochet permet de pointer sur une valeur du « tableau ».
- N'oubliez jamais que le premier indice d'un « tableau » est toujours 0. Par exemple : `monTableau[0]`; pointe sur "lunette".

- Enoncé :

- Créez une 1^{ère} variable.
- Assignez le « tableau » : `["un", 2, "trois", 4.5, "un dernier texte"]` à cette variable.
- En manipulant la variable pointant vers ce « tableau » :
 1. Affichez la valeur trois dans une boite de dialogue
 2. Affichez la valeur 4.5 dans une boite de dialogue
 3. Calculez la somme des deux nombres du « tableau » et affichez le résultat dans une boite de dialogue.
- Créez une 2^{ème} variable.
- Assignez le « tableau » : `["Il", "fait", "beau"]` à cette variable.
- En manipulant la variable pointant vers le « tableau » précédent :
 1. Concaténez les 3 chaînes de caractères contenues dans le « tableau » et assignez le résultat à une 3^{ème} variable. N'oubliez pas d'ajouter les espaces et un point à la fin de la phrase que vous êtes en train de créer.
 2. Affichez le résultat dans une boite de dialogue.

Exercice 7/11 : Utiliser des « tableaux » - suite

- Préparation :
- Idem
- Présentation :
- Il est possible d'ajouter des valeurs à un « tableau » après l'avoir créé. Par exemple :
 1.

```
var monTableau;
```
 2.

```
monTableau = [15, "une information", 24];
```
 3.

```
monTableau[3] = "un autre info";
```

- Enoncé :
 - Créez une 1^{ère} variable.
 - Assignez le « tableau » : `["lundi", "mardi", "mercredi", "jeudi", "vendredi"]` à cette variable.
 - Puis, ajoutez les 2 derniers jours de la semaine au « tableau » sans modifier la déclaration initiale.
 - Affichez le dernier indice du « tableau » dans une boîte de dialogue.
 - Concaténez la 1^{ère} et la dernière valeur du « tableau » sans oublier de les séparer par un espace.
 - Ajoutez la valeur obtenue au « tableau ».
 - Affichez dans une boîte de dialogue la dernière valeur du « tableau ».

Exercice 8/11 : Utiliser des « tableaux » - suite

- Préparation :
 - Idem
- Présentation :
 - Un « tableau » peut contenir des types primitifs comme des nombres, booléens et chaînes de caractères. Il peut également contenir 1 ou plusieurs autres « tableaux ». Par exemple : `var monTableau = ["fraise", "melon", "orange", "pomme", ["Golden", "Gala", "Pink lady"], "poire"];`
 - Si on souhaite pointer sur le texte "Gala", on écrira alors `monTableau[4][1]` où le premier indice, 4, est relatif au « tableau » et le second indice, 1, au « sous-tableau » dans le « tableau » à l'indice 4.
 - Bien entendu, il n'y a pas de limite. Un « tableau » peut contenir un « tableau » contenant lui-même un « tableau » ...
- Enoncé :
 - Créez une 1^{ère} variable.
 - Assignez le « tableau » : `["Berlin", ["Mur de Berlin", "Porte de Brandebourg", "Château de Charlottenburg"], "Paris", ["Notre-Dame, Tour-Eiffel", "Beaubourg", "Opera", "Palais du Luxembourg"], "Rome", ["Forum", "Colisée", "Chapelle Sixtine", "Pantheon"]]`; à cette variable.
 - En utilisant ce « tableau », affichez dans une boîte de dialogue le 2^{ème} monument de Berlin.
 - En utilisant ce « tableau », affichez dans une boîte de dialogue le 5^{ème} monument de Paris.
 - En utilisant ce « tableau », affichez dans une boîte de dialogue le 3^{ème} monument de Rome.
 - Créez une 2^{ème} variable.

- Assignez le « tableau » : [40, 58, [478, 85, 745, 8], 74, [72, 14], [80, 741, 97]]; à cette variable.
- Vous allez devoir faire, à trois reprises, une opération entre deux entrées du « tableau ». Vous devez chercher à chaque fois les valeurs nécessaires dans le tableau pour obtenir la bonne réponse comme suit :
 1. Obtenez 320 à l'aide d'une multiplication.
 2. Obtenez 10 à l'aide d'une division.
 3. Obtenez 755 à l'aide d'une addition.
- Créez une 3^{ème} variable.
- Assignez le « tableau » : [6, ["herbe", "route", "maison", "sont"], ["rue", [5, "vaches", "Les", "mouton"], "chèvre"], ["devant", "lui", "la"], "", ["!", "."]]; à cette variable.
- En utilisant uniquement le « tableau » ci-dessus, affichez dans une boîte de dialogue, à l'aide de concaténations, la phrase suivante : *Les 6 vaches sont devant la maison.*

Exercice 9/11 : Déclarer et utiliser des fonctions

- Préparation :
- Idem
- Présentation :
- On peut déclarer une fonction comme suit :


```
function() {
  "ceci est une fonction anonyme";
}
```
- Une fonction de ce type est appelée **fonction anonyme**. Pour l'instant, on ne peut pas l'exécuter puisqu'on ne dispose pas d'un nom permettant de l'appeler.
- On peut également créer une **fonction nommée** comme suit :


```
function tomCruise() {
  "ceci est une fonction nommée";
}
```
- Une fonction nommée peut être exécutée en utilisant son nom et les symboles (et). Par exemple :


```
tomCruise();
```
- On peut assigner une référence à une fonction nommée ou anonyme à une variable. Par exemple :


```
var goose = function() {
  "ceci est une fonction anonyme assignée à une
  variable";
}
```

Ce qui nous permet de pouvoir exécuter la fonction anonyme en écrivant :

```
goose();
```

Et par exemple :

```
var maverick = function tomCruise() {
    "ceci est une fonction nommée assignée à une
     variable";
}
```

Ce qui nous permet de pouvoir exécuter la fonction nommée en écrivant :

```
maverick();
```

Attention, si on assigne une fonction nommée à une variable, on ne peut pas exécuter la fonction en utilisant son nom mais uniquement en utilisant la variable à laquelle on l'a assigné.

- Le moteur du langage traite le code d'un programme en 2 étapes. D'abord le « parse time », temps pendant lequel le moteur du langage lit le code et le transforme en code exécutable. Puis le « run time », temps pendant lequel le moteur du langage exécute les instructions.
- Les fonctions sont stockées en mémoire lors du « parse time », les assignations sont effectuées lors du « run time ».
- Une fonction nommée qui n'est pas assignée à une variable peut donc être appelée n'importe quand dans le code en utilisant son nom.
- Une fonction nommée ou anonyme qui est assignée à une variable ne peut être appelée qu'après son assignation.
- L'exécution d'une fonction peut être effectuée dans les emplacements réservés aux scripts :
 1. Entre les chevrons d'une balise `<script>` ;
 2. En tant que valeur d'un attribut d'évènement en `HTML` comme `onmouseover`, `onclick`, `onmouseout`, ... Par exemple :
 - `onclick="maverick();"`, cet attribut d'évènement doit être écrit sur une balise `HTML`.

- Enoncé :

- Déclarer la fonction suivante comme suit entre les chevrons d'une balise `<script>` :

```
var uneFonctionAnonyme = function(){
    alert("La fonction alert est déclenchée par la
          fonction référencée dans la variable
          uneFonctionAnonyme");
}
```

- Exécuter la fonction `uneFonctionAnonyme` après l'avoir déclarée.

- Exécutez la fonction `uneFonctionAnonyme` AU moment où l'utilisateur clique sur le premier paragraphe. Il faut utiliser l'attribut d'événement `onclick`.
- Exécutez la fonction `uneFonctionAnonyme` APRÈS le chargement du document. Il faut utiliser l'attribut d'événement `onload`.
- Exécutez la fonction `uneFonctionAnonyme` APRÈS que le navigateur ait chargé et affiché le premier paragraphe et AVANT qu'il ait chargé et affiché le second paragraphe. Il faut utiliser une balise `script`.
- Créez un 2^{ème} fichier. Copiez-y la déclaration de la fonction `uneFonctionAnonyme`. Supprimez cette fonction de votre fichier HTML. Indiquez au navigateur Internet qu'il doit télécharger le fichier externe créé. Il faut utiliser une balise `script` avec l'attribut `src`. Toutes les demandes d'exécution de la fonction `uneFonctionAnonyme` doivent fonctionner.

Exercice 10/11 : Déclarer et utiliser des fonctions avec arguments

- Préparation :
 - Idem
- Présentation :
 - On peut déclarer des fonctions avec 1 ou plusieurs arguments qui correspondent à des paramètres attendus en entrée.
 - Les instructions qui sont contenues dans la fonction pourront alors utiliser ces arguments comme s'il s'agissait de variables déclarées. Par exemple une fonction déclarée avec 1 argument :


```
var uneFonctionAvecUnArgument = function(argument){
    var message = argument + "jour";
    alert(message);
}
```
 - Pour utiliser une fonction qui a 1 argument, il faut fournir 1 paramètre. Par exemple :


```
uneFonctionAvecUnArgument ("bon");
```

Crée et affiche le texte *bonjour*.

```
uneFonctionAvecUnArgument ("se");
```

Crée et affiche le texte *séjour*.
 - On peut déclarer des fonctions avec plus d'1 argument. Par exemple :


```
var uneFonctionAvecDeuxArguments = function(argUn, argDeux) {
    var message = argUn + argDeux;
    alert(message);
}
```
 - Pour utiliser une fonction qui a 2 arguments, il faut fournir 2 paramètres. Par exemple :


```
uneFonctionAvecDeuxArguments ("bon", "heur");
```

Crée et affiche le texte *bonheur*.

```
uneFonctionAvecDeuxArguments ("heu", "reux");
```

Crée et affiche le texte *heureux*.

- On peut déclarer une fonction avec autant d'arguments que nécessaire.
- Enoncé :
 - Créez une 1^{ère} fonction avec 1 argument. Cette fonction produit l'affichage de la valeur de l'argument dans une boite de dialogue.
 - Exécutez cette fonction en lui fournissant 1 chaîne de caractères comme paramètre.
 - Créez une 2^{ème} fonction avec 2 arguments. Cette fonction produit l'affichage de la somme des valeurs des deux arguments dans une boite de dialogue.
 - Exécutez cette fonction en lui fournissant 2 nombres comme paramètres.
 - Créez une 3^{ème} fonction avec 1 argument. Cette fonction considère que la valeur de l'argument sera un « tableau » de 2 indices. Elle concatène la valeur au 1^{er} indice avec la valeur au 2^{ème} indice et produit l'affichage de cette concaténation dans une boite de dialogue.
 - Exécutez cette fonction en lui fournissant 1 tableau contenant 2 chaîne de caractères comme paramètre.

Exercice 11/11 : Déclarer et utiliser des fonctions avec valeurs de retour

- Préparation :
 - Idem
- Présentation :
 - On peut préciser, à l'intérieur d'une fonction, qu'on souhaite retourner une valeur à l'aide du mot-clé `return`
 - La valeur sera retournée exactement là où la fonction est exécutée. Par exemple, pour la fonction suivante :

```
var fonctionAvecRetour = function() {
    var calcul = 3 + 5;
    return calcul;
}
```

La valeur retournée est la valeur de la variable `calcul`. Cette valeur sera retournée là où la fonction sera exécutée. Par exemple :

```
// ici la valeur de retour n'existe pas encore
fonctionAvecRetour();
// ici la valeur de retour n'existe plus
```

On a donc 2 solutions pour utiliser une valeur retournée :

1. On exécute la fonction et on stocke la valeur retournée dans une variable sur la même ligne. Par exemple :

```
var retour = fonctionAvecRetour();
var calcul = 2 + retour;
```

2. Ou alors, on utilise directement le résultat de l'exécution de la fonction dans une instruction. Par exemple :

```
var calcul = 2 + fonctionAvecRetour();
```

- Les instructions après l'exécution du mot-clé `return` ne sont jamais exécutées.
- Une fonction pour laquelle on n'a pas spécifié de valeur de retour retourne la valeur `undefined`
- Enoncé :
 - Créez 1 fonction qui :
 1. Concatène un nombre à une chaîne de caractère.
 2. Retourne la valeur obtenue.
 - Exécutez cette fonction et récupérez la valeur de retour dans une variable.
 - Affichez la valeur de retour dans une boite de dialogue.
 - Reprenez la dernière fonction de l'exercice précédent.
 - Supprimez l'instruction au sein de la fonction qui est responsable de l'affichage de la concaténation dans une boite de dialogue.
 - A la place, la fonction retourne la valeur de la concaténation.
 - Exécutez cette fonction et récupérez la valeur de retour dans une variable.
 - Affichez la valeur de retour dans une boite de dialogue.

b. Bases niveau 2

Objectif : Savoir écrire de conditions et de boucles

Exercice 1/9 : Ecrire une condition simple

- Préparation :
 - Créez un document `HTML` valide contenant un paragraphe de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - Les blocs conditionnels sont exprimés à l'aide des mots-clés `if` et optionnellement `else`. Le mot-clé `if` reçoit un booléen qui prend pour valeur soit `true` soit `false`. Si le booléen reçu est évaluée à `true` par le moteur du langage, alors l'instruction correspondant au `if` est exécutée. Sinon elle ne l'est pas. Par exemple :

```

if(true) {
    // cette instruction est toujours exécutée.
}

if(false) {
    // cette instruction n'est jamais exécutée.
}

```

Optionnellement, on peut prévoir une instruction à exécuter si la valeur reçue par le `if` est évaluée à `false` en utilisant le mot-clé `else`. Par exemple :

```
if(false){
    // cette instruction n'est jamais exécutée.
} else{
    // cette instruction est toujours exécutée.

}
```

- Pour obtenir un booléen, on peut utiliser les opérateurs de comparaison (`<=`, `>`, `==`, `=====`, etc.).
- Pour combiner des booléen et en obtenir un seul, on peut utiliser les opérateurs logiques. Les opérateurs logiques disponibles en ES3/5 sont :
 1. `&&` le ET logique
 2. `||` le OU logique
 3. `!` le NON logique
- Enoncé :
 - 1^{ère} Partie :
 1. Créez une 1^{ère} variable.
 2. Assignez une valeur numérique de votre choix à cette variable.
 3. Créer une 2^{ème} variable.
 4. Utilisez l'opérateur de comparaison `>=` pour obtenir un booléen en comparant la valeur de la 1^{ère} variable avec le nombre 10.
 5. Assignez le résultat de ce calcul (le booléen obtenu) à la 2^{ème} variable.
 6. Utilisez la 2^{ème} variable avec le mot-clé `if` et le mot-clé `else` pour écrire un bloc conditionnel qui affiche le texte *Entrée* dans une boîte de dialogue si `if` reçoit `true` et qui affiche le texte *Sortie* sinon.
 7. Modifiez la valeur de la 1^{ère} variable pour tester les 2 cas possibles.
 - 2^{ème} partie :
 1. Reprenez le code précédent à l'exception de la 1^{ère} variable.
 2. Mettez ce code dans une fonction avec 1 argument. Maintenant, c'est l'argument de la fonction qui doit être utilisé avec l'opérateur de comparaison et la valeur 10.
 3. Exécutez plusieurs fois la fonction en fournissant des paramètres de telle sorte que vous puissiez tester tous les cas possibles.

Exercice 2/9 : Utiliser une fonction qui retourne un booléen pour écrire une condition

- Préparation :
 - Créez un document `HTML` valide contenant 2 paragraphes de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :

- `confirm` est une « fonction » fournie de base par le moteur du langage.
- Lorsque la « fonction » `confirm` est exécutée, le navigateur Internet affiche une boîte de dialogue dans laquelle sont affichés les boutons OK et Annuler.
- Contrairement à la fonction `alert`, cette « fonction » retourne le choix de l'utilisateur sous la forme d'un booléen.
- Enoncé :
 - 1^{ère} Partie :
 1. Lorsque l'utilisateur clique sur le 1^{er} paragraphe ; une fonction, que vous avez déclaré dans l'en-tête du document, s'exécute.
 2. Cette fonction doit, elle-même, déclencher la fonction qui affiche une boîte de dialogue contenant le texte « Ceci est un choix » et le boutons OK et Annuler.
 3. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, le mot-clé `function`, la fonction `confirm`.
 - 2^{ème} Partie :
 1. Lorsque l'utilisateur clique sur le 2^{ème} paragraphe ; une autre fonction que vous avez, également, déclaré dans l'en-tête du document, s'exécute.
 2. Cette fonction doit, elle-même, déclencher une fonction qui affichera une boîte de dialogue contenant le texte « Ceci est un choix » ainsi que les boutons OK et Annuler.
 3. Assignez la valeur renvoyée par la fonction affichant la boîte de dialogue à une variable.
 4. Affichez la valeur de cette variable dans une boîte de dialogue.
 5. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, le mot-clé `function`, la fonction `confirm` et la fonction `alert`.
 - Quelle information nous retourne la fonction `confirm` si l'utilisateur clique sur le choix OK ? Sur Annuler ?
 - 3^{ème} Partie : En vous inspirant des parties précédentes, écrivez le programme (algorithme) suivant :
 1. Au chargement du document, j'affiche une boîte de dialogue contenant le message « Voulez-vous continuer ? ». Si l'utilisateur clique sur OK, j'affiche un message « Vous avez cliqué sur OK » ; sinon rien ne s'affiche.
 2. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clé `function` et `if` la fonction `confirm` et la fonction `alert`.

Exercice 3/9 : Utiliser une fonction qui retourne un booléen pour écrire une condition et utiliser différents attributs d'évènement.

- Préparation :
 - Créez un document `HTML` valide contenant 1 titre et 2 paragraphes de texte ;
 - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML` ;
 - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
 - Idem.
- Enoncé :
 - 1^{ère} Partie : En vous inspirant de l'exercice précédent, écrivez le programme (algorithme) suivant :
 1. Au chargement du document, j'affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur OK, j'affiche une boite de dialogue avec le message « Vous avez cliqué sur OK ! » SINON j'affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
 2. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clé `function`, `if` et `else`, la fonction `confirm` et la fonction `alert`.
 - 2^{ème} Partie : En vous inspirant de la partie précédente, écrivez le programme (algorithme) suivant :
 1. Quand l'utilisateur clique sur le premier paragraphe, j'affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur OK, j'affiche une boite de dialogue avec le message « Vous avez cliqué sur OK ! » SINON j'affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
 2. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, les mots-clé `function`, `if` et `else`, la fonction `confirm` et la fonction `alert`.
 - 3^{ème} Partie : En vous inspirant de la partie précédente, écrivez le programme (algorithme) suivant :
 1. Quand l'utilisateur survole le premier titre, j'affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur OK, j'affiche une boite de dialogue avec le message « Vous avez cliqué sur OK ! » SINON j'affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
 2. Vous devez utiliser une balise `<script>`, l'attribut d'événement `onmouseover`, les mots-clé `function`, `if` et `else`, la fonction `confirm` et la fonction `alert`.