ES3/5

Bases et techniques avancées

Livret d'exercices didactiques

Précède : ES2018 et « Design Patterns »

Table des matières

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 1ère variable à laquelle vous assignerez une chaine de caractère. Affichez la valeur vers laquelle pointe cette variable dans une boite de dialogue.
  - Créez une 2ème variable à laquelle vous assignerez une chaine 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 boite de dialogue.

EXERCICE 2/11 : UTILISER DES OPERATEURS ARITHMETIQUES

- 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 :
    - `+` pour l'addition, attention à la confusion avec la concaténation.
    - `-` pour la soustraction
    - `/` pour la division, attention aux divisions par 0.
    - `%` pour le reste entier de la division, attention prend toujours le signe du dividende (nombre divisé).
    - `++` pour l'incrémentation.
    - `--` pour la décrémentation.
    - 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 4ème variable. Affichez la valeur vers laquelle pointe la 4ème variable dans une boite de dialogue.
  - Multipliez la valeur vers laquelle pointe la 4ème variable par le nombre 5. Assignez le résultat de cette opération à une 5ème variable. Affichez la valeur vers laquelle pointe la 5ème variable dans une boite de dialogue.
  - Divisez la valeur vers laquelle pointe cette 5ème variable par 3. Assignez le résultat de cette opération à une 6ème variable. Affichez la valeur vers laquelle pointe la 6ème variable dans une boite de dialogue.

EXERCICE 3/11 : UTILISER DES OPERATEURS ARITHMETIQUES, 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 :
    - `7 % 2;` donne 1
    - `29 % 25;` donne 4
    - `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 1ère variable la valeur 90 ;
  - Assignez à la 2ème variable la valeur 11 ;
  - Assignez à la 3ème variable la valeur 20 ;
  - Assignez à la 4ème variable la valeur 9 ;

- Le diviseur est toujours la 1ère variable (le diviseur est la variable à droite de l'opérateur).
- Calculez le reste de la division de la 2ème variable par la 1ère et affichez le résultat dans une boite de dialogue.
- Calculez le reste de la division de la 3ème variable par la 1ère et affichez le résultat dans une boite de dialogue.
- Calculez le reste de la division de la 4ème variable par la 1ère et affichez le résultat dans une boite de dialogue.
- Que remarquez-vous ?

EXERCICE 4/11 : UTILISER L'OPERATEUR DE CONCATENATION

- 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 1ère variable le texte : *La nuit*
  - Assignez à la 2ème variable le texte : *tous les*
  - Assignez à la 3ème variable le texte : *chats*
  - Assignez à la 4ème variable le texte : *sont gris*
  - Concaténez les 4 variables et assignez le résultat à une 5ème variable.
  - Affichez la valeur vers laquelle pointe la 5ème variable dans une boite de dialogue.

EXERCICE 5/11 : UTILISER L'OPERATEUR DE CONCATENATION – SUITE

- Préparation :
  - Idem
- Présentation :
  - Idem
- Enoncé :
  - Créez 3 variables.
  - La 1ère pointe vers une chaîne de caractères.
  - La 2ème pointe vers un nombre.
  - La 3ème pointe vers un nombre.
  - Concaténez la 1ère variable avec la 2ème et assignez le résultat à une 4ème variable.
  - Concaténez la 4ème variable avec la 3ème et assignez le résultat à une 5ème variable.
  - Affichez la valeur vers laquelle pointe la 5ème 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 :

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

```
var monTableau;
monTableau = [15,"une information",24];
monTableau[3] = "un autre info";
```

- Enoncé :
  - Créez une 1ère 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 boite de dialogue.

- Concaténez la 1ère et la dernière valeur du « tableau » sans oubliez de les séparer par un espace.
- Ajoutez la valeur obtenue au « tableau ».
- Affichez dans une boite 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 chaines 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]` ou 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 1ère 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 boite de dialogue le 2ème monument de Berlin.
  - En utilisant ce « tableau », affichez dans une boite de dialogue le 5ème monument de Paris.
  - En utilisant ce « tableau », affichez dans une boite de dialogue le 3ème monument de Rome.
  - Créez une 2ème 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 :
    - Obtenez 320 à l'aide d'une multiplication.
    - Obtenez 10 à l'aide d'une division.
    - Obtenez 755 à l'aide d'une addition.
  - Créez une 3ème 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 boite de dialogue, à l'aide de concaténations, la phrase suivante : *Les 6 vaches sont devant la maison.*

EXERCICE 9/11 : DECLARER 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 :
  - Entre les chevrons d'une balise `<script>` ;
  - En tant que valeur d'un attribut d'évènement en `HTML` comme `onmouseover`, `onclick`, `onmouseover`, … 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
      referencée dans la variable uneFonctionAnonyme");
}
```

- Exécutez 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 2ème 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 : DECLARER 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 1ère 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 2ème 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 3ème fonction avec 1 argument. Cette fonction considère que la valeur de l'argument sera un « tableau » de 2 indices. Elle concatène la valeur au 1er indice

avec la valeur au 2ème 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 : DECLARER 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 ou 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 :
    - Concatène un nombre à une chaîne de caractère.
    - 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 des conditions et des boucles

EXERCICE 1/8 : 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 :
    - `&&` le ET logique
    - `||` le OU logique
    - `!` le NON logique
- Enoncé :
  - 1ère Partie :
    - Créez une 1ère variable.
    - Assignez une valeur numérique de votre choix à cette variable.
    - Créer une 2ème variable.
    - Utilisez l'opérateur de comparaison `>=` pour obtenir un booléen en comparant la valeur de la 1ère variable avec le nombre 10.
    - Assignez le résultat de ce calcul (le booléen obtenu) à la 2ème variable.
    - Utilisez la 2ème variable avec le mot-clé `if` et le mot-clé `else` pour écrire un bloc conditionnel qui affiche le texte *Entrée* dans une boite de dialogue si `if` reçoit `true` et qui affiche le texte *Sortie* sinon.
    - Modifiez la valeur de la 1ère variable pour tester les 2 cas possibles.
  - 2ème partie :
    - Reprenez le code précédent à l'exception de la 1ère variable.
    - 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.
    - Exécutez plusieurs fois la fonction en fournissant des paramètres de telle sorte que vous puissiez tester tous les cas possibles.

EXERCICE 2/8 : UTILISER UNE FONCTION QUI RETOURNE UN BOOLEEN POUR ECRIRE 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 boite 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 :
  - 1ère Partie :
    - Lorsque l'utilisateur CLIQUE sur le 1er paragraphe ; une fonction, que vous avez déclaré dans l'en-tête du document, s'exécute.
    - Cette fonction doit, elle-même, déclencher la fonction qui affiche une boite de dialogue contenant le texte « Ceci est un choix » et le boutons « OK » et « Annuler ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, le mot-clé `function`, la fonction `confirm`.
  - 2ème Partie :
    - Lorsque l'utilisateur CLIQUE sur le 2ème paragraphe ; une autre fonction que vous avez, également, déclaré dans l'en-tête du document, s'exécute.
    - Cette fonction doit, elle-même, déclencher une fonction qui affichera une boite de dialogue contenant le texte « Ceci est un choix » ainsi que les boutons « OK » et « Annuler ».
    - Assignez la valeur retournée par la fonction affichant la boite de dialogue à une variable.
    - Affichez la valeur de cette variable dans une boite de dialogue.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, le mot-clé `function`, les fonctions `confirm` et `alert`.
  - Quelle information nous retourne la fonction `confirm` si l'utilisateur clique sur le choix OK ? Sur Annuler ?
  - 3ème Partie : En vous inspirant des parties précédentes, écrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur « OK », le navigateur Internet affiche un message « Vous avez cliqué sur OK » ; SINON rien ne s'affiche.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function` et `if`, les fonctions `confirm` et `alert`.

EXERCICE 3/8 : UTILISER UNE FONCTION QUI RETOURNE UN BOOLEEN POUR ECRIRE UNE CONDITION ET UTILISER DIFFERENTS ATTRIBUTS D'EVENEMENT

- Préparation :
  - Créez un document `HTML` valide contenant 1 titre principal 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 :
  - 1ère Partie : En vous inspirant de l'exercice précèdent, écrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur « OK », le navigateur Internet affiche une boite de dialogue avec le message « Vous avez cliqué sur OK ! » SINON le navigateur Internet affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `if` et `else`, les fonctions `confirm` et `alert`.
  - 2ème Partie : En vous inspirant de la partie précédente, écrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le premier paragraphe, le navigateur Internet 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 le navigateur Internet affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, les mots-clés `function`, `if` et `else`, les fonctions `confirm` et `alert`.
  - 3ème Partie : En vous inspirant de la partie précédente, écrivez le programme (algorithme) suivant :
    - Quand l'utilisateur SURVOLE le premier titre, le navigateur Internet affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». SI l'utilisateur clique sur OK, le navigateur Internet affiche une boite de dialogue avec le message « Vous avez cliqué sur OK ! » SINON le navigateur Internet affiche une boite de dialogue avec le message « Vous avez choisi d'annuler ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onmouseover`, les mots-clés `function`, `if` et `else`, les fonctions `confirm` et `alert`.

EXERCICE 4/8 : ECRIRE UNE BOUCLE SIMPLE

- Préparation :
  - Créez un document `HTML` valide contenant 1 titre principal 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 :
  - Les boucles sont des constructions syntaxiques du langage qui permettent de répéter plusieurs fois les mêmes instructions.
  - Les boucles sont exprimées à l'aide du mot-clé `while`. Le mot-clé `while` reçoit un booléen qui prend pour valeur soit `true` soit `false`. Si le booléen reçu est `true`, alors l'instruction contenue entre les accolades du `while` est exécutée. Sinon elle ne l'est pas. Par exemple :

```
while(true){
```

```
        // Cette instruction est toujours exécutée.
}

while(false){
        // Cette instruction n'est jamais exécutée.
}
```

- Pour utiliser les boucles, on peut lui fournir une variable qui pointe vers un booléen. Tant que ce booléen est `true`, le moteur exécute les instructions contenues dans la boucle. Pour arrêter la boucle, il faut modifier la valeur du booléen à l'INTERIEUR de la boucle. Par exemple :

```
var variable = true;
while(variable){
        // Cette instruction est exécutée.
        variable = false;
        // La valeur de la variable utilisée par la boucle a changé. La prochaine exécution de la boucle n'aura pas lieu.
}
```

- On appelle ce booléen « condition de sortie ».

- Enoncé :
  - En vous inspirant de l'exercice précèdent, écrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ». TANT QUE l'utilisateur clique sur « OK », le navigateur Internet affiche une boite de dialogue contenant le message « Voulez-vous continuer ? ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function` et `while`, les fonctions `confirm` et `alert`.

EXERCICE 5/8 : ECRIRE UNE CONDITION COMPLEXE

- Préparation :
  - Idem.
- Présentation :
  - `prompt` est une « fonction » fournie de base par le moteur du langage.
  - Lorsque la « fonction » `prompt` est exécutée, le navigateur Internet affiche une boite de dialogue dans laquelle sont affichés un champ de saisie, les boutons « OK » et « Annuler ».
  - Sur le même principe que la fonction `confirm`, cette « fonction » retourne la saisie de l'utilisateur.
- Enoncé :
  - 1ère Partie : Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le premier paragraphe, le navigateur Internet affiche une boite de dialogue contenant le message « Veuillez saisir un message ». PUIS le navigateur Internet affiche la saisie de l'utilisateur dans une boite de dialogue.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, le mot-clé `function`, les fonctions `prompt` et `alert`.
  - 2ème Partie :
    - Quelle information retourne la fonction `prompt` si l'utilisateur clique sur « Annuler » ?
    - Quelle information retourne la fonction `prompt` si l'utilisateur ne saisit rien et clique sur « OK » ?

- Quelle information retourne la fonction `prompt` si l'utilisateur saisit un texte et clique sur « OK » ?

- 3ème Partie : Ecrivez le programme (algorithme) suivant :
  - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'attention de l'utilisateur.
    SI l'utilisateur clique sur « Annuler », le navigateur Internet affiche le message « Vous avez annulé la saisie ».
    SI l'utilisateur clique sur « OK » sans rien saisir, le navigateur Internet affiche le message « Vous n'avez rien saisi ».
    SINON (si l'utilisateur n'a pas cliqué sur « Annuler » ET n'a pas rien saisi), le navigateur Internet affiche la saisie de l'utilisateur dans une boite de dialogue.
  - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `if` et `else`, les fonctions `prompt` et `alert`.

EXERCICE 6/8 : ECRIRE UNE BOUCLE AVEC UNE CONDITION DE SORTIE COMPLEXE

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - En vous inspirant de l'exercice précédent, écrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le premier paragraphe, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'attention de l'utilisateur.
      TANT QUE l'utilisateur clique sur « Annuler » ou clique sur « OK » sans rien saisir, le navigateur Internet affiche une boite de dialogue avec un champ de saisie.
      PUIS, le navigateur Internet affiche la saisie de l'utilisateur dans une boite de dialogue.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, les mots-clés `function`, `while`, les fonctions `prompt` et `alert`.

EXERCICE 7/8 : ECRIRE UNE BOUCLE AVEC COMPTEUR

- Préparation :
  - Idem.
- Présentation :
  - On parle de boucle avec compteur quand la condition de sortie de la boucle est basée sur une variable dont le contenu est incrémenté ou décrémenté un certain nombre de fois.
  - Quand cette variable atteint une limite haut ou basse, la condition de sortie est validée et la boucle s'arrête. Ce type de construction permet de fixer précisément le nombre de fois que le code contenu dans la boucle doit être exécuté.
  - On peut écrire une boucle avec compteur comme suit (forme développée) :

```
var compteur = 0;
// On calcule la valeur de la condition de sortie.
var conditionDeSortie = compteur < 5;
while(conditionDeSortie){
      // Code à exécuter.
      // On incrémente le compteur dans la boucle.
```

```
        compteur = compteur + 1;
        // On recalcule la valeur de la condition de sortie.
        conditionDeSortie = compteur < 5 ;
}
```

- On peut également réduire le code nécessaire à la boucle en se passant de variables intermédiaires comme suit (forme raccourcie) :

```
var compteur = 0;
while(compteur < 5) {
        // Code à exécuter.
        compteur++; // Equivaut à compteur = compteur + 1;
}
```

- Enoncé :
  - 1ère Partie : Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, initialisez un compteur à 0. TANT QUE la valeur du compteur n'a pas atteint la valeur -5, le navigateur Internet affiche dans une boite de dialogue la valeur courante du compteur. Le navigateur Internet affichera donc successivement les valeurs que prend le compteur.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `while`, la fonction `alert`.
  - 2ème Partie : Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, initialisez un « tableau » contenant 7 valeurs de votre choix. Créez boucle avec compteur qui affichera successivement chaque valeur contenue dans le « tableau » dans une boite de dialogue.
    - Vous devez utiliser la valeur d'un compteur comme indice du « tableau » dont vous souhaitez afficher les valeurs.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `while`, la fonction `alert`.

EXERCICE 8/8 : ECRIRE UNE BOUCLE AVEC COMPTEUR – SUITE

- Préparation :
  - Idem.
- Présentation :
  - Une boucle avec compteur peut être écrite de façon encore plus concise à l'aide du mot-clé `for`.
  - Par exemple la boucle `while` suivante :

```
var compteur = 0;
while(compteur < 5) {
        // Code à exécuter.
        compteur++; // Equivaut à compteur = compteur + 1;
}
```

Peut être écrite comme suit :

```
for(var compteur = 0; compteur < 5; compteur++) {
        // Code à exécuter.
}
```

Les trois indications nécessaires pour la boucle (valeur de départ, valeur maximale et incrément) sont regroupées au même endroit.

- Enoncé :
  - 1ère Partie : Reprenez le code la 1ère partie de l'exercice précèdent mais écrivez-le avec un `for`.
  - 2ème Partie : Reprenez le code la 2ème partie de l'exercice précèdent mais écrivez-le avec un `for`.

c. Intermédiaire

Objectif : Savoir écrire conditions, boucles et fonctions complexes

EXERCICE 1/11 : ECRIRE UN ALGORITHME SIMPLE AVEC DES BOUCLES

- Préparation :
  - Créez un document `HTML` valide contenant 1 titre principal 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 :
  - Néant.
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le premier paragraphe, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'utilisateur.
      TANT QUE l'utilisateur clique sur « Annuler » OU clique sur « OK » sans rien saisir, le navigateur Internet affiche une boite de dialogue avec un champ de saisie.
      PUIS, le navigateur Internet affiche encore une boite de dialogue avec un champ de saisie à l'utilisateur.
      TANT QUE l'utilisateur clique sur « Annuler » OU clique sur « OK » sans rien saisir, le navigateur Internet affiche une boite dialogue avec un champ de saisie.
      L'utilisateur a donc saisi 2 informations. Vous devez concaténer avec l'opérateur `+` ces 2 informations.
      PUIS le navigateur Internet doit afficher le résultat de la concaténation dans une boite de dialogue.
    - Quel résultat s'affiche si l'utilisateur saisit 2 chiffres ? A votre avis pourquoi ?
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, les mots-clés `function`, `while`, les fonctions `alert` et `prompt`.

EXERCICE 2/11 : ECRIRE UN ALGORITHME SIMPLE AVEC UNE CONDITION

- Préparation :
  - Idem
- Présentation :
  - `parseFloat` est une « fonction » fournie de base par le moteur du langage.
  - La « fonction » `parseFloat` attend un paramètre en entrée :
    - Si ce paramètre représente un nombre, elle retourne une valeur de type Nombre.
    - Si ce paramètre représente autre chose qu'un nombre, elle retourne le nombre NaN.
  - Cette fonction est fréquemment utilisée pour transformer des chaines de caractère représentant des nombres (par exemple `"2"` en 2, `"5px"` en 5, …).
  - `isNaN` est une « fonction » fournie de base par le moteur du langage.
  - La « fonction » `isNaN` attend un paramètre en entrée :
    - Si ce paramètre représente un nombre, elle retourne `false`.
    - Si ce paramètre ne représente pas un nombre, elle retourne `true`.

- Cette « fonction » est fréquemment utilisée pour vérifier si une variable contient un nombre ou pas.

- Enoncé :
  - 1ère Partie : Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'utilisateur.
      PUIS, utilisez la fonction parseFloat pour transformer la saisie de l'utilisateur.
      Affichez le résultat de la transformation dans une boite de dialogue.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, le mot-clé `function`, les fonctions `alert`, `prompt` et `parseFloat`.
  - 2ème Partie : Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'utilisateur.
      PUIS, utilisez la fonction `parseFloat` pour transformer la saisie de l'utilisateur.
      PUIS, utilisez la fonction `isNaN` avec le la saisie transformée pour obtenir un booléen indiquant s'il s'agit d'un nombre ou pas.
      Affichez ce booléen dans une boite de dialogue.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, le mot-clé `function`, les fonctions `alert`, `prompt`, `parseFloat` et `isNaN`.
  - 3ème Partie : Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le premier paragraphe, le navigateur Internet affiche une boite de dialogue pour demander à l'utilisateur de saisir un nombre.
      SI l'utilisateur n'a pas saisi un nombre, le navigateur Internet affiche le message « Vous devez saisir un nombre » dans une boite de dialogue.
      SINON, le navigateur Internet affiche le message « Merci d'avoir saisi un nombre ».
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onclick`, les mots-clés `function`, `if`, `else` et les fonctions `alert`, `prompt`, `parseFloat` et `isNaN`.

EXERCICE 3/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC UNE BOUCLE

- 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 :
  - Néant
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet demande à l'utilisateur de saisir un nombre.
      TANT QUE il ne l'a pas saisi, le navigateur Internet lui demande de saisir un nombre.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `while` et les fonctions `alert`, `prompt`, `parseFloat` et `isNaN`.

EXERCICE 4/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC DEUX BOUCLES

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Au CHARGEMENT du document, le navigateur Internet demande à l'utilisateur de saisir un 1er nombre.
      TANT QUE il ne l'a pas saisi, le navigateur Internet lui demande de saisir un 1er nombre.
      PUIS le navigateur Internet demande à l'utilisateur de saisir un 2ème nombre.
      TANT QUE il ne l'a pas saisi, le navigateur Internet lui demande de saisir un 2ème nombre.
      ENFIN, le navigateur Internet affiche dans une boite de dialogue le résultat de la somme du 1er et du 2ème nombre.
    - Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `while` et les fonctions `alert`, `prompt`, `parseFloat` et `isNaN`.

EXERCICE 5/11 : CREER UNE FONCTION

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - Dans l'exercice précédent, deux blocs de code se ressemblent beaucoup. Ce sont les parties du programme qui permettent d'obliger l'utilisateur à saisir un nombre.
  - On pourrait donc créer une fonction qui évite d'avoir à écrire à deux reprises des instructions quasi identiques.
  - Écrivez cette fonction.
  - Veillez aux trois points suivants pour l'écriture de cette fonction :
    - La fonction ne peut pas ne se terminer tant que l'utilisateur n'a pas saisi de nombre.
    - La fonction prend en argument le message qui s'affichera à l'utilisateur lors de la saisie.
    - La fonction retourne le nombre saisi par l'utilisateur.

EXERCICE 6/11 : ECRIRE UN ALGORITHME COMPLEXE A L'AIDE DE PLUSIEURS BOUCLES ET D'1 TABLEAU

- Préparation :
  - Idem
- Présentation :
  - Nous avons vu dans un des exercices précédents qu'il était possible d'ajouter une valeur à un tableau.
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :

- Au CHARGEMENT du document, obligez 4 fois l'utilisateur à saisir du texte. Assignez chaque saisie à un indice d'un « tableau ». Enfin, affichez les différentes valeurs saisies à l'aide de boites de dialogue.
- Vous devez utiliser une balise `<script>`, l'attribut d'événement `onload`, les mots-clés `function`, `for` `while` et les fonctions `alert` et `prompt`.

EXERCICE 7/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC BOUCLE ET CONDITIONS – 1ERE PARTIE

- Préparation :
  - Idem
- Présentation :
  - La fonction `open` permet d'ouvrir une nouvelle fenêtre de navigation avec l'`URL` indiquée.
  - La fonction `open` a besoin de 3 paramètres :
    - L'`URL` de la fenêtre à ouvrir, sous la forme d'une chaine de caractères ;
    - Un identifiant unique SANS ESPACES pour la fenêtre, sous la forme d'un nombre ou d'une chaine de caractères ;
    - Une chaine de caractère représentant des options pour la nouvelle fenêtre.
  - La chaine de caractère des options peut comporter une ou plusieurs des « options » suivantes séparée par des virgules sans espaces :
    - `directories=yes` ou `directories=no`, pour afficher ou non les boutons de navigation ;
    - `location=yes` ou `location=no`, pour afficher ou non la barre d'adresse ;
    - `menubar=yes` ou `menubar=no`, pour afficher ou non la barre de menu ;
    - `resizable=yes` ou `resizable=no`, pour définir si la taille de la fenêtre est modifiable ou non ;
    - `scrollbars=yes` ou `scrollbars=no`, affiche ou non les barres de défilement ;
    - `status=yes` ou `status=no`, affiche ou non la barre d'état ;
    - `toolbar=yes` ou `toolbar=no`, affiche ou non la barre d'outils ;
    - `width=nombre`, (en pixels) définit la largeur de la fenêtre ;
    - `height=nombre`, (en pixels) définit la hauteur de la fenêtre.
  - Par exemple :

```
open("http://www.google.com","Google","directories=no,menubar=no");
```

- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le 1er titre, le navigateur Internet ouvre une fenêtre vers Google.
    - Quand l'utilisateur CLIQUE sur le 1er paragraphe, le navigateur Internet ouvre une fenêtre vers Bing.
    - Quand l'utilisateur SURVOLE le 2ème paragraphe, la navigateur Internet ouvre une fenêtre vers DuckDuckGo.

EXERCICE 8/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC BOUCLE ET CONDITIONS – 2EME PARTIE

- Préparation :

- Idem
- Présentation :
  - Idem.
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :

    Quand l'utilisateur CLIQUE sur le 1er paragraphe, le navigateur Internet affiche une boite de dialogue avec un champ de saisie à l'utilisateur.
    L'utilisateur doit impérativement saisir une chaine de caractère.
    Cette chaine de caractère doit ensuite être utilisée pour ouvrir une nouvelle fenêtre à l'aide de la fonction `open`.

EXERCICE 9/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC BOUCLE ET CONDITIONS – 3EME PARTIE

- Préparation :
  - Idem
- Présentation :
  - Idem
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le 1er paragraphe, le navigateur Internet affiche une boite de dialogue pour lui demander s'il souhaite ouvrir une fenêtre vers Google.
      SI il clique sur « OK », le navigateur Internet ouvre une nouvelle fenêtre vers Google.
      SINON le navigateur Internet affiche une boite de dialogue avec un champ de saisie contenant le texte : « *Veuillez saisir l'URL d'un site alternatif* ».
      - SI l'utilisateur saisit un texte, ce texte est utilisé comme URL pour ouvrir une nouvelle fenêtre.
      - SINON un message s'affiche dans une boite de dialogue avec le texte : « *Tant pis !* ».

EXERCICE 10/11 : ECRIRE UN ALGORITHME COMPLEXE AVEC BOUCLE ET CONDITIONS – 4EME PARTIE

- Préparation :
  - Idem
- Présentation :
  - Idem.
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Quand l'utilisateur CLIQUE sur le 1er paragraphe, le navigateur Internet affiche une boite de dialogue pour lui demander s'il souhaite ouvrir une fenêtre vers Google.
      SI il clique sur « OK », le navigateur Internet ouvre une nouvelle fenêtre vers Google.
      SINON le navigateur Internet affiche une boite de dialogue avec un champ de saisie contenant le texte : « *Veuillez saisir l'URL d'un site alternatif* ».
      - SI l'utilisateur saisit un texte, ce texte est utilisé comme URL pour ouvrir une nouvelle fenêtre.

- SINON un message s'affiche dans une boite de dialogue avec le texte : « *Tant pis !* » et l'utilisateur et « renvoyé » à l'étape 1.

En bref, il s'agit de créer une boucle qui OBLIGE l'utilisateur à ouvrir une nouvelle fenêtre vers google ou un site alternatif de son choix.

EXERCICE 11/11 : ECRIRE UNE BOUCLE A « POP-UPS »

- Préparation :
  - Idem
- Présentation :
  - Idem
- Enoncé :
  - Ecrivez le programme (algorithme) suivant :
    - Créez un « tableau » contenant 5 URLs.
    - TANT QUE l'ensemble du « tableau » n'a pas été parcouru, utilisez l'URL contenue à l'indice courant du « tableau » pour ouvrir une nouvelle fenêtre vers cet URL.
    - Utilisez une boucle `while` avec un compteur de tours.

d. Perfectionnement

Objectif : Se perfectionner dans l'écriture de conditions, boucles et fonctions complexes

EXERCICE 1/11 : OPTIMISER UN ALGORITHME EN UTILISANT LE MECANISME DES « CLOSURES »

- Préparation :
  - Idem
- Présentation :
  - Une fonction peut retourner une référence à une « sous » fonction. Par exemple :

```
var leNombreEst = function(nombre){
            var suffixe = " millions !";
            return function(prefixe){
                  alert(prefixe + nombre + suffixe);
            }
};
```

  - Lorsqu'on exécute la « sous » fonction retournée, celle-ci à accès aux arguments et variables déclarés dans le contexte de la fonction où elle a été originellement déclarée. Par exemple :

```
var ecrirePhrase = leNombreEst(5);
// Assigne une référence à la "sous " fonction à la variable ecrirePhrase.
ecrirePhrase("Je serais heureux d'avoir ");
// Affiche une boite de dialogue contenant le texte " Je serais heureux d'avoir 5 millions !"
```

  - On pourrait dire que la « sous » fonction se « souvient » du contexte dans lequel elle a été défini et de tous les arguments et variables définis dans ce contexte.
  - Ce mécanisme s'appelle une fermeture (« closure » en anglais).
- Enoncé :
  - En vous inspirant de ce mécanisme, vous devez créer une fonction qui déclare un « tableau » et qui retourne ensuite une « sous » fonction qui utilise ce « tableau ».
  - Voici le « tableau » :

```
var mots = [8,"je me lève", 13, "je déjeune", 23, "je vais dormir"];
```

- 1ère Partie : Déclarez une fonction qui crée ce tableau et qui retourne une « sous » fonction.
- 2ème Partie : Ecrivez une boucle avec compteur dans la « sous » fonction pour parcourir le « tableau » et affichez la phrase : "Il est 8h, je me lève. Il est 13h, je déjeune. Il est 23h, je vais dormir."
- 3ème Partie : Exécutez la fonction pour obtenir une référence à la « sous » fonction et assignez cette dernière à une variable. Exécutez la « sous » fonction pour valider son fonctionnement.

EXERCICE 2/11 : ECRIRE UNE FONCTION DONT LE NOMBRE D'ARGUMENTS EST VARIABLE

- Préparation :
  - Idem
- Présentation :
  - Dans une fonction, la variable spéciale `arguments` est un « tableau » contenant l'ensemble des paramètres passés en argument de la fonction.
  - On peut donc imaginer de passer un nombre variable de paramètres à une fonction et utiliser la variable spéciale `arguments` pour les utiliser au sein de la fonction. Par exemple :

```
function exemple(){
      alert(arguments[1]);
}
exemple(1, "bidule", 3) ;
// Affiche bidule.
```

- Enoncé :
  - 1ère Partie : Créez une fonction acceptant un nombre variable de paramètres en entrée. Cette fonction calcule la moyenne des valeurs passées en entrée. Pour ce faire vous allez devoir écrire une boucle qui va additionner une à une chaque valeur du tableau et diviser le total par 2. Cette fonction retourne la valeur calculée.
  - 2ème Partie : Exécutez cette fonction avec 3 paramètres en entrée et affichez le résultat dans une boite de dialogue. Même opération avec 5 paramètres. Même opération avec 10 paramètres.

EXERCICE 3/11 : ECRIRE UN ALGORITHME QUI UTILISE LE MOT-CLE FOR

- Préparation :
  - Idem
- Présentation :
  - Le mot clé `for` permet de simplifier l'écriture de boucles avec compteur. Par exemple :

```
var compteur = 0 ;
while ( compteur < 5 ) {
     // Début des instructions.
      // …
      // Fin des instructions.
      compteur ++ ;
}
```

Devient

```
for (var compteur = 0 ; compteur < 5 ; compteur++) {
              // Début des instructions.
      // …
      // Fin des instructions.
}
```

- Enoncé :
  - Vous réalisez le programme suivant :
    Demandez à l'utilisateur de saisir un nombre. TANT QUE il saisit un nombre, demandez-lui de saisir un autre nombre. Chaque valeur fournie par l'utilisateur doit être ajoutée dans un « tableau ».
    Quand l'utilisateur cesse de saisir des nombre ou annule, le programme utilise la fonction créée lors de l'exercice précédent pour afficher la moyenne des nombre saisis.

EXERCICE 4/11 : DEMANDER L'EXECUTION DE FONCTIONS DE FAÇON ASYNCHRONE

- Préparation :
  - Idem
- Présentation :
  - Pour cet exercice vous devrez utiliser les « fonctions » `setTimeout` et `setInterval` fournies par le langage.
  - La fonction `setTimeout` permet de transmettre au moteur du langage une fonction qui sera exécutée **par** le moteur du langage après un certain délai. Cette fonction prend en paramètre :
    - La fonction **à exécuter** sous la forme d'une fonction ;
    - Le délai d'attente avant son exécution sous la forme d'un nombre (en millisecondes).

    La fonction `setTimeout` retourne un nombre. Ce nombre est un identifiant unique pour ce « compte à rebours ».
    Cet identifiant unique peut être utilisé en paramètre de la fonction `clearTimeout` pour arrêter le décompte avant l'exécution de la fonction fournie à `setTimeout`.
    Par exemple :
    ```
    setTimeout(function(){
          // Cette fonction sera exécutée dans 5 secondes.
    }, 5000);
    ```
  - La fonction `setInterval` permet de transmettre au moteur du langage une fonction qui sera exécutée **par** le moteur du langage à intervalle de temps régulier. Cette fonction prend en paramètre :
    - La fonction **à exécuter** sous la forme d'une fonction ;
    - Le délai d'attente entre chaque exécution de cette fonction sous la forme d'un nombre (en millisecondes).

    La fonction `setInterval` retourne un nombre. Ce nombre est un identifiant unique pour cette « exécution périodique ».
    Cet identifiant unique peut être utilisé en paramètre de la fonction `clearInterval` pour arrêter l'exécution périodique de la fonction fournie à `setInterval`.
    Par exemple :
    ```
    setInterval(function(){
          // Cette fonction sera exécutée toutes les 5 secondes
    }, 5000);
    ```
  - On dit que ces fonctions permettent d'exécuter du code de façon **asynchrone** par rapport au code principal. En effet, les fonctions fournies à `setTimeout` ou `setInterval` peuvent être exécutées par le moteur du langage en parallèle de l'exécution du code principal.
- Enoncé :
  - 1ère Partie : Déclenchez l'affichage d'une boite de dialogue après un délai de 3 secondes.

- 2ème Partie : Déclenchez l'affichage d'une boite de dialogue toutes les 5 secondes.
- 3ème Partie : Quand l'utilisateur clique sur un paragraphe déclenchez l'apparition d'une boite de dialogue toutes les secondes. Et, quand l'utilisateur clique sur un autre paragraphe arrêtez l'apparition périodique de cette boite de dialogue. Dans la fonction qui arrête l'apparition périodique vous devrez avoir accès à la variable qui contient l'identifiant d'apparition périodique. Pour ce faire vous devrez créer une variable dans l'espace mémoire global.

EXERCICE 5/11 : CREER UN PIEGE A POP-UP

- Préparation :
  - Idem
- Présentation :
  - L'attribut d'évènement `onbeforeunload` sur la balise `<body>` permet faire en sorte que le navigateur Internet déclenche des instructions juste avant la fermeture du document.
- Enoncé :
  - 1ère Partie : Juste avant la fermeture du document Web, affichez une nouvelle fenêtre vers un site Internet de votre choix. Pensez à désactiver le bloqueur de Pop-Up de votre navigateur.
  - 2ème Partie : Créez 5 documents Web.
    - Au chargement du 1er document Web, affichez un Pop-Up vers le 2ème document Web ;
    - Juste avant la fermeture du 2ème document Web, déclencher l'apparition d'une Pop-Up vers le 3ème.
    - Juste avant la fermeture du 3ème document Web, déclencher l'apparition d'une Pop-Up vers le 4ème.
    - Juste avant la fermeture du 4ème document Web, déclencher l'apparition d'une Pop-Up vers le 5ème.
    - Juste avant la fermeture du 5ème document Web, déclencher l'apparition d'une Pop-Up vers le 1er.
    - Juste avant la fermeture du 1er, déclenchez l'apparition des 5 Pop-Ups d'un coup !!!

EXERCICE 6/12 : ECRIRE DES CONDITIONS COMPLEXES UNIQUEMENT AVEC LES MOTS CLES IF ET ELSE

- Préparation :
  - Idem
- Présentation :
  - La fonction `parseInt` fournie par le moteur du langage prend en paramètre une donnée. Si cette donnée correspond à un nombre, elle retourne le nombre entier correspondant. Sinon elle retourne `NaN`.
    Par exemple :
    ```
    parseInt("4.7"); // Retourne 4.
    ```

- Enoncé :
  - Au chargement de la page, demandez à l'utilisateur quel jour de la semaine il est né (en donnant un entier entre 1 et 7). L'utilisateur doit forcément donner un nombre entre 1 et 7.
    - SI il répond lundi, vous affichez une réponse dans une boite de dialogue.

- SI il répond mardi, mercredi ou jeudi ; vous affichez une autre réponse dans une boite de dialogue.
- SI il répond vendredi, vous affichez une troisième réponse dans une boite de dialogue.
- SI il répond samedi ou dimanche, vous affichez une quatrième réponse dans une boite de dialogue.

  - Pour réaliser cette exercice vous utilisez en particulier les mot-clé `if` et `else`.

EXERCICE 7/11 : ECRIRE DES CONDITIONS COMPLEXES UNIQUEMENT AVEC LES MOTS CLES SWITCH, CASE, BREAK ET DEFAULT

- Préparation :
  - Idem
- Présentation :
  - Les blocs d'instructions basés sur `if`/`else` permettent seulement de prendre en compte deux cas. En revanche, les blocs d'instructions basés sur `switch` permettent d'évaluer une expression selon un nombre de cas non limité.
  - Un blocs d'instructions basé sur `switch` s'écrit de la façon suivante :

```
switch (expression) {
  case valeur1:
      // Instruction s'exécutant lorsque l'expression correspond à valeur1.
      instructions1;
      break;
  case valeur2:
      // Instruction s'exécutant lorsque l'expression correspond à valeur2.
      instructions 2;
      break;
  ...
  default:
      // Instructions quand aucune des valeurs ne correspond.
      instructions_par_defaut;
}
```

  - Le mot clé `case` permet de définir le test d'égalité stricte qui doit être effectué (par exemple `expression === valeur1`).
  - Le mot clé `break` permet d'arrêter les test et de sortir du `switch`. Son utilisation n'est pas obligatoire.
  - Le mot clé `default` permet de spécifier le cas où aucun des tests précédent n'est concluant.
- Enoncé :
  - Reprenez le programme de l'exercice précédent en utilisant un bloc d'instructions `switch` plutôt que `if`/`else`.

EXERCICE 8/11 : PERFECTIONNER SON UTILISATION DES BOUCLES

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - Créez une fonction qui retourne le nombre de digits d'un nombre **entier** passé en argument. Un « digit » est un des chiffres qui compose le nombre. Par exemple : pour 101, les digits sont 1, 0 et 1. Il y'a donc 3 digits.
  - Pour vous aider : essayez de compter le nombre de divisions par 10 nécessaires avant que le nombre soit inférieur à 0.

- Votre fonction doit produire un résultat correct quel que soit la taille de l'entier fourni.
- Vous ne devez **pas utiliser** de notions qui n'ont pas été étudiées dans cette partie des exercices (comme par exemple la propriété `.length`).

EXERCICE 9/11 : IMPLEMENTER UN TRI-BULLE

- Préparation :
  - Idem
- Présentation :
  - Le « tri-bulle » est un algorithme de tri dans un « tableau ». Il repose sur l'utilisation de 2 boucles imbriquées l'une dans l'autre.
- Enoncé :
  - Créez une fonction qui prend en paramètre un « tableau » contenant des valeurs numériques. Cette fonction trie le tableau par ordre croissant. Cette fonction doit pouvoir accepter un tableau de n'importe quelle taille.
  - Par exemple : `[5, 8, 13, 2]` doit donner `[2, 5, 8, 13]`
  - Vous ne devez **pas utiliser** de notions qui n'ont pas été étudiées dans cette partie des exercices (comme par exemple la méthode `.sort`).

EXERCICE 10/11 : ECRIRE DES CONDITIONS BASEES SUR DES CALCULS DE RESTES DE DIVISIONS ENTIERES

- Préparation :
  - Idem
- Présentation :
  - Une année est bissextile dans 2 cas :
    - Elle est divisible par 4 mais pas par 100

    Ou
    - Elle est divisible par 400
  - Par exemple :
    1900 : n'est pas bissextile, divisible par 100 donc cas 1 invalide, et pas divisible par 400 donc cas 2 invalide.
    2000 : est bissextile, divisible par 400 donc cas 2 validé (mais cas 1 invalide ; pour le OU il suffit qu'un des deux cas soit validé)
    1904 : est bissextile, cas 1 validé.
    2003 : n'est pas bissextile, cas 1 et 2 invalides.
- Enoncé :
  - Vous devez réaliser une fonction qui teste si une année donnée par l'utilisateur est bissextile.
  - Il est possible de faire des tests successifs, ou bien de créer des conditions complexes pour ne faire qu'un seul test.
  - L'opérateur % permet de calculer le reste des divisions entières successives du premier opérande par le second (Par exemple : `10%2` donne `0`, `10%3` donne `1`)

EXERCICE 11/11 : ECRIRE DES RECURSIONS

- Préparation :
  - Idem
- Présentation :
  - Une récursion est une fonction qui s'exécute elle-même.
  - On utilise généralement la récursivité en remplacement des boucles.

- Pour qu'un récursion s'arrête, il faut impérativement une condition de sortie. Une condition de sortie est une condition qui, si elle est validée entraîne la fin de l'exécution de la fonction à l'aide du mot clé `return`.

- Enoncé :
  - Reprenez les boucles des exercices 2, 3, 8 et 9 de cette série et remplacez-les par des fonctions récursives avec conditions de sortie.

2. Fondamentaux et techniques avancées en ES3/5

a. Les Objets

Objectif : Savoir créer et manipuler des objets

EXERCICE 1/8 : MANIPULER UN (OBJET) LITTERAL 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 :
  - Idem
- Enoncé :
  - Reprenez la déclaration de littéral suivante dans votre balise `<script>` :

```
var schmilblick = {
        taille:"grand",
        dimensions:{
          largeur:2.5,
          longueur:3,
          hauteur:0.02
        },
        couleurs: [
                "rouge",
                "jaune",
                "or",
                "bleu"
          ],
        peutVoler: false,
        matiere: "laine",
        queSuisJe: function mystereEtBouleDeGomme(){
          alert("A votre avis que suis-je ?");
        }
};
```

  - VOUS NE DEVEZ PAS MODIFIER CETTE DECLARATION EN REPONDANT AUX QUESTIONS QUI SUIVENT.
  - Quelle est la taille du `schmilblick` ? Affichez dans une boite de dialogue la taille du `schmilblick`.
  - Quelle est la matière dont est composé le `schmilblick` ? Affichez dans une boite de dialogue la matière du `schmilblick`.
  - Quelle est la superficie du `schmilblick` ? Affichez dans une boite de dialogue la superficie du `schmilblick` (largeur x longueur).

- De quel couleur est le `schmilblick` ? Affichez successivement dans des boites de dialogue les différentes couleurs du `schmilblick`.
- Le `schmilblick` peut voler ! Changez la valeur de la propriété `peutVoler` du `schmilblick` (sans modifier la déclaration).
- Le `schmilblick` est magique ! Ajoutez une propriété `estMagique` au `schmilblick` avec comme valeur `true` (sans modifier la déclaration).
- Exécutez la méthode `queSuisJe` du `schmilblick`.

EXERCICE 2/8 : MANIPULER UN (OBJET) LITTERAL COMPLEXE

- 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 :
  - Idem
- Enoncé :
  - Reprenez la déclaration de littéral suivante dans votre balise `<script>` :

```
var blender = {
    mixerDesNombres: function(a, b, c){
        if("number" === typeof a && "number" === typeof b && "number" === typeof c){
            var d = (a * b) + c;
            alert(d);
            return d;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDesChaines: function(i, j){
        if("string" === typeof i && "string" === typeof j){
            var k = "Bonjour " + i + j + "!";
            alert(k);
            return k;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDesBooleens: function(x, y, z){
        if("boolean" === typeof x && "boolean" === typeof y && "boolean" === typeof z){
            var message = "";
            if(x && y && z){
                message = "Tout est vrai !";
            }else{
                message = "Au moins un argument n'est pas vrai !";
            };
            alert(message);
            return message;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDesTableaux: function(g, z){
        if("object" === typeof g && "object" === typeof z){
            var t = g[2] + z[1];
            var u = g[1] + z[2];
```

```
            var v = t * u;
            alert("Si ceci : " + v + " est un nombre vous avez réussi !");
            return v;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDesObjets: function(o, t){
        if("object" === typeof o && "object" === typeof t){
            var p = (o.a * o.b) / (t.x * t.y);
            alert("Si ceci : " + p + " est un nombre vous avez réussi !");
            return p;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDesFonctions: function(z, w){
        if("function" === typeof z && "function" === typeof w){
            z();
            w();
            alert("Bravo ! Les types de données fournis sont corrects .");
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    },
    mixerDeTout: function(k, l, m, n){
        if("string" === typeof k && "object" === typeof l && "function" ===
typeof m && "number" === typeof n){
            alert("Voici un message : " + k);
            alert("Voici un message trouvé dans un objet : " + l.message);
            m();
            alert("Mais ceci est un nombre : " + n);
            return n;
        }else{
            alert("Les types de données fournis sont incorrects !");
        };
    }
};
```

- VOUS NE DEVEZ PAS MODIFIER CETTE DECLARATION EN REPONDANT AUX QUESTIONS QUI SUIVENT.
- L'objet littéral ci-dessus contient plusieurs méthodes. Ces méthodes prennent en paramètre des valeurs et en retournent d'autres. Exécutez chacune de ces méthodes en fournissant les bons types de données à l'exécution.

EXERCICE 3/8 : CREER UN (OBJET) LITTERAL AVEC PLUSIEURS METHODES

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - 1ère Partie :
    - Dans les exercices de syntaxe et d'algorithmie, vous avez créé plusieurs fonctions. Vous devez regrouper toutes ces fonctions (au moins 5 d'entre-elles) dans un objet littéral. Déclarez un objet dans lequel chaque propriété contiendra une fonction parmi celles créées précédemment.
    - Exécutez chacune des méthodes créées dans cet objet pour les tester.
  - 2ème Partie :

- Déclarez 1 objet littéral qui contient :
  - 1 propriété qui contient une méthode :
    - qui prend en argument 1 nombre ;
    - qui affiche ce nombre dans une boîte de dialogue ;
    - qui RETOURNE ce nombre multiplié par 100.
  - Exécutez cette méthode et AFFICHEZ la VALEUR DE RETOUR dans une boîte de dialogue.
  - 1 propriété qui contient une méthode :
    - qui prend en argument 1 texte ;
    - qui affiche ce texte dans une boîte de dialogue ;
    - qui RETOURNE ce texte concaténé avec la chaîne de caractère `"hello !"`.
  - Exécutez cette méthode et AFFICHEZ la VALEUR DE RETOUR dans une boîte de dialogue.
  - 1 propriété qui contient une méthode :
    - qui prend en argument un tableau de 3 indices ;
    - qui affiche le contenu à l'indice 1
    - qui remplace ce que contient l'indice 1 par ce qui est contenu à l'indice 2.
    - qui RETOURNE ce tableau de 3 indices modifié.
  - Exécutez cette méthode et AFFICHEZ la valeur du 2ème indice de la VALEUR DE RETOUR dans une boîte de dialogue.
  - 1 propriété qui contient une méthode :
    - qui prend en argument un objet de 5 propriétés qui contiennent chacune un nombre ;
    - qui met dans la 4ème propriété la somme de chacun des nombre de chacune des propriétés.
    - qui RETOURNE 1 objet de 5 propriétés.
  - Exécutez cette méthode et AFFICHEZ la valeur de la 4ème propriété de la VALEUR DE RETOUR dans une boîte de dialogue.

EXERCICE 4/8 : CREER ET MANIPULER PLUSIEURS (OBJETS) LITTERAUX IMBRIQUES

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - 1ère Partie : Créez 1 objet littéral. Cet objet représente un **enseignant**.
    Il a 5 propriétés : son nom, son prénom, son âge, sa discipline enseignée, sa présence.
    - Son âge est un nombre ;
    - Sa présence est un booléen ;
    - Son nom est une chaîne de caractères ;
    - Son prénom est une chaîne de caractères ;
    - Sa discipline enseignée est une chaîne de caractères.
  - 2ème Partie : Créez 1 objet littéral. Cet objet représente une **classe**.
    Il a 1 propriété : son numéro, sa spécialité, son enseignant.
    - Son numéro est un nombre ;
    - Sa spécialité est une chaîne de caractères ;

- Son enseignant est un objet représentant un **enseignant**.

- 3ème Partie : Créez 3 objets littéraux. Ces objets représentent des **élèves**. Chaque élève a 4 propriétés : son nom, son prénom, son âge, sa présence.
  - Son âge est un nombre ;
  - Sa présence est un booléen ;
  - Son nom est une chaîne de caractères ;
  - Son prénom est une chaîne de caractères.

- 4ème Partie : Ajoutez une nouvelle propriété dans l'objet représentant la **classe**. Cette propriété contiendra les **élèves**. Elle devra contenir les 3 références vers les 3 objets représentant chacun des 3 élèves.
  - Les **élèves** est un objet de type « tableau ».

- 5ème Partie : Changez la propriété de présence du 2ème élève de la liste d'élèves de la classe sans toucher à la déclaration.

- 6ème Partie : Réalisez l'algorithme suivant :
  - POUR CHAQUE élève dans la classe, AFFICHEZ dans une boite de dialogue le message (où X et Y sont les prénoms et noms de l'élève considéré) : "Je suis X Y et je suis présent ".

- 7ème Partie : Assurez-vous que les booléens de présence des élèves ne sont pas tous les mêmes. Réalisez l'algorithme suivant :
  - POUR CHAQUE élève dans la classe, SI sa propriété présence est vraie, AFFICHEZ dans une boite de dialogue le message (où X et Y sont les prénoms et noms de l'élève considéré) : "Je suis X Y et je suis présent".

EXERCICE 5/8 : CREER ET MANIPULER UNE MULTITUDE D'OBJETS A L'AIDE D'UNE FONCTION CONSTRUCTEUR

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - 1ère Partie : Créez une fonction constructeur. Cette fonction vous permettra de construire des objets représentant des **robots**. Les robots ont tous 6 propriétés : un numéro de série, un nom de modèle, un nom d'usage, une couleur, une matière, une durée de garantie.
    - Le numéro de série est un nombre qui doit être fourni pour construire un robot ;
    - Le numéro de modèle est un nombre qui doit être fixé au départ ;
    - Le nom d'usage est une chaîne de caractère qui n'est pas fixé au départ ;
    - La couleur est une chaîne de caractère qui doit être fournie pour construire un robot ;
    - La matière est une chaîne de caractère qui est fixée au départ ;
    - La durée de garantie est une chaîne de caractère qui est fixée au départ.

- 2ème Partie : Créez 12 robots à l'aide de cette fonction constructeur. Chaque robot aura donc un numéro de série et une couleur différente.

- 3ème Partie : Créez un objet de type « tableau » qui sera le magasin. Ce « tableau » devra contenir les 12 références vers les 12 objets représentant chacun des 12 robots.

- 4ème Partie : Réalisez l'algorithme suivant :

    - POUR CHAQUE robot dans le magasin, AFFECTEZ un nom d'usage qui sera la concaténation du modèle, du numéro de série et de l'indice du robot dans le magasin.

EXERCICE 6/8 : CREER UN ET MANIPULER UN OBJET DONT LES METHODES MODIFIENT INTRINSEQUEMENT SES AUTRES PROPRIETES

- Préparation :
    - Idem
- Présentation :
    - Néant
- Enoncé :
    - 1ère Partie : Créez un objet littéral qui représente un robot. Ce robot a 6 propriétés : un numéro de série, un nom de modèle, un nom d'usage, une couleur, une matière, une durée de garantie, un nombre de bonjour et une méthode qui lui permet de dire bonjour à l'aide d'une boite de dialogue.
        - Le numéro de série est un nombre ;
        - Le numéro de modèle est un nombre ;
        - Le nom d'usage est une chaîne de caractère ;
        - La couleur est une chaîne de caractère.
        - La matière est une chaîne de caractère ;
        - La durée de garantie est une chaîne de caractère ;
        - Le nombre de bonjour est un nombre (de base on le fixera à 0) ;
        - Dire bonjour est une fonction.

    - 2ème Partie : Exécutez la méthode qui permet au robot de dire bonjour. Elle doit entraîner l'affichage d'une boite de dialogue avec le texte "Bonjour humain !" écrit dedans.

    - 3ème Partie : Modifiez la méthode dire bonjour pour que le robot incrémente le nombre de bonjour de 1 à chaque fois qu'on exécute la méthode dire bonjour. Vous pouvez utiliser le mot clé `this` pour accéder à la propriété nombre de bonjour lorsque vous programmez dans la méthode.

    - 4ème Partie : Exécutez 3 fois de suite la méthode dire bonjour et vérifiez dans l'onglet console de votre navigateur Internet que la propriété nombre de bonjour de votre robot à bien été modifiée en conséquence.

    - 5ème Partie : Modifiez la méthode dire bonjour pour que le robot affiche une boite de dialogue avec le texte "Bonjour humain ! Je m'appelle X" ou X est un mot constitué de la concaténation du modèle et du numéro de série du robot. Vous devez utiliser le mot clé `this` pour accéder à ses propriétés lorsque vous programmez dans la méthode.

- 6ème Partie : Exécutez la méthode dire bonjour.

EXERCICE 7/8 : MANIPULER DES OBJETS EN UTILISANT LA CONSOLE

- Préparation :
  - Idem
- Présentation :
  - Pensez à utiliser fréquemment la console pour consulter le contenu des objets qui sont référencés dans les variables.
- Enoncé :
  - Reprenez les 2 déclarations de littéraux suivantes dans votre balise `<script>` :

```
var monObjet = {
  e:function(){
    alert(this.t);
  },
  r:true,
  t:"Ceci est un texte dans un objet"
};
```

```
var monAutreObjet = {
  e:[function(){
    alert("Vous êtes ici !");
  },true],
  r:false,
  t:function(){
    this.y = {
      e:function(){
        alert("Ou suis je ? Qui suis je ?");
      }
    };
    alert("Desormais ma propriété y contient un objet");
  }
};
```

  - VOUS NE DEVEZ PAS MODIFIER CETTE DECLARATION EN REPONDANT AUX QUESTIONS QUI SUIVENT.

  - Les différentes parties sont interdépendantes. Ne commentez pas chaque partie après l'avoir écrite.

  - 1ère Partie : Affichez dans une boite de dialogue le contenu de la propriété `r` de l'objet référencé dans la variable `monObjet`.

  - 2ème Partie : Affichez dans une boite de dialogue le contenu de la propriété `r` de l'objet référencé dans la variable `monAutreObjet`.

  - 3ème Partie : Exécutez la méthode située en première position dans le « tableau » dans la propriété `e` de l'objet référencé dans la variable `monAutreObjet` (si vous réussissez le texte "Vous êtes ici !" devrait s'afficher).

  - 4ème Partie : Mettez l'objet référencé dans la variable `monAutreObjet` dans une propriété `y` de l'objet référencé dans la variable `monObjet` (Vous allez donc mettre un objet dans une - nouvelle - propriété d'un autre objet).

- 5ème Partie : Déclenchez la méthode située dans la propriété `t` dans l'objet référencé dans la propriété `y` contenue dans l'objet référencé dans la variable `monObjet`. Si vous réussissez, le message "Désormais ma propriété y contient un objet" devrait s'afficher.

- 6ème Partie : Déclenchez la méthode située dans la propriété `e` de l'objet nouvellement référencé dans la propriété `y` lors de l'exécution de la précédente méthode. Si vous réussissez, vous devriez voir s'afficher successivement les textes : "Ou suis-je ? Qui suis-je ?".

EXERCICE 8/8 : CREER UN OBJET LITTERAL CONTENANT UNE REFERENCE CIRCULAIRE

- Préparation :
  - Idem
- Présentation :
  - Néant
- Enoncé :
  - 1ère Partie : Créez 1 objet littéral qui représente une **maison**. Cet objet contient une propriété `adresse` qui contient l'adresse de la maison et une propriété `chambre` qui ne contient l'objet nul (`null`).

  - 2ème Partie : Créez 1 objet littéral qui représente une **chambre**. Cet objet contient une propriété `lit` qui contient la chaîne de caractère *matelas* et une méthode `quelEstMonAdresse` qui ne contient pas d'instructions pour l'instant.

  - 3ème Partie : Mettez la référence à l'objet littéral qui représente la **chambre** dans la propriété `chambre` de l'objet littéral qui représente la **maison**.

  - 4ème Partie : Affichez dans une boite de dialogue le contenu de la propriété `lit` de l'objet représentant la **chambre** en partant de l'objet représentant la **maison**.

  - 5ème Partie : Je souhaite exécuter la méthode `quelEstMonAdresse` de l'objet représentant la **chambre** pour afficher l'adresse de la maison contenue dans la propriété `adresse` de l'objet représentant la **maison**. En d'autres termes, comment puis-je faire pour accéder à la propriété de la **maison** (objet « parent ») à partir d'une méthode de la **chambre** (objet « enfant ») ?
    - Vous savez que vos objets sont uniques en mémoire. Vous ne manipulez que des références.
    - Vous savez que vos objets peuvent contenir n'importe quelle valeur dans leurs propriétés.
    - Vous savez que le mot clé `this` fait référence à l'objet dans lequel le mot clé est employé.
  - Indice : vous devez créer une nouvelle propriété dans l'objet **chambre** dans laquelle vous mettrez une référence à l'objet **maison**.

b. Le Prototypage

Objectif : Savoir créer et utiliser des prototypes

EXERCICE 1/2 : CREER PLUSIEURS OBJETS BASES SUR UN OBJET LITTERAL QUI SERT DE PROTOTYPE

- 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 :

  Dans cet exercice vous devrez utiliser l'objet fondamental `Object` et en particulier la méthode `.create`.
  - La méthode `Object.create()` permet de créer un objet à partir d'un prototype. Par exemple :
    ```
    var enfant = Object.create({ prenom: "Anakin", nom: "Skywalker"});
    ```
  - On peut ensuite définir les propriétés propres de l'objet « enfant ». Par exemple :
    ```
    enfant.prenom = "Luke";
    ```
  - On peut également utiliser le 2ème paramètre (optionnel) de la méthode `.create` pour créer un objet à partir d'un prototype et en même temps définir les propriétés propres de l'objet « enfant ». Ce 2ème paramètre s'appelle un objet de définition de propriétés et est défini dans la documentation officielle du langage sur le MDN. Par exemple :
    ```
    var enfant = Object.create({ prenom: "Anakin", nom: "Skywalker"}, { prenom : {value : "Luke", writable: true, enumerable: true, configurable: true}});
    ```
- Enoncé :
  - 1ère Partie : Créez un objet littéral qui représente un avion Concorde. Cet objet est doté de plusieurs propriétés :
    - Son modèle qui une chaine de caractère au choix ;
    - Sa capacité qui est un nombre au choix ;
    - Son prix qui est un nombre au choix ;
    - Ses passagers qui est un objet « tableau » vide initialement : `[]` ;
    - Son réservoir qui est un nombre fixé : `100` ;
    - Sa couleur qui est un objet « tableau » dont le contenu est : `["blanc"]`.
  - 2ème Partie : Cet objet est l'avion de base tel qu'il est livré. Chaque compagnie aérienne qui achète un avion le personnalise. Utilisez l'objet fondamental `Object` pour créer un nouvel avion Airbus à partir de ce prototype. Ce nouvel avion est vendu à Air France. Vous ajouterez une propriété `compagnie` qui contiendra le nom de la compagnie sous la forme d'une chaîne de caractères.
  - 3ème Partie : Effectuez la même opération mais pour un avion destiné à British Airways.

EXERCICE 2/2 : CREER PLUSIEURS OBJETS BASES SUR DES FONCTIONS CONSTRUCTEURS DONT UN SERVIRA DE PROTOTYPE AUX AUTRES

- Préparation :
  - Idem
- Présentation :
  - Dans cet exercice, vous ne devez utiliser **que** des *fonctions constructeur*.
  - Les *fonction constructeur* permettent de définir la structure des objets qui seront créés à l'aide de cette *fonction constructeur*. Comme toutes les fonctions, les *fonctions constructeur* sont également des objets qui possèdent de propriétés.

- On peut utiliser la propriété `.prototype` d'une fonction constructeur pour définir l'objet qui sera le prototype des objets créés à l'aide de cette fonction constructeur.

- Enoncé :
  - 1ère Partie : Écrivez une fonction constructeur qui permet de construire des objets représentant des mammouths. Les mammouths avaient plusieurs caractéristiques :
    - Leur âge qui est un nombre ;
    - Leur poids qui est un nombre ;
    - Leur espèce qui est une chaîne de caractères ;
    - Le fait qu'ils aient une fourrure, qui est un booléen ;
    - Leur genre (mâle ou femelle) qui est une chaîne de caractères ;
    - L'action de barrir qui est une méthode.
  - 2ème Partie : Les éléphants contemporains héritent probablement des caractéristiques d'un mammouth qui est leur prototype. Écrivez une fonction constructeur qui permet de construire des objets représentant des éléphants. Les éléphants ont des caractéristiques qui différent des mammouths :
  - Leur poids qui est un nombre ;
  - Leur espèce qui est une chaîne de caractère ;
  - Le fait qu'ils n'aient pas de fourrure, qui est un booléen ;

  Et ils ont une caractéristique supplémentaire :
  - L'action de manger des cacahuètes qui est une méthode.

- 3ème Partie : Créez 2 éléphants. Vous devez pouvoir exécuter la méthode qui leur permet de barrir et qui provient de leur prototype et, exécuter la méthode qui leur permet de manger des cacahuètes et qui leur est propre.

c. Les Objets fondamentaux

Objectif : Savoir utiliser des objets fondamentaux

EXERCICE 1/4 : UTILISER DES OBJETS FONDAMENTAUX DU LANGAGE

- 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 :
  - En JavaScript la plupart des fonctions de base du langage sont en réalité des méthodes disponibles dans des objets. On dispose par exemple, de base, d'un objet en mémoire référencé dans la variable `Math` (voir le MDN à son sujet ici). Cet objet comporte un ensemble de méthodes pour réaliser des calculs mathématiques.
  - Exemple issu de la documentation :
    - Méthode : `Math.round()`
    - Présentée de la sorte :

```
Integer Math.round(Float x)
```

A gauche le type de données retournée : un entier (`Integer`).
A droite le type de données à fournir : un « flottant » (`Float x`).
Un « flottant » est un réel (nombre à virgule).

- Retourne l'arrondi (entier) du réel passé en paramètre.

- Enoncé :
  - 1ère Partie : Obligez l'utilisateur à saisir un nombre. Affichez ce nombre dans une boite de dialogue l'arrondi de ce nombre (reprendre l'exercice que vous avez réalisé dans la partie JavaScript « Syntaxe et Algorithmie » et l'améliorer en utilisant l'objet `Math`).
  - 2ème Partie : Les méthodes `.ceil()` et `.floor()` de l'objet `Math` permettent d'obtenir respectivement l'arrondi à l'inférieur et l'arrondi au supérieur. Affichez dans une boite de dialogue le nombre saisi arrondi à l'inférieur puis le nombre saisi arrondi au supérieur.

EXERCICE 2/4 : UTILISER LE MECANISME DE COERCITION AVEC DES TYPES PRIMITIFS

- Préparation :
  - Idem.
- Présentation :
  - En JavaScript, un mécanisme de **coercition** transforme les types primitifs en objets en cas de besoin. Le moteur JavaScript peut transformer :
    - Un nombre en objet de type `Number` (i.e. à l'aide de la **fonction constructeur** `Number`) ;
    - Un booléen en objet de type `Boolean` (i.e. à l'aide de la **fonction constructeur** `Boolean`) ;
    - Une chaîne de caractère en objet de type `String` (i.e. à l'aide de la **fonction constructeur** `String`).
  - La liste des propriétés et méthodes disponibles pour les objets de type `String` est disponible sur le MDN ici.
- Enoncé
  - 1ère Partie : Obligez l'utilisateur à saisir une chaîne de caractères dans une boite de dialogue. Un fois la saisie validée. Affichez dans une boite de dialogue le nombre de caractères de la chaîne de caractères saisie (propriété `.length`).
  - 2ème Partie : Utilisez la méthode `.substr()` pour extraire et afficher dans une boite de dialogue tous les caractères de la chaîne saisie à l'exception du premier et du dernier.
  - 3ème Partie : Utilisez la méthode `.split()` pour récupérer chacun des mots de la saisie utilisateur dans un tableau. Affichez séparément chaque mot dans une boîte de dialogue.

EXERCICE 3/4 : CREER ET UTILISER DES OBJETS BASES SUR DES FONCTIONS CONSTRUCTEURS FONDAMENTALES DU LANGAGES – 1ERE PARTIE

- Préparation :
  - Idem.
- Présentation :
  - En JavaScript, les « tableaux » sont en réalité des objets (issus de la **fonction constructeur fondamentale** `Array`) dont les propriétés ont des noms numériques. La liste des propriétés et méthodes disponibles pour les objets de type `Array` est disponible sur le MDN ici.
- Enoncé :
  - 1ère Partie : Créez un « tableau » contenant 10 mots qui forment une phrase. Utilisez la méthode `.join()` des « tableaux » pour afficher l'ensemble des éléments du tableau sous la forme d'une chaîne de caractères.

- 2ème Partie : Utilisez la méthode `.push()` pour ajouter 3 mots à la fin du « tableau » et la méthode `.shift()` pour retirer les 3 premiers mots du « tableau ». Sauvegardez au fur et à mesure les 3 premiers mots extrait dans un autre tableau à l'aide de la méthode `.push()`.
- 3ème Partie : Utilisez la méthode `.map()` pour exécuter une fonction qui met en majuscule (en utilisant la méthode `.toUpperCase()` de `String`) chaque mot du dernier « tableau ».

EXERCICE 4/4 : CREER ET UTILISER DES OBJETS BASES SUR DES FONCTIONS CONSTRUCTEURS FONDAMENTALES DU LANGAGES – 2EME PARTIE

- Préparation :
  - Idem.
- Présentation :
  - En JavaScript la plupart des fonctions de base du langage sont en réalité des méthodes disponibles dans des objets. On dispose par exemple, de base, d'une **fonction constructeur** `Date`. Cette fonction constructeur permet d'obtenir un objet contenant un ensemble de méthodes pour gérer une date.
  - Si on exécute cette fonction constructeur sans paramètres on obtient un objet de type `Date` qui contient la date au moment de l'exécution de l'instruction.
  - Sinon, on peut fournir les informations relatives à la date souhaitée en paramètre. Cette fonction constructeur est documentée sur le MDN ici.
- Enoncé :
  - 1ère Partie : Créez un nouvel objet de type `Date` dans la mémoire.
  - 2ème Partie : Consultez la liste des méthodes de l'objet construit avec la fonction constructeur `Date` à l'adresse fournie.
  - 3ème Partie : Utilisez l'objet `Date` créé pour afficher dans une boite de dialogue le nombre d'heures, de minutes et de secondes avant la fin du cours à `17:00:00`.

d. Le Modèle Objet du Document

Objectif : Savoir utiliser le BOM (Browser Object Model) et le DOM (Document Object Model)

EXERCICE 1/23 : APPRENDRE A UTILISER LA CONSOLE DU NAVIGATEUR

- 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 :
  - Le navigateur Internet met à la disposition du moteur JavaScript une **interface** permettant d'avoir accès à certaines informations relatives au navigateur Internet, à l'historique de navigation, à la fenêtre et au document.
  - On appelle cette **interface** le BOM (**B**rowser **O**bject **M**odel). Elle se présente sous la forme d'un objet de type `Window` référencé dans la variable globale `window`.
  - L'objet `window` contient des propriétés référençant des « sous » objets que nous allons exploiter.
  - Un des « sous » objets de l'objet `window` est référencé dans la propriété `.console`. Il s'agit d'un objet de type `Console`. On peut donc écrire

`window.console` pour pointer vers cet objet. Ce « sous » objet est documenté ici dans le MDN.

- Enoncé :
  - 1ère Partie : Utilisez la méthode `.log()` de l'objet `console` pour afficher un texte dans la console de debug du navigateur Internet.
  - 2ème Partie : Accédez à la page d'accueil des technologies Web sur le MDN et regardez le contenu de la console de votre debugger.
  - 3ème Partie : Accédez à la page d'accueil de facebook et regardez le contenu de la console de votre debugger.
  - Vous savez maintenant comment sont affichés les messages que vous avez pu voir apparaître dans la console. On utilise la méthode `.log()` de l'objet `console` pour debugger un programme de façon non intrusive (sans bloquer le programme comme avec la méthode `.alert()`). Vous pouvez l'utiliser pour afficher le contenu de variables par exemple.
  - 4ème Partie : On peut utiliser des caractères de substitution sur les messages produits dans la console. Par exemple, le caractère de substitution `%c` peut être utilisé pour définir les styles CSS à appliquer sur le message produit et ainsi améliorer sa lisibilité. Inspirez-vous des exemples disponibles sur le MDN pour afficher un message en gras, vert et en 20px dans la console.

EXERCICE 2/23 : DETECTER LE NAVIGATEUR A L'AIDE DE L'OBJET NAVIGATOR

- Préparation :
  - Idem.
- Présentation :
  - Un des « sous » objets de l'objet `window` est référencé dans la propriété `.navigator`. Il s'agit d'un objet de type `Navigator`. On peut donc écrire `window.navigator` pour pointer vers cet objet. Ce « sous » objet est documenté ici dans le MDN.
- Enoncé :
  - 1ère Partie : Affichez dans une boite de dialogue le contenu de la propriété `.appName` de l'objet référencé dans `window.navigator`. Puis affichez le contenu de la propriété `.userAgent` dans une boite de dialogue. Qu'est-ce qui s'affiche sur Mozilla Firefox, Chrome et Edge ?
  - 2ème Partie : Utilisez une de ces propriétés pour réaliser l'algorithme suivant :

    SI je démarre mon script sur Edge, j'affiche une boite de dialogue avec le texte "So Lame ! " à l'intérieur et j'ouvre une PopUp vers le site de Mozilla Firefox.
    SINON SI je démarre mon script sur Chrome j'affiche une boite de dialogue avec le texte "Spy On Me !" et j'ouvre une PopUp vers la page wikipédia des navigateurs Web.
    SINON j'affiche une boite de dialogue avec le texte "Freed from Desire".

EXERCICE 3/23 : APPRENDRE A MODIFIER UNE PROPRIETE DU DOM

- Préparation :
  - Idem.
- Présentation :
  - L'objet `window` contient des propriétés qui contiennent des informations relatives à la fenêtre ou onglet de navigateur. Cet objet est documenté ici dans le MDN. Si on modifie ces informations en mémoire, l'affichage change en conséquence. Nous

allons nous intéresser à la propriété `.title` du sous objet `window.document`. Cette propriété contient le titre de l'onglet courant.

- Enoncé :
  - 1ère Partie : Affichez dans une boite de dialogue le contenu de la propriété `.title` de l'objet `window.document`.
  - 2ème Partie : Écrivez l'algorithme suivant :
    - AU CLIC sur le titre de la première section, on affecte la valeur "Ceci est un nouveau titre" à la propriété `.title` de l'objet `window.document`.
  - 3ème Partie : Écrivez l'algorithme suivant :
    - AU CLIC sur le titre de la deuxième section, SI le titre est "Ceci est un nouveau titre" ALORS affectez le titre initial à la propriété `.title` de `window.document` SINON affectez la valeur "Ceci est un nouveau titre".

EXERCICE 4/23 : ETUDIER LE MECANISME DE CHARGEMENT DU DOM

- Préparation :
  - Idem.
- Présentation :
  - Nous allons nous intéresser à la propriété `.document` de l'objet `window`. Cette propriété contient un objet de type `Document` qui contient toutes les informations relatives au document Web. On qualifie cet objet et ses « sous » objets de **DOM** (**D**ocument **O**bject **M**odel). Cet objet est documenté sur le MDN ici.
- Enoncé :
  - 1ère Partie : La propriété `body` du « sous » objet `document` de l'objet `window` contient l'objet correspondant à la balise `<body>` du document Web affiché. Affichez le contenu de la propriété `.body` à l'aide de la méthode `.log()` du sous objet `console`, à l'aide de la méthode `.dir()` du sous objet `console`.
  - 2ème Partie : Vous avez pu constater que cette propriété contient `null`. En effet, lorsque votre script est exécuté dans l'en-tête du document Web, le navigateur n'A PAS ENCORE LU ET CHARGÉ en mémoire la suite du document Web. A partir de maintenant vous avez 2 possibilités :
    - Soit vous programmez dans une balise `<script>` que vous positionnerez juste avant la balise `</body>` fermante ;
    - Soit vous créez une fonction dans laquelle vous programmerez et vous exécuterez cette fonction AU CHARGEMENT du document à l'aide de l'attribut d'évènement `onload` sur la balise `<body>`.
  - 3ème Partie : Affichez le contenu de la propriété `.body` à l'aide de la méthode `.log()` du sous objet `console`, à l'aide de la méthode `.dir()` du sous objet `console`, en respectant les consignes proposées précédemment.
  - 4ème Partie : La propriété `.body` contient donc un objet de type `HTMLBodyElement`. Affectez la chaîne de caractère "lime" à la propriété `.bgColor` de l'objet référencé dans la propriété `.body`.

EXERCICE 5/23 : UTILISER UNE METHODE DU DOM POUR ACCEDER A UN ELEMENT

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - 1ère Partie : La méthode `.getElementById()` de l'objet `document` prend en argument une chaîne de caractère qui correspond à l'identifiant d'une balise du

document Web. Elle retourne une référence vers l'objet correspondant en mémoire. Identifiez une balise du document Web (utilisez l'attribut `id`). Une fois la balise identifiée, écrivez l'algorithme suivant :

- AU CHARGEMENT du document, récupérez une référence vers l'objet correspondant à la balise identifiée et affichez le contenu de cet objet dans la console.

- 2ème Partie : Vous savez désormais récupérer une référence à l'objet correspondant à une balise identifiée. Identifiez une autre balise du document Web (toujours en utilisant l'attribut `id`). Une fois la balise identifiée, écrivez l'algorithme suivant :
  - AU CLIC sur le titre principal, récupérez une référence vers l'objet correspondant à la balise identifiée. Cet objet contient des propriétés et des méthodes. Affectez à la propriété `.innerHTML` de l'objet correspondant à la balise la valeur suivante : "Ce texte est écrit dans cette balise par mon script".

EXERCICE 6/23 : FAIRE LE LIEN ENTRE ATTRIBUTS HTML ET PROPRIETE DU DOM – PARTIE 1

- Préparation :
  - Créez un document `HTML` valide contenant le code suivant entre ses 2 chevrons `<body>` :

```
<h1>Titre principal</h1>
<ul>
  <li>Premier point</li>
  <li>Deuxième point</li>
  <li>Troisième point</li>
</ul>
<div>
  <div>
    <h2>
      <span>1</span>. Titre de la première section
    </h2>
    <p>
      <span>1.1</span>. Premier paragraphe de la première section et un <a href="#">lien</a>.
    </p>
    <p>
      <span>1.2</span>. Deuxième paragraphe de la deuxième section et un <a href="#">lien</a>.
    </p>
  </div>
  <div>
    <h2>
      <span>2</span>. Titre de la deuxième section
    </h2>
    <p>
      <span>2.1</span>. Premier paragraphe de la deuxième section.
    </p>
    <p>
      <span>2.2</span>. Deuxième paragraphe de la deuxième section.
    </p>
  </div>
  <div>
    <h2>
      <span>3</span>. Titre de la troisième section
    </h2>
    <p>
      <span>3.1</span>. Premier paragraphe de la troisième section.
    </p>
    <p>
      <span>3.2</span>. Deuxième paragraphe de la troisième section.
    </p>
  </div>
</div>
```

  - 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 éléments `HTML` du document Web sont lus et chargés en mémoire par le navigateur Internet. Le navigateurs Internet charge les éléments HTML sous la forme d'objets en mémoire, comme on a pu le constater dans l'exercice précédent.
  - Les **noms des propriétés** de ces objets correspondent aux **noms des attributs** `HTML` des balises (à quelques exceptions près). Les objets de type `HTMLElement` sont documentés sur le MDN ici.
- Enoncé :
  - Écrivez l'algorithme suivant : AU CLIC sur le titre de la première section, renseignez les propriétés `href` ainsi que les attributs `title` des 2 objets correspondant aux liens dans les deux paragraphes qui suivent.
  - Écrivez l'algorithme suivant : AU CLIC sur le titre de la deuxième section, assignez une fonction à la propriété `onclick` de l'objet correspondant au deuxième paragraphe de la section.

EXERCICE 7/23 : FAIRE LE LIEN ENTRE ATTRIBUTS HTML ET PROPRIETE DU DOM – PARTIE 2

- Préparation :
  - Idem.
- Présentation :
  - Certains attributs de balise en `HTML` ne conservent pas le même nom lorsqu'ils sont chargés en mémoire.
  - En effet, ils correspondent à des mots clés réservés du langage. Par exemple, l'**attribut** `class` en `HTML` correspond à la **propriété** `className` dans l'objet correspondant dans le modèle objet du document.
- Enoncé :
  - Créez une classe `CSS` qui spécifie plusieurs propriétés de style pour une `<div>` correspondant à une section.
  - Ecrivez l'algorithme suivant : AU SURVOL sur une `<div>`, lui attribuer la classe `CSS` créée. QUANT ON NE SURVOLE PLUS la `<div>`, lui retirer la classe `CSS` créée.

EXERCICE 8/23 : APPRENDRE A MODIFIER LES PROPRIETES DE STYLE DES ELEMENTS – PARTIE 1

- Préparation :
  - Idem.
- Présentation :
  - Les objets correspondants aux éléments `HTML` ont tous une propriété `.style` qui contient une référence à un sous objet.
  - Ce sous objet est de type `CSSStyleDeclaration`. Cet objet contient des propriétés qui correspondent aux propriétés de style de l'attribut `style` de la balise. L'utilisation de cet objet est documentée sur le MDN ici.
- Enoncé :
  - Écrivez le programme suivant : AU CLIC sur le premier `<li>` changez la couleur, le poids et la taille de la police du premier `<span>` du titre de la première section. AU CLIC sur le deuxième `<li>` changez la couleur, le poids et la taille de la police du premier `<span>` du titre de la deuxième section. AU CLIC sur le troisième `<li>` changez la couleur, le poids et la taille de la police du premier `<span>` du titre de la troisième section.
  - Écrivez le programme suivant : AU SURVOL du titre principal, effacez le propriétés de style des 3 `<span>` des 3 titres de section.

EXERCICE 9/23 : APPRENDRE A MODIFIER LES PROPRIETES DE STYLE DES ELEMENTS – PARTIE 2

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - Attribuez la propriété de style `display: none` à chacune des `<div>` du document Web. A l'affichage du document, seul le titre `<h1>` est donc affiché.
  - Écrivez l'algorithme suivant : AU SURVOL de la balise `<h1>`, changer la propriété de style de la `<div>` principale (celle qui contient toutes les sections) pour faire en sorte qu'elle s'affiche. Quand on NE SURVOLE PLUS la balise `<h1>`, changer la propriété de style de la `<div>` principale pour faire en sorte qu'elle ne s'affiche plus.

EXERCICE 10/23 : APPRENDRE A MODIFIER LES PROPRIETES DE STYLE DES ELEMENTS – PARTIE 3

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - Écrivez le programme suivant : AU CLIC sur la balise `<h1>`, la `<div>` principale doit s'afficher. Si on CLIC à nouveau sur le titre, la `<div>` principale doit se masquer. L'algorithme est le suivant : SI la `<div>` est masquée, je l'affiche, SINON je la masque.

EXERCICE 11/23 : APPRENDRE A CREER UNE ANIMATION – PARTIE 1

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - Écrivez le programme suivant : AU CLIC sur le titre principal, ajoutez `2px` à sa propriété de style de marge à gauche. Cela signifie qu'à chaque fois qu'on clique sur le titre principal, sa marge à gauche va augmenter de `2px`.
  - Note :
    - Pour ajouter `2px` à `2px`, vous devez d'abord transformer `2px` en nombre, additionner le nombre obtenu et reconcaténer ce nombre avec le texte `"px"`.
    - La propriété de style correspondant à la marge a une valeur initiale correspondant à une chaine de caractère vide `""`. Si on essaie de convertir en nombre une chaine de caractère vide, on obtient la valeur `NaN`. Il faut donc prévoir de vérifier la valeur initiale de la marge et, si elle n'est pas définie, de la remplacer par une valeur par défaut numérique.

EXERCICE 12/23 : APPRENDRE A CREER UNE ANIMATION – PARTIE 2

- Préparation :
  - Idem.

- Présentation :
  - Idem.
- Enoncé :
  - Écrivez le programme suivant : Au CHARGEMENT du document, toutes les `200ms`, ajoutez `2px` à la propriété de style de marge à gauche du titre principal. Cela signifie que toutes les `200ms`, la marge à gauche du titre principal va augmenter de `2px`.
  - Vous utiliserez la méthode `.setInterval()`.

EXERCICE 13/23 : APPRENDRE A CREER UNE ANIMATION – PARTIE 3

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - Faîtes la même chose que pour l'exercice précédent mais en faisant varier l'opacité du titre.

EXERCICE 14/23 : APPRENDRE A CREER UNE ANIMATION – PARTIE 4

- Préparation :
  - Idem.
- Présentation :
  - Idem.
- Enoncé :
  - Faîtes la même chose que pour l'exercice précédent mais en faisant varier la position `top` du titre. Pensez à passer le positionnement du titre en `absolute`.

EXERCICE 15/23 : INTRODUCTION A LA GESTION DES EVENEMENTS

- Préparation :
  - Créez un document `HTML` valide contenant le code après le chevron fermant `</head>` :

```
<body onload="window.alert(\"Ce message est affiché au chargement du document !\");">
<h1>Titre principal</h1>
<ul>
  <li onmouseover="window.alert(\"Ce message est affiché lorsqu'on survole ce point.\");">Premier point</li>
  <li>Deuxième point</li>
  <li>Troisième point</li>
</ul>
<div>
  <div>
    <h2>
      <span>1</span>. Titre de la première section
    </h2>
    <p>
      <span>1.1</span>. Premier paragraphe de la première section et un <a href="#">lien</a>.
    </p>
    <p>
      <span onmouseout="window.alert(\"Ce message est affiché lorsqu'on ne survole plus cet élément.\");">1.2</span>. Deuxième paragraphe de la deuxième section et un <a href="#">lien</a>.
    </p>
  </div>
  <div>
    <h2>
      <span>2</span>. Titre de la deuxième section
    </h2>
    <p>
      <span>2.1</span>. Premier paragraphe de la deuxième section.
```

```
    </p>
    <p>
      <span>2.2</span>. Deuxième paragraphe de la deuxième section.
    </p>
  </div>
  <div>
    <h2>
      <span>3</span>. Titre de la troisième section
    </h2>
    <p>
      <span>3.1</span>. Premier paragraphe de la troisième section.
    </p>
    <p>
      <span>3.2</span>. Deuxième paragraphe de la troisième section.
    </p>
  </div>
  <form method="http://www.google.com" action="GET">
    <input name="q" type="text" value="" onfocus="window.alert(\"Vous avez sélectionné le
champ de saisie!\");" />
  </form>
</div>
</body>
```

- 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 se passer des attributs d'évènements dans le code `HTML` pour gérer les évènements exclusivement à partir du modèle objet du document.
  - On a la possibilité d'affecter des fonctions aux propriétés qui correspondent aux attributs d'événements dans les objets du DOM. Ces fonctions seront exécutées par le navigateur Internet en réaction à certains événements.
  - Plusieurs éléments `HTML` du document Web que vous allez créer portent des attributs d'évènements. Ces attributs d'évènement vont être lus et chargés en mémoire par le navigateur Internet dans le DOM. Ainsi, lorsque les évènements surviendront, le navigateur Internet exécutera les instructions associées.
- Enoncé :
  - Vous devez gérer tous ces évènements à partir du DOM. L'évènement `onload` par exemple est une méthode (fonction dans une propriété) de l'objet `window`. En revanche, les autres évènements sont des méthodes des objets correspondant aux éléments `HTML`.
  - Supprimez les attributs d'évènements du document Web et faites-en sorte de définir les instructions qui doivent être exécutées en réaction à ces évènements directement en JavaScript en renseignant correctement les bonnes propriétés du DOM.

EXERCICE 16/23 : APPRENDRE A COMBINER ANIMATION ET GESTION DES EVENEMENTS

- Préparation :
  - Créez un document `HTML` valide contenant le code suivant entre ses 2 chevrons `<body>` :

```
<h1>Play !</h1>
  <div>
    <div id="container" class="masque" >
      <img id="contenu" class="sprite"/>
  </div>
</div>
```

  - Créez une balise `<script>` dans l'en-tête (entre les balises `<head>`) du document `HTML`. Cette balise contient le code suivant :

```
window.onload = function(){
    window.onkeydown = function(event){
      var code = event.keyCode;
        switch(code){
          case 37:
            // Instructions.
            alert("gauche");
            break;
          case 38:
            // Instructions.
            alert("haut");
            break;
          case 39:
            // Instructions.
            alert("droite");
            break;
          case 40:
            // Instructions.
            alert("bas");
            break;
        };
    };
};
```

    - Tout votre code devra être écrit entre les deux chevrons de cette balise.
- Présentation :
    - Lorsque le navigateur Internet exécute la méthode correspondant à un évènement, il lui transmet en paramètre un objet d'événement, cet objet d'événement contient des informations relatives à l'évènement qui est survenu. Les objets d'évènements sont documentés sur le MDN ici.
- Enoncé :
    - Chargez votre document Web contenant les éléments de code décrits plus haut. Que fait le code JavaScript fournit plus haut ? Au chargement du document, lorsqu'on presse une touche, on récupère le code correspondant à la touche. Selon ce code, on affiche une boite de dialogue.
        - Essayez ce code.
    - Votre document Web comporte une balise `img` dans une `div`. La `div` sert de *masque* pour un *sprite* qui sera chargé à l'aide de la balise `img`. Un *sprite* est une image qui contient les décompositions successives d'un mouvement.
    - Cherchez un *sprite* sur Google.
        - Chargez le *sprite* dans votre document Web en utilisant la balise `img` prévue à cet effet.
    - Paramétrez les styles de la `div` *masque* pour que une seule décomposition de mouvement du *sprite* soit visible (utilisez la propriété de style `overflow:hidden` sur le *masque*).
        - La `div` doit être en position `absolute`.
        - L'`img` dans la `div` doit être en position `absolute`.
    - En utilisant comme base le code fourni, lorsque appuyez sur la touche du clavier qui correspond à une direction :
        - Déclenchez du code JavaScript (dans la zone avec le commentaire `// Instructions.`) qui va déplacer de `2px` dans cette direction la `div` *masque*.
    - En utilisant comme base le code fourni, lorsque je presse la même direction :
        - Déclenchez du code Javascript (dans la zone avec le commentaire `// Instructions.`) qui va déplacer l'`img` dans le masque dans la direction

opposée au déplacement précédent pour afficher la décomposition suivante du mouvement.
    - Reproduire cela pour les directions restantes.
    - Trouvez le code numérique qui correspond à la touche "*entrée*". Dans le cas où l'utilisateur appuie sur la touche "*entrée*" :
        - Affichez un « mouvement spécial » de votre *sprite* (coup d'épée, explosion, etc ... ou autre). L'événement contraire de `onkeydown` est `onkeyup`.
        - Inspirez-vous du code fourni pour faire en sorte que lorsque l'utilisateur relâche la touche entrée, le *sprite* revienne à son état initial.

EXERCICE 17/23 : APPRENDRE A CREER UNE NOUVELLE « BRANCHE » DU DOM – PARTIE 1

- Préparation :
    - Reprenez le même modèle de document Web que pour l'exercice 6 (1 titre et 3 sections).
- Présentation :
    - La méthode `.createElement()` du sous objet `document` de l'objet `window` prend en argument le nom d'un élément `HTML` :
        - Exemple:
          ```
          window.document.createElement("div ");
          ```
    - Cette méthode créé et retourne un objet correspondant à un élément `HTML` de ce type. Cette méthode est documentée sur le MDN ici.
    - La méthode `.appendChild()` des objets correspondant à des éléments `HTML` du document prend en argument un objet correspondant à un élément `HTML`. Elle ajoute cet objet à l'arborescence d'objets du DOM correspondant au document affiché sous l'objet à partir duquel elle a été exécutée. Cette méthode est documentée sur le MDN ici.
- Enoncé :
    - AU CLIC sur le titre principal, utilisez la méthode `.createElement()` pour créer un objet correspondant à une balise image. Renseignez les propriétés `.title`, `.alt` et `.src` de cet objet. Utilisez la méthode `.appendChild()` pour ajouter cette balise image dans le `<span>` 2.2.

EXERCICE 18/23 : APPRENDRE A CREER UNE NOUVELLE « BRANCHE » DU DOM – PARTIE 2

- Préparation :
    - Idem.
- Présentation :
    - La méthode `.createTextNode()` du sous objet `document` de l'objet `window` prend en argument une chaîne de caractère :
        - Exemple :
          ```
          window.document.createTextNode("Ceci est un texte") ;
          ```
    - Cette méthode créé et retourne un objet correspondant à un objet de type `text` (selon la spécification du DOM). Cette méthode est documentée sur le MDN ici.
- Enoncé :
    - AU CLIC sur le premier point, utilisez la méthode `.createElement()` pour créer un objet correspondant à une balise `<a>`.
    - Puis utilisez la méthode `.createTextNode()` pour créer un objet de texte.
    - Puis utiliser la méthode `.appendChild()` pour « attacher » l'objet de texte à l'objet correspondant à la balise `<a>` que vous avez créé.

- Puis utilisez la méthode `.appendChild()` pour ajouter cette « branche » à l'arborescence d'objets du DOM correspondant au document `HTML` au niveau du `<span>` 3.2.

EXERCICE 19/23 : APPLIQUER DE BONNES PRATIQUES POUR GERER LES EVENEMENTS

- Préparation :
  - Idem.
- Présentation :
  - La méthode `.addEventListener()` est disponible sur tous les objets du DOM. Elle prend en argument 2 à 3 paramètres (on s'intéresse aux 2 premiers) :
    - Le type d'évènement (`"click"`, `"load"`, `"mouseover"`, `"mouseout"`, ...);
    - Une fonction à déclencher en réaction à cet évènement.
  - Elle présente l'avantage de pouvoir associer le déclenchement de plusieurs fonctions à un évènement survenu sur le navigateur Internet ou sur un élément `HTML`. Par exemple :

```
window.addEventListener("load", function(){
    alert("chargé");
});
```

  - Cette méthode est documentée sur le MDN ici.
- Enoncé :
  - Dans les exercices précédents vous avez associé des déclenchements de fonctions à des événements soit dans le code `HTML` soit dans le script sous la forme de méthodes copiées directement dans la propriété correspondant à l'attribut d'évènement.
  - Modifiez tous les programmes que vous avez réalisé jusqu'à présent pour utiliser la méthode `.addEventListener()`.

A partir de maintenant vous n'avez PLUS LE DROIT d'utiliser des attributs d'événements dans votre code HTML ou de copier directement des fonctions dans les propriétés du DOM correspondant aux différents événements. Toutes la gestion des évènements sera faite avec la méthode `.addEventListener()`.

EXERCICE 20/23 : RESOUDRE LES PROBLEMES DE COMPATIBILITE LIES A LA GESTION DES EVENEMENTS – PARTIE 1

- Préparation :
  - Idem
- Présentation :
  - L'objectif de cet exercice est de créer une `div` qui bouge en fonction du `scroll` de la souris. Vous devez impérativement réaliser l'exercice sur **FireFox** pour commencer.
- Enoncé :
  - Créez une `<div>` qui aura pour identifiant ascenseur. Dotez cette `<div>` de styles CSS de telle sorte qu'elle soit visible par l'utilisateur (un carré rouge par exemple). Cette `<div>` doit être positionnée en `absolute`.
  - Lorsqu'on confie une fonction au gestionnaire d'évènement du navigateur Internet, celui-ci l'exécute toujours en lui transmettant en paramètre un objet contenant les caractéristiques de l'évènement survenu. C'est pourquoi on pouvait écrire :

```
window.onload = function(unObjetFourni){
```

```
        unObjetFourni; // Ceci est la variable contenant l'objet fourni
        par le gestionnaire d'évènement à la fonction déclenchée par le
        navigateur lorsqu'il a fini de charger le document en mémoire.
};
```

- Ou encore et **de préférence** :

```
window.addEventListener("load", function(unObjetFourni){
        unObjetFourni;
}, false);
```

- Au chargement du document, attachez une fonction à l'évènement `DOMMouseScroll` en utilisant la méthode `.addEventListener()`. Cette fonction doit déclencher l'affichage d'une boite de dialogue avec le texte : `"This is a magic scroll"`.
- Modifiez cette fonction pour afficher dans la console la propriété `.detail` contenue dans l'objet d'évènement fourni à chaque fois qu'on « scroll ». Cette propriété contient une valeur qui exprime la direction vers laquelle se décale la molette.
- Modifiez cette fonction pour la `<div>` se décale en fonction de la valeur de la propriété `.detail`.
- Compatibilité Inter Navigateur :
  - L'évènement concernant le « scroll » de souris s'appelle "`DOMMouseScroll`" sur FireFox.
  - L'évènement concernant le « scroll » de souris s'appelle "`mousewheel`" sur IE11, Chrome, Safari, Opera.
  - La méthode `.addEventListener()` n'existe pas sur IE 6/7/8, il faut utiliser la méthode `.attachEvent()` et l'évènement concernant le « scroll » de souris s'appelle "`onmousewheel`" sur ces navigateurs.
  - L'information concernant le décalage de la molette qu'on trouve dans l'objet fourni par le gestionnaire d'évènement sur IE s'appelle `.wheelDelta` (la propriété `.detail` n'existe pas).
- Tenez compte des informations précédentes pour modifier votre script de telle sorte qu'il fonctionne sur IE 8,9,10,11; Edge, FF, Chrome, ...
- Astuce :
  - Vous pouvez tester l'existence d'une propriété effectuant une comparaison avec `undefined`.
  - Vous pouvez utiliser les « *modes de compatibilité* » dans IE11 (dans les outils de développement) pour effectuer des tests sur d'anciennes versions de IE.

EXERCICE 21/23 : RESOUDRE LES PROBLEMES DE COMPATIBILITE LIES A LA GESTION DES EVENEMENTS – PARTIE 2

- Préparation :
  - Idem.
- Présentation :
  - L'objectif de cet exercice est de créer une `<div>` **qui suit le pointeur** de la souris.
- Enoncé :
  - Créez une `<div>` qui aura pour identifiant *lechat*. Dotez cette `<div>` de styles CSS de telle sorte qu'elle soit visible par l'utilisateur (avec en fond une photo de petit chat par exemple). Cette `<div>` doit être positionnée en `absolute`.
  - Consultez la MDN ici pour savoir quelles sont les propriétés disponibles dans l'objet d'évènement fourni par le navigateur Internet lorsque vous déplacez la souris. Parmi ces propriétés, certaines contiennent des informations relatives aux coordonnées de

la souris. Utilisez les notions vues dans l'exercice précédent pour créer un code compatible IE8, 9, 10, 11; Edge, FF, Chrome, ... qui donne vie au chat.

EXERCICE 22/23 : EMPECHER LE COMPORTEMENT PAR DEFAUT DU NAVIGATEUR DE SE PRODUIRE

- 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.
  - Créez un formulaire avec un **champ de saisie** et un **bouton** `submit`.
    - La valeur de l'attribut `action` du formulaire sera : http://www.google.com
    - La valeur de l'attribut `method` sera : `GET`
- Présentation :
  - Si, dans un *gestionnaire d'évènement* (fonction/méthode associée à l'événement), j'exécute la méthode `.preventDefault()` de l'*objet d'évènement*, le navigateur Internet ne poursuit pas le comportement par défaut associé à l'événement.
  - La méthode `.preventDefault()` est documentée sur le MDN ici.
- Enoncé :
  - 1ère Partie : A la soumission du formulaire (vous devez utiliser l'évènement `submit` spécifique au éléments `HTML` de type `form`), afficher dans une boite de dialogue la valeur saisie dans le champ de saisie du formulaire. L'attribut `HTML` correspondant à la valeur du champ et la propriété du DOM correspondant à la valeur du champ ont le même nom.
  - 2ème Partie : A la soumission du formulaire, SI la valeur du champ est vide, je crée une élément `HTML` de type `p` avec la `class` CSS *erreur* contenant un message d'erreur, je l'ajoute dans le document en dessous du formulaire et j'empêche le formulaire d'être envoyé par le navigateur Internet.

EXERCICE 23/23 : APPRENDRE A UTILISER L'API DE L'OBJET CANVAS POUR CREER DESSINS ET ANIMATIONS

- 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.
  - Créez une balise `<canvas>` dans ce document :
    - Vous devez impérativement renseigner les attributs `width` et `height` de la balise. Attention à ne pas confondre *attributs* (sur la balise) et *propriétés de style* (valeurs de la propriété `style` ou styles `CSS`).
- Présentation :
  - L'objectif de cet exercice est d'apprendre à dessiner dans une balise `<canvas>`.
  - L'objet du DOM correspondant à la balise `<canvas>` dispose d'une méthode `.getContext()` qui vous permet de récupérer un autre objet adapté au dessin 2d dans ce contexte. Cet objet est documenté dans le MDN ici.
- Enoncé :
  - 1ère Partie : Créez l'algorithme suivant : Quand on clique sur un paragraphe de texte, on déclenche un script qui récupère une référence à l'objet `<canvas>` (en utilisant la méthode `.getElementById()` par exemple). Vérifier à l'aide de la méthode

`.dir()` de la console que vous arrivez à récupérer un objet de type `HTMLCanvasElement`.

- 2ème Partie : Créez l'algorithme suivant : Quand on clique sur un paragraphe de texte, on déclenche un script qui récupère une référence à l'objet `<canvas>` (en utilisant la méthode `.getElementById()` par exemple) et qui récupère le contexte de dessin 2d pour dessiner un carré dans le canvas.
- 3ème Partie : Créez l'algorithme suivant : Quand on clique sur un paragraphe de texte, on déclenche un script qui récupère une référence à l'objet `<canvas>` (en utilisant la méthode `.getElementById()` par exemple) :
  - puis qui récupère le contexte de dessin 2d;
  - puis efface de précédents dessins du contexte de dessin 2d;
  - puis dessine un carré avec un décalage de 2 pixels par rapport à la dernière fois qu'on a cliqué.
- 4ème Partie : Créez un *dessin animé* à l'aide de la méthode `window.setInterval()`, puis avec la méthode `window.setTimeout()` et enfin avec la méthode `window.requestAnimationFrame()`.

e. Comprendre certaines techniques avancées

EXERCICE 1/4 : CHAINAGE D'EXECUTION

- 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 :
  - En JavaScript, on peut, en utilisant certaines *ASTUCES* de programmation chaîner l'exécution de méthodes d'un objet. En d'autres termes, exécuter plusieurs méthodes d'un objet à la suite et en une seule ligne. Pour faire cela, on crée des méthodes qui retournent une référence à l'objet à partir duquel elles ont été exécutées.
- Enoncé :
  - Créer un objet littéral qui contient 3 méthodes.
  - Comment obtient-on, dans une méthode, une référence à l'objet contenant la méthode ? Retournez cette référence à partir de chacune des méthodes.
  - Exécutez les 3 méthodes sur UNE SEULE ligne.
  - Créez une fonction usine. Une fonction usine est une fonction qui retourne l'objet littéral lorsqu'elle est exécutée.
  - Exécutez la fonction usine et les 3 méthodes sur UNE SEULE ligne.
  - Renommez votre fonction usine en utilisant le caractère `$` et exécutez la fonction usine et les 3 méthodes sur une seule ligne.
  - Vous venez de comprendre un mécanisme de base utilisé dans des librairies comme `jQuery` !

EXERCICE 2/4 : LE PRECHARGEMENT

- 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 :
  - Le préchargement consiste à précharger un script, une image, une vidéo, ou un son. Pour faciliter la compréhension, ces type de fichier seront appelés "ressources" dans les explications qui suivent. Pour précharger une ressource on procède comme suit :
    - On créé en mémoire l'objet de type élément HTML correspondant au type de ressource à précharger en utilisant la méthode `.createElement()` de l'objet `document`. Par exemple :

```
var HTMLImgElement = window.createElement("img");
```

    - On affecte, comme valeur de la propriété `.src` de l'élément `HTML`, le lien vers la ressource. Le navigateur Internet envoi immédiatement une requête `HTTP` à l'`URL` affectée pour télécharger la ressource. Par exemple :

```
HTMLImgElement.src = "http://exemple/image-de-tres-grande-taille.jpg";
```

    - On affecte un `gestionnaire d'évènement` à l'évènement `load` de l'élément `HTML`. Au chargement de la ressource, cette fonction sera donc déclenchée par le navigateur Internet. Par exemple :

```
HTMLImgElement.addEventListener("load", function(event){
   // Instructions à exécuter après le chargement de la ressource
});
```

    - Lorsque la ressource est chargée, on déclenche un programme qui va positionner la ressource dans le DOM du document Web affiché.
    - Additionnellement, on peut imaginer que, initialement, l'élément `HTML` dans lequel la ressource sera positionnée contient une image au format `.GIF` animée. Il s'agit d'une une animation d'attente qui sera masquée après l'insertion de la ressource.

- Enoncé :
  - Implémentez le préchargement d'un image de très grande taille (+ de 10 Mo). Pensez à désactiver le cache de votre navigateur Internet ou le vider entre chaque essai pour que l'exercice soit pertinent.

EXERCICE 3/4 : GESTION ASYNCHRONE DES MODULES

- 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 :
  - Nous allons traiter ici d'un mécanisme connu sous le nom d'**AMD** : **A**synchronous **M**odule **D**éfinition. Sous ce nom *barbare*, ce cache l'idée suivante. Il s'agit de charger du code et de n'utiliser ses variables, fonctions ou objets qu'en cas de besoin. Cela permet de ne pas surcharger les document Web de ressources inutiles.
- Enoncé :
  - 1ère Partie : Créez un fichier `.js` qui contient une déclaration d'objet littéral représentant un *fantôme*. Cette objet à 2 caractéristiques :
    - Son nom qui est une chaîne de caractère ;

- Son râle qui est une action qui affiche une boîte de dialogue avec un cri horrible !

- NE CHARGEZ PAS ce fichier dans le document Web. Le fantôme n'existe pas ici.
- 2ème Partie : AU CLICK sur un élément HTML du document Web, créez dans le DOM un élément `HTML` correspondant à une balise script avec pour fichier source le fichier `.js` que vous avez créé.
- Vous devez donner la valeur `true` à la propriété `.async` de l'objet du DOM représentant la balise script.
- AU CHARGEMENT de votre balise script (évènement `load` de la balise script), exécutez une fonction que vous avez créé. Cette fonction exécute la méthode qui affiche le râle du fantôme.
- L'élément du DOM correspondant à une balise script est documenté ici sur le MDN.
- Lorsque cela fonctionne, on peut dire que vous avez réussi à charger votre script de façon *asynchrone* (c'est-à-dire de façon non bloquante) et à la *demande* (c'est-à-dire suite à un événement).
- 3ème Partie : Un peu de factorisation. Créez une fonction qui prend en argument le nom d'un fichier (sans le `.js`) ET une fonction :
  - La fonction à créer charge le fichier et exécute la fonction fournie.
  - La fonction fournie est appelée dans le jargon des développeur un *callback* (littéralement, *fonction de rappel* ; une fonction qui sera appelée plus tard).
  - La fonction à créer est généralement appelée « `require` » ou « `define` ».
- Lorsque cela fonctionne, on peut dire que vous avez réussi créer une fonction pour AMD ! Vous pouvez utiliser la librairie RequireJS pour bénéficier d'un boîte à outil pour les AMD qui prend en compte la compatibilité inter navigateurs, un ensemble d'optimisation et de bonnes pratiques pour la gestion des AMD.

EXERCICE 4/4 : GESTION DES FONCTIONNALITES NATIVES DES NAVIGATEURS

- 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.
  - Ce document `HTML` doit contenir plusieurs paragraphes et chaque paragraphe doit avoir un titre.
  - Ce document `HTML` doit contenir suffisamment de lignes pour que son affichage sur votre navigateur Internet fasse apparaître l'ascenseur sur la droite :
    - Vous pouvez utiliser le générateur de Lorem Ipsum ici.
- Présentation :
  - Par « fonctionnalités natives des navigateurs » on entend les fonctions : *Précédent*, *Suivant* et *Ajouter au favoris*.
  - Lorsqu'un ou plusieurs gestionnaires d'évènement qui modifient le DOM sont déclenchés par le navigateur Internet, l'affichage est modifié :
    - On appellera l'affichage initial du document `HTML` : « **état A** ».
    - On appellera l'affichage initial du document `HTML` : « **état B** ».
  - On remarque alors que si on utilise la fonctionnalité :
    - *Précédent* à partir de l'**état B** : on ne retrouve pas l'**état A** du document `HTML` mais le document `HTML` affiché précédemment.
    - *Suivant* à partir de l'**état A** : on ne retrouve pas l'**état B** du document `HTML` mais le document `HTML` affiché ensuite.

- *Ajouter aux favoris* à partir de l'**état B** : on ne retrouve pas l'**état B** lorsqu'on accède au favori enregistré.

- Comment peut-on gérer les fonctionnalités natives des navigateurs Internet en JavaScript ?
  - On peut utiliser `window.history` disponible sur les navigateurs Internet qui supportent `HTML5` et documenté ici sur le MDN.
  - On peut utiliser du JavaScript en conjonction avec le comportement natif du navigateur Internet concernant les *ancres* que le navigateur prenne en charge le `HTML5` ou pas.
- Nous allons nous intéresser à la techniques impliquant l'utilisation des *ancres*. Les *ancres* sont un mécanisme qui permet d'accéder à un emplacement d'un document `HTML` en utilisant un lien (`<a>`) sur lequel on a indiqué le nom (attribut `name`) ou l'identifiant (attribut `id`) d'une balise précédé d'un symbole `#`. Par exemple (dans un document `HTML` de n lignes) :

```
Ligne n : <a href="#/plus-bas ">Descendre plus bas</a>
…
Ligne n + 100 : <div id="/plus-bas ">Ici nous sommes plus bas</div>
```

- L'exercice suivant a pour objectif de vous apprendre à gérer les fonctionnalités natives du navigateur Internet, en JavaScript, en utilisant les *ancres*.

- Enoncé :
  - 1ère Partie : Créez une liste de liens qui sont des *ancres* vers le différents titres de votre document `HTML` et testez vos ancres. Vous remarquez qu'à chaque fois que vous testez une *ancre*, l'`URL` dans la barre d'adresse du navigateur Internet change :
    - Après avoir cliqué sur une *ancre*, utilisez la fonctionnalité *Précédent* du navigateur Internet. Que remarquez-vous (voir *barre d'adresse*) ?
    - Après avoir cliqué sur une *ancre* puis sur la fonctionnalité *Précédent* du navigateur Internet, cliquez sur la fonctionnalité *Suivant* du navigateur Internet. Que remarquez-vous (voir *barre d'adresse*) ?
    - Après avoir cliqué sur une *ancre*, cliquez sur la fonctionnalité *Ajouter au favoris* du navigateur Internet. Fermez le document. Ouvrez la favori créé. Que remarquez-vous (voir *barre d'adresse*) ?
  - 2ème Partie : Ajoutez, sur chacun des liens, un *gestionnaire d'évènement* qui change la couleur du paragraphe sous le titre correspondant au lien et qui réinitialise la couleur de tous les autres paragraphes.
  - Désormais, à chaque fois que l'utilisateur clique sur un lien, l'ancre le positionne au niveau du paragraphe concerné et modifie la barre d'adresse (fonctionnalité native du navigateur) ET la couleur du paragraphe concerné change (fonctionnalité implémentée en JavaScript).
  - On remarque que si on clique sur la fonctionnalité Précédent après avoir cliqué sur une ancre, le navigateur Internet modifie l'`URL` dans la barre d'adresse et positionne l'affichage sur l'emplacement précédent MAIS le script qui correspondant à l'emplacement précédent n'est pas exécuté :
    - Pour exécuter le script correspondant à l'emplacement précédent, nous allons utilisez le changement survenu dans l'URL de la barre d'adresse.
  - 3ème Partie : Exécutez le gestionnaire d'évènement correspondant à un paragraphe en fonction de l'ancre dans l'`URL`. L'`URL` complète est dans `window.location.href`.

- Commencez par utiliser les méthodes de l'objet `String` sur `window.location.href` pour extraire le « *hash* » de l'`URL` (partie de l'`URL` contenant le symbole `#`).
- AU CHARGEMENT du document Sauvegarder ce « *hash* » dans une variable. C'est la **VALEUR INITIALE** du « *hash* » au chargement du document.
- Utilisez la méthode `.setInterval` pour déclencher toutes les 20ms la fonction qui effectue les opérations suivantes :
  - Extraire le « *hash* » de l'`URL` (partie de l'`URL` contenant le symbole `#`) et le sauvegarder dans une variable. C'est la **VALEUR ACTUELLE** du « *hash* ».
  - SI la **VALEUR INITIALE** du « *hash* » est différente de la **VALEUR ACTUELLE** du « *hash* » :
    - Utilisez la **VALEUR ACTUELLE** du « *hash* » pour exécuter le bon gestionnaire d'évènement (c'est-à-dire qui change la couleur du bon paragraphe).
    - Remplacez la **VALEUR INITIALE** du « *hash* » par la **VALEUR ACTUELLE**.

- Si vous avez réussi, vous devriez pouvoir changer l'état de votre document `HTML` en JavaScript et retrouver l'état *Suivant*, *Précédent* ou même à partir de vos favoris.