|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| M:\ADMIN\henallux_montgolfiere.png  Implantation IESN | C:\Users\DubyLocal\Documents\ABRacine\Coordination\Logo Henallux\IG-v1.1-R1.jpg**Technologies du web**  Section IG, 2e année — Examen — novembre 2022 | | | | |
| *Nom et prénom en MAJUSCULES* | | | | / 20 |
| **CHARLIER KILLIAN** | | | |
| *Groupe* | **C** | *Enseignant* |  |

|  |  |  |
| --- | --- | --- |
| **Consignes**   * Lisez les énoncés attentivement et respectez les consignes indiquées ! * Dans votre code, utilisez avec pertinence les outils vus au cours et les particularités du langage Javascript et respectez les « bons principes » de programmation. * Pour répondre à l’examen :  |  |  | | --- | --- | | Remplissez les cadres bleutés | Les cadres orangés sont pour la correction. |  * Complétez la déclaration sur l’honneur ; sans ça, votre questionnaire sera non valide ! |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Connaissance du vocabulaire et compréhension des concepts généraux*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT -1.5 | JUSTE SUFFISANT | SATISFAISANT +1.5 | BIEN ACQUIS +2.5 | |
| *Comprendre et savoir utiliser le vocabulaire et les concepts liés au DHTML et aux langages en général* | | | | | |
| ***Maîtrise du langage Javascript*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT -1.5 | JUSTE SUFFISANT | SATISFAISANT +1.5 | BIEN ACQUIS +2.5 | |
| *Connaître et savoir utiliser les éléments de base et les particularités du langage Javascript (hors OO et DOM)* | | | | | |
| ***Utilisation de l’orienté objet en Javascript*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT -1.5 | JUSTE SUFFISANT | SATISFAISANT +1.5 | BIEN ACQUIS +2.5 | |
| *Connaître les mécanismes de l’orienté objet en Javascript et savoir les mettre en pratique.* | | | | | |
| ***Manipulation du Document Object Model*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT -1.5 | JUSTE SUFFISANT | SATISFAISANT +1.5 | BIEN ACQUIS +2.5 | |
| *Savoir comment utiliser le DOM pour lire, modifier et dynamiser le contenu HTML.* | | | | | |
| ***Clean code*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT *pénalité* | JUSTE SUFFISANT |  |  | |
| ***Qualité algorithmique du code*** | | | | | |
| TRÈS INSUFFISANT  max 6/20 | INSUFFISANT *pénalité* | JUSTE SUFFISANT |  |  | |
| cote finale (valeur indicative) = 10 – 1.5 par INSUFFISANT, + 1.5 par SATISFAISANT, +2.5 par BIEN ACQUIS. Si au moins un TRÈS INSUFFISANT : cote maximale de 6/20. Clean code & qualité modifie la cote finale. | | | | | |
| ***Commentaires*** | | | | | |
|  | | | | | |
| **PARTIE 0 (Déclaration sur l’honneur) nécessaire pour la validité de l’examen** | | | | |

En participant à cet examen et en soumettant ma copie pour correction, moi,

*(prénom et nom)*

|  |  |
| --- | --- |
| **CHARLIER KILLIAN** | *(prénom et nom)* |

m’engage à :

* ne pas partager mon questionnaire ou mes réponses avec une autre personne[[1]](#footnote-1) ;
* ne pas aider un autre étudiant participant au même examen à répondre à son questionnaire ;
* ne pas consulter un autre questionnaire du même examen ni solliciter l’aide d’une autre personne1 pour répondre à mon questionnaire.

|  |  |
| --- | --- |
| **PARTIE 1 (Vocabulaire et concepts généraux)** |  |

1. Donnez une expression régulière permettant de capturer les éléments décrits ci-dessous.

|  |  |
| --- | --- |
| 1. Les plaques d’immatriculation de la Belgique, dont le format correspond à 1 chiffre indice dont la valeur est comprise entre 1 et 7, suivi d’un tiret optionnel puis de 3 lettres majuscules, suivies d’un tiret optionnel et de 3 chiffres (par exemple : 1-ACB-279, 5AIO745, 7-AAA-255, 2AQK745). |  |
| \d- ?[A – Z]{3}- ?\d{3} |
| 1. Un fichier texte contient une suite d’informations sur un certain nombre d’aliments pour chats ; les informations sont placées à la queue-leu-leu mais toujours structurées de la manière suivante : *type-de-nourriture (un espace)* *nom-de-la-marque (un espace) le-prix-à-virgule-avec-le-symbole-euro-ou-dollar*. Par exemple :   pate Whiskas 3.59€ croquettes Felix 5.72$ friandises Royal Canin 1.29€ croquettes Schesir 6.99€ pate Gourmet Gold 7.83$  On désire modifier le fichier pour obtenir les informations au format suivant :  Whiskas, pate (3.59€), Felix, croquettes (5.72$), Royal Canin, friandises (1.29€), Schesir, croquettes (6.99€), Gourmet Gold, pate (7.83$)  Quelle expression régulière et quelle expression de remplacement utiliser pour cette transformation ? |  |
| *Expression régulière:*  [a-z]+ [a-zA-Z]+ \d.\d{2}[$€]  *Expression de remplacement:*  [a-zA-Z]+, [a-z]+ (\d.\d{2}[$€], |

1. Supposons que les navigateurs définissent automatiquement une variable qui contient leur numéro de version (sous forme d’une chaîne de caractères) et est accessible dans les scripts Javascript. Certains navigateurs appellent cette variable browserVer alors que d’autres l’appellent versionNumber. On rencontre l’instruction suivante :   
      
    const version = browserVer || versionNumber;

|  |  |
| --- | --- |
| 1. De quel type sera le contenu de la variable version après cette instruction ? |  |
| Si l’un des deux variables, soit browserVer ou versionNumber est assigné ( soit l’une, soit l’autre pas les deux ) il prendre celle assignée |
| 1. Expliquez brièvement l’effet de cette instruction en décrivant la signification de l’opérateur ||. |  |
| Le « ou » permet ici, de prendre la valeur assignée sauf pour les valeurs falsy, il ignorera le « undefined » |

1. Vrai ou faux. Dans le cas des assertions fausses, ajoutez une courte justification ou correction.

|  |  |  |
| --- | --- | --- |
| 1. Si le prototype de l’objet obj possède un attribut valeur, alors la valeur de obj.valeur sera toujours celle de cet attribut. | *V* |  |
|  | |
| 1. Si deux objets Javascript ont le même prototype, alors on peut utiliser sur eux exactement les mêmes méthodes. | V |  |
|  | |
| 1. Tous les objets qui sont des chaînes de caractères ont comme prototype l’objet String. | F |  |
| Si c’est un object donc, obj = { … }, le prototype d’obj est Object. Si il s’agit d’une déclaration comme let js = « Javascript » dans ce cas il sera le prototype de String | |
| 1. L’instruction val = val + 0; ne change jamais la valeur de la variable val. | F |  |
| Elle change de « undefined » à « NaN » | |
| 1. Si l’objet Javascript obj possède une méthode meth, alors Object.getPrototypeOf(obj['meth']).constructor est l’objet Function. | V |  |
|  | |

|  |  |
| --- | --- |
| **PARTIE 2 (Code Javascript)** |  |

**Consignes générales** :

• Les bons principes du clean code s’appliquent !

• Le code doit être présenté correctement (indentation, structure…).

• La qualité de l’implémentation (efficacité, structure…) sera prise en compte.

• Utilisez les outils vus au cours quand c’est pertinent (déstructuration, fonctions de haut niveau…).

Les chiens d’un refuge sont représentés informatiquement par des objets reprenant entre autres leur nom, leur date de naissance, un nombre maximum de friandises prévues pour l’apprentissage et un tableau reprenant les tours de passe-passe que le chien a appris. Un tour de passe-passe est composé d’un nom, d’une date de début d’apprentissage et (potentiellement) d’une date de fin d’apprentissage ainsi qu’un nombre de friandises reçues durant l’apprentissage. Voici deux exemples :

|  |
| --- |
| {  nom: 'Rex',  dateDeNaissance: 20180507,  friandisesPrevues: 150,  tours: [  {  nom: 'assis',  debutApprentissage: 20180901,  finApprentissage: 20181007,  friandisesRecues: 24  },  {  nom: 'couché',  debutApprentissage: 20181014  }  ]  } |
| {  nom: 'Bella',  dateDeNaissance: 20221011,  friandisesPrevues: 120,  tours: [  {  nom: 'reste',  debutApprentissage: 20221109,  finApprentissage: 20221217,  friandisesRecues: 12  }  ]  } |

1. Écrivez une fonction pretPourApprentissage qui reçoit un objet correspondant à un chien (au format présenté ci-dessus) et qui renvoie un booléen indiquant si tous les tours enseignés sont bien acquis. Un tour est considéré comme acquis lorsqu’une date de fin d’apprentissage est renseignée. Utilisez la notation => et une fonction de haut niveau !

|  |  |
| --- | --- |
| function pretPourApprentissage (chien)  {  return chien.tours.every((tour) => "finApprentissage" in tour);  } |  |

1. Écrivez une fonction majTours qui reçoit un tableau de tours (décrits avec le nom du tour et la date de début et/ou de fin d’apprentissage de celui-ci) et le chien. Elle doit ajouter les tours qui sont en cours d’apprentissage et mettre à jour les tours qui sont considérés comme acquis. Le tableau des tours fourni pourra être éventuellement vide ou undefined. Il se peut qu’une erreur se glisse dans le tableau et qu’un tour acquis ne porte pas le bon nom…

Voici un exemple de tableau de tours :

|  |
| --- |
| [  { nom: 'donne la patte', debutApprentissage: 20191204 },  {  nom: 'couché',  finApprentissage: 20181105,  friandisesRecues: 21  },  {  nom: 'va chercher',  debutApprentissage: 20191117,  finApprentissage: 20191201,  friandisesRecues: 5  },  {  nom: 'danse avec les stars',  finApprentissage: 20220105,  friandisesRecues: 37  }  ] |

*Astuce :* pour récupérer l’indice d’un tour dans le tableau des tours d’un chien sur base du nom, utilisez la méthode tab.findIndex(fonction). La fonction doit être écrite avec une notation => !

|  |  |
| --- | --- |
| function majTours (tours, chien)  {  for (const tour of tours)  {  const indexTourChien = chien.tours.findIndex((tourChien) => tourChien["nom"] === tour["nom"]);    if (indexTourChien > -1)  {  chien.tours[indexTourChien].finApprentissage = tour.finApprentissage;  chien.tours[indexTourChien].friandisesRecues = tour.friandisesRecues  }  else  {  if (tour.debutApprentissage)  {  chien.tours.push(tour);  }  }  }  } |  |

1. Écrivez une fonction assezDeBonbons qui reçoit un chien comme paramètre et qui vérifie que la somme des friandises utilisées dans l’apprentissage des tours de passe-passe ne dépasse pas (inférieur ou égal à) le total de friandises prévues pour l’animal.

|  |  |
| --- | --- |
| function assezDeBonbons (chien)  {  let totalBonbons = 0;  for (const tour of chien.tours)  {  if (tour.friandisesRecues)  {  totalBonbons += tour.friandisesRecues  }  }  return totalBonbons <= chien.friandisesPrevues;  } |  |

|  |  |
| --- | --- |
| **PARTIE 3 (Javascript orienté objet)** |  |

On a prévu le code suivant pour gérer sous forme d’objets Javascript les chiens du refuge. En plus du nom et de la date de début d’apprentissage du tour de passe-passe, on précise la difficulté d’apprentissage de celui-ci vis-à-vis du caractère du chien.

|  |
| --- |
| function Chien(nom, dateDeNaissance, friandisesPrevues) {  this.nom = nom;  this.dateDeNaissance = dateDeNaissance;  this.friandisesPrevues = friandisesPrevues;  this.tours = [];  this.ajouteTour = function (nom, debutApprentissage, difficulte) {  tours.splice(0, 0, { nom, debutApprentissage, difficulte });  };  } |

1. Sur base du code ci-dessus, quel est le type de structure utilisé pour stocker les différents tours qu’un chien apprend/a appris ?

|  |  |
| --- | --- |
| Un objet / tableau associatif |  |

1. Corrigez ce code afin qu’il fonctionne et qu’il soit plus optimal.

|  |  |
| --- | --- |
| function Chien (nom, dateDeNaissance, friandisesPrevues)  {  this.nom = nom;  this.dateDeNaissance = dateDeNaissance;  this.friandisesPrevues = friandisesPrevues;  this.tours = [];  }  Chien.prototype.ajouteTour = function (nom, debutApprentissage, difficulte)  {  this.tours.push({ nom, debutApprentissage, difficulte});  } |  |

1. Ajoutez une méthode de sorte que chien.tourLePlusDifficile() renvoie le tour qui a été le plus difficile à apprendre pour le chien en question.

|  |  |
| --- | --- |
| Chien.prototype.tourLePlusDifficile = function ()  {  let tourLePlusDur;  for (const tour of this.tours)  {  if (!tourLePlusDur)  {  tourLePlusDur = tour;  }  else  {  tourLePlusDur = tourLePlusDur.difficulte < tour.difficulte ? tour : tourLePlusDur;  }  }  return tourLePlusDur;  } |  |

1. Faites en sorte qu’il soit possible à tout moment de retrouver la liste (un tableau) des noms de tous les tours de passe-passe déjà enseignés à tous chiens confondus (en Java, cela reviendrait à créer un attribut de classe à mettre à jour à chaque ajout d’un tour). Cette liste ne doit pas contenir de doublon.

*Astuce :* utilisez la méthode tab.includes(elem) qui renvoie un booléen indiquant si elem fait partie de tab.

Indiquez également quelle syntaxe utiliser pour récupérer cette liste.

|  |  |
| --- | --- |
| function Chien (nom, dateDeNaissance, friandisesPrevues)  {  this.nom = nom;  this.dateDeNaissance = dateDeNaissance;  this.friandisesPrevues = friandisesPrevues;  this.tours = [];  }  Chien.prototype.nomsTours = [];  Chien.prototype.ajouteTour = function (nom, debutApprentissage, difficulte)  {  this.tours.push({ nom, debutApprentissage, difficulte});  if (!this.nomsTours.includes(nom))  {  this.nomsTours.push(nom);  }  }  Chien.prototype.tourLePlusDifficile = function ()  {  let tourLePlusDur;  for (const tour of this.tours)  {  if (!tourLePlusDur)  {  tourLePlusDur = tour;  }  else  {  tourLePlusDur = tourLePlusDur.difficulte < tour.difficulte ? tour : tourLePlusDur;  }  }  return tourLePlusDur;  } |  |

|  |  |
| --- | --- |
| **PARTIE 4 (Manipulation du DOM)** |  |

On désire réaliser un petit carrousel de photos de chats. Le but est de montrer à l’utilisateur un ensemble de photos de chats les unes après les autres. L’utilisateur peut décider de passer à l’image suivante ou de revenir à la précédente, ainsi que d’arrêter le carrousel et de le démarrer.

La première étape est de demander combien de chats, entre 1 et 8, doivent être affichés. Ensuite, lorsque le joueur clique le bouton « Démarrer », chaque chat apparaît pendant 2.5 secondes avant de disparaître. Lorsque le dernier chat disparaît, le tout premier doit évidemment réapparaître.

La page web proposée à la page suivante contient une zone divCarrousel dans laquelle vous devez ajouter autant d’images de chats que demandées.

Les images sont ajoutées sur base d’un numéro généré aléatoirement entre 1 et 6. Une image peut être présente plusieurs fois. Les images sont fournies dans le fichier *images.rar*.

* Écrivez le code Javascript qui permet, au chargement de la page, de
* demander le nombre d’images à l’utilisateur. On considère que le nombre obtenu est valide et est donc bien un entier entre 1 et 8 inclus.
* générer la liste d’images sur base des explications ci-avant.
* Écrivez une fonction Javascript qui fait en sorte que, lorsqu’on clique sur le bouton « Démarrer », le carrousel démarre, ce qui signifie que les images sont visibles pendant 2.5 secondes les unes à la suite des autres en boucle.
* Écrivez une fonction Javascript qui fait en sorte que, lorsqu’on clique sur le bouton « Précédent », l’image de chat précédente est affichée.
* Écrivez une fonction Javascript qui fait en sorte que, lorsqu’on clique sur le bouton « Suivant », l’image de chat suivante est affichée.
* Écrivez une fonction Javascript qui fait en sorte que, lorsqu’on clique sur le bouton « Stop », le carrousel s’arrête, ce qui signifie que l’image actuellement visible le reste indéfiniment.

Conseils :

* La méthode random de la fonction constructrice Math permet de générer un nombre aléatoire.
* La propriété CSS display permet d’afficher ou non un élément en lui associant soit la valeur none, soit la valeur block.
* Pour modifier le style des images, pensez à la propriété classList et à sa méthode toggle.
* Vous pouvez définir d’autres éléments à ajouter à la page web (comme des styles CSS par exemple).

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Carrousel</title>  <style>  img {  height: 200px;  width: 300px;  }  </style>  </head>  <body>  <div id="divCarrousel"></div>  <button id="btnPrecedent">Précédent</button>  <button id="btnSuivant">Suivant</button>  <button id="btnStop">Stop</button>  <button id="btnStart">Démarrer</button>  </body>  </html> |

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Carrousel</title>  <style>  img  {  height: 200px;  width: 300px;  }  </style>  <script src = "dom.js"></script>  </head>  <body>  <div id="divCarrousel"></div>  <button id="btnPrecedent">Précédent</button>  <button id="btnSuivant">Suivant</button>  <button id="btnStop">Stop</button>  <button id="btnStart">Démarrer</button>  </body>  </html>  Partie JS  let nbImages;  let emplacementImage;  let timeoutId;  window.onload = init;  function init ()  {  emplacementImage = document.getElementById("divCarrousel")  nbImages = parseInt(prompt("Combien d'images voulez-vous ? : "));  ajouteImageCadre();  document.getElementById("btnStart").addEventListener("click", démarrer);  document.getElementById("btnSuivant").addEventListener("click", suivant);  document.getElementById("btnPrecedent").addEventListener("click", précédent);  document.getElementById("btnStop").addEventListener("click", stop);  }  function ajouteImageCadre ()  {  for (let i = 0; i < nbImages; i++)  {  const imageAffichée = document.createElement("img");  imageAffichée.classList.add("img");  imageAffichée.src = `chat${Math.floor(Math.random() \* 6) + 1}.jpg`;  imageAffichée.style.display = "none";  emplacementImage.appendChild(imageAffichée);  }  }  function démarrer ()  {  for (let i = 0; i < nbImages; i++)  {  const chat = emplacementImage.children[i];    timeoutId = setTimeout(() =>  {  console.log(timeoutId);  toggleVisibilitéImage(chat);  setTimeout(() => toggleVisibilitéImage(chat), 2500);  timeoutId--;  }, 2500 \* i);  }  setTimeout(toggleVisibilitéImage,2500 \* nbImages, emplacementImage.children[0]);  }  function chercheImageAffichée ()  {  let indexImageAffichée;  for (let i = 0; i < nbImages; i++)  {  if (emplacementImage.children[i].classList.contains("image-visible"))  {  indexImageAffichée = i;  }  }  return indexImageAffichée;  }  function suivant ()  {  const indexImageVisible = chercheImageAffichée();  if (emplacementImage.children[indexImageVisible].nextElementSibling == null)  {  toggleVisibilitéImage(emplacementImage.children[indexImageVisible]);  toggleVisibilitéImage(emplacementImage.children[0]);  }  else  {  toggleVisibilitéImage(emplacementImage.children[indexImageVisible]);  toggleVisibilitéImage(emplacementImage.children[indexImageVisible].nextElementSibling);  }  }  function précédent ()  {  const indexImageVisible = chercheImageAffichée();  if (emplacementImage.children[indexImageVisible].previousElementSibling == null)  {  toggleVisibilitéImage(emplacementImage.children[indexImageVisible]);  toggleVisibilitéImage(emplacementImage.children[nbImages - 1]);  }  else  {  toggleVisibilitéImage(emplacementImage.children[indexImageVisible]);  toggleVisibilitéImage(emplacementImage.children[indexImageVisible].previousElementSibling);  }  }  function stop ()  {  clearTimeout(timeoutId);  }  function toggleVisibilitéImage (image)  {  const visible = image.classList.contains("image-visible");  image.style.display = visible ? "none" : "inline";  image.classList.toggle("image-visible");  }  Pas eu le temps de faire le button stop, j’ai fais les deux autres avant le stop ( je n’ai pas vu le message discord ) |
|  |

1. À l’exception des enseignants en charge de l’examen en question. [↑](#footnote-ref-1)