HTML

Indique le/les erreur(s) possible(s) dans le code suivant (Une seule réponse possible):

```
1 <div>
2    <h1 id="title main-title" class="title">Hello World</h1>
3    <p id="title" class="title secondary-title">
4       This is a <bold>quiz</bold>.
5    </p>
6 <div>
```

1. Le parent (div) ne contient pas de balise fermante.
2. Le même identifiant est utilisé pour plusieurs éléments.
3. <bold></bold> n'est pas un élément HTML valide.
4. La même classe « title » a été utilisée pour plusieurs éléments.
5. L'attribut id ne doit contenir qu'une seule valeur.

- a. 2, 5
- b. 1, 2, 3, 4, 5
- c. 1, 2, 3, 5 ✔
- d. 1, 3, 5
- e. 4

Lequel/lesquels parmi les suivants est/sont des syntaxes HTML valides?

- a. <body onload="myFunction()"></body> ✔
- b.
- c. Visit My Website
- d. <button data-button-id="0" onclick="showId()">Click Me</button> ✔

À travers de quoi le Document Object Model (DOM) peut-il être modifié et manipulé une fois construit?

- a. HTML et CSS
- b. JavaScript ✔
- c. HTML et JavaScript
- d. CSS
- e. CSS et JavaScript
- f. HTML

Lequel parmi les faits suivants est vrai?

a. HTML désigne les comportements de la page, CSS permet de styler les composantes d'un site Web et Javascript permet d'organiser et placer des éléments sur le site Web.

b. Aucune de ces réponses.

c. HTML est un langage structurel qui permet l'affichage des éléments du contenu de la page Web, CSS permet de mettre en forme le contenu d'un site Web et Javascript est un langage de programmation qui permet de définir la logique et le comportement d'un site Web. ✔

d. HTML détermine le style et l'apparence des éléments sur le site Web et CSS ainsi que Javascript sont des langages de programmation permettant de gérer l'interactivité avec les différents éléments affichés par le navigateur.

Sélectionnez une réponse en analysant le bout de code et les options ci-dessous.

```
<form action="www.mypizza.com" method="POST" target="_blank">
    <label for="my-name">Name:</label>
    <input type="text" id="my-name" name="my-name" required />

    <label for="pineapple">Pineapple?</label>
    <checkbox id="pineapple" name="pineapple" />

    <label for="description">Description</label>
    <textfield id="description" name="description" placeholder="Enter optional details"></textfield>
</form>
```

1. <textfield></textfield> et <checkbox /> ne sont pas des éléments HTML valides.
2. Les contenus du formulaire seront envoyés à *www.mypizza.com* via une méthode POST.
3. L'attribut *for* de l'élément *label* doit nécessairement correspondre à l'attribut name d'un *input*.
4. La réponse obtenue suite à l'envoie du formulaire sera ouverte et affichée dans un nouvel onglet ou fenêtre.
5. Les données du formulaire sont visibles dans l'URL du client lorsque le formulaire utilise une méthode POST.
6. Le *input* ayant un identifiant «my-name» désigne un champ obligatoire.

a. 1, 2, 4, 6 ✔

b. 1, 2, 3, 4, 6

c. Toutes les réponses

d. 1, 2, 4, 5, 6

e. 2, 3, 5

Lequel parmi les faits suivants concernant la construction d'une page Web est vrai?

a. Le DOM et CSSOM forment le *Render Tree*, alors que l'étape *Paint* calcule la taille et position de chaque élément.

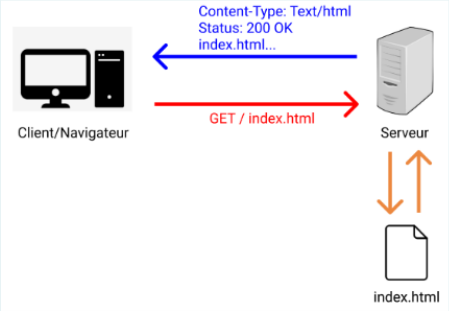
b. Le DOM est généré par l'intermédiaire de CSS alors que le HTML permet de créer le CSSOM.

c. Premièrement, le CSSOM est formé. Deuxièmement, le HTML est analysé pour créer le DOM. Troisièmement, le DOM, CSSOM et Javascript sont tous utilisés pour bâtir le *Render Tree*. Quatrièmement, le *Layout* calcule la position et la taille de chaque élément. Finalement, l'étape *Paint* affiche les éléments sur l'écran.

d. Le DOM et CSSOM forment le Render Tree. Celui-ci génère ensuite le *Layout* de chaque élément visible qui sert finalement à une entrée pour la procédure de *Paint* qui s'occupe de la composition des pixels sur l'écran. ✔

e. Le *Render Tree* ainsi que le *Layout* sont des étapes qui permettent de combiner le DOM et le CSSOM.

Décrivez le scénario correct en se basant sur l'image ci-dessous.



a. Le serveur envoie une requête HTTP /GET et le navigateur répond avec un code HTTP ainsi que la page index.html.

b. Le client envoie la page index.html au serveur à travers la méthode HTTP /GET. Le serveur s'en occupe pour sauvegarder cette page et répond avec un code HTTP 200 OK.

c. Le client fait une requête HTTP en précisant une méthode HTTP GET /index.html au serveur. Ce dernier reçoit et traite la requête et répond au client avec les informations demandées (index.html) et un code HTTP 200 OK. ✔

d. Le serveur demande un fichier avec le contenu text/html nommée index.html au client qui renvoie cette page en question au serveur.

CSS :

Quel type de modèle de mise en page inclut *margin*, *border*, *padding* et le contenu de l'élément?

a. Modèle de boîtes flexibles (Flexbox Layout)

b. Modèle de boîtes positionnées ✔

c. Modèle de grille (Grid Layout)

d. Modèle de table

Quel(s) type(s) de modèle de mise en page requiert/requirent un élément parent (un conteneur)?

a. Modèle de boîtes positionnées

b. Toutes les réponses

c. Modèle de boîtes flexibles (Flexbox Layout) ✔

d. Modèle de grille (Grid Layout) ✔

Selon le code HTML ci-dessous, quelles propriétés et/ou pseudo-éléments CSS parmi les options suivantes (1 à 10) ont possiblement été appliquées pour obtenir l'image désirée de la Figure 1?

```html
<div>
   <p>Lorem ipsum dolor sit amet.</p>
   <p>Lorem ipsum dolor sit amet.</p>
   <p>Lorem ipsum dolor sit amet.</p>
   <p>Lorem ipsum dolor sit amet.</p>
</div>
```

Code HTML

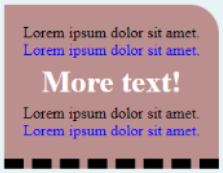


Figure 1: Résultat voulu

1. p(2)::after { content: "More text!"; }
2. border-bottom: 10px dashed #000;
3. border-top-right-radius: 30px;
4. p::third-child { content: "More text!"; }
5. p:nth-child(even) { color: blue; }
6. p::before:nth-child(3) { content: "More text!"; }
7. border[down]: 10px solid black;
8. p:nth-child(2, 4) { color: blue; }
9. border-radius: 0 30px 0 0;
10. p:nth-child(3)::before { content: "More text!"; }

- a. 3, 4, 5, 6, 8
- b. 2, 4, 7, 9
- c. 1, 3, 6, 9
- d. 2, 3, 5, 6, 7
- e. 1, 2, 3, 5
- f. 2, 5, 9, 10 ✔

Quelle est la syntaxe correcte pour sélectionner tous les paragraphes ayant les classes "title" et "info" qui se retrouvent après l'élément *span*?

```html
<p class="title info">First text</p>
<span class="title info">This is a span</span>
<p class="title info">Second text</p>
<p>Third text</p>
<p class="title info">Forth text</p>
```

- a. span > p .title.info { }
- b. span p .title .info { }
- c. span + p .title .info { }
- d. span p.title.info { }
- e. span:not(p.title.info) { }
- f. .title, .info { }
- g. span ~ p.title.info { } ✔

De quelle façon le CSS d'une page peut être référé?

- a. Feuille de style interne (Internal Style Sheet)
- b. Feuille de style externe (External Style Sheet)
- c. Déclaration CSS en ligne (Inline Styling)
- d. Les trois options sont des réponses valides. ✔

Que représente le bout de code suivant?

```
grid-template-rows: 100px 200px 2fr 1fr;
```

- a. La 3ème et 4ème ligne occupent l'espace restant dans le grid. Cependant, la 3ème ligne est 2 fois plus grande que la 4ème ligne. ✔
- b. La 3ème et 4ème ligne occupent l'espace restant dans le grid. Cependant, la 3ème ligne est 2 fois plus petite que la 4ème ligne.
- c. La 1ère et 3ème ligne sont associées: la 3ème ligne est égale à 200px puisqu'elle possède une fraction de 2fr (100px multiplié par 2). La 2ème et 4ème ligne sont associées: la 4ème ligne est égale à 200px puisqu'elle possède une fraction de 1fr (200px multiplié par 1fr).
- d. Les deux premières valeurs correspondent aux lignes alors que les deux dernières valeurs correspondent aux colonnes.

Quelle est la forme HTML correcte pour inclure un fichier CSS externe?

- a. <link type="text/css" href="style" rel="stylesheet" />
- b. <stylesheet>"style.css"></stylesheet>
- c. <link rel="stylesheet" type="text/css" href="style.css"> ✔
- d. <style src="style.css">

En prenant en compte la notion de spécificité et requête média, quelles seront les couleurs respectives du contenu «Hello World» pour:

1) un écran de largeur 500px

2) un écran de largeur 800px

3) un écran de largeur 1250px

```html
<div class="box">
    <p id="title" class="message">Hello World</p>
</div>
```

```css
p#title.message {
  color: red;
}

@media screen and (min-width: 1000px) {
  p#title.message {
    color: green;
  }
}

@media all and (min-width: 800px) {
  .box p#title.message {
    color: black;
  }
}

@media all and (max-width: 600px) {
  .box p {
    color: blueviolet;
  }
}
```

- a. blueviolet, black, red
- b. blueviolet, red, green
- c. red, blueviolet, green
- d. red, red, black
- e. red, red, red
- f. red, blueviolet, black
- g. red, red, green
- h. red, black, black ✔
- i. blueviolet, black, green

JAVASCRIPT

Qu'affichera le code ci-dessous étant donné les fonctions imbriquées sont tous appelées et les paramètres *c* et *d* possèdent la valeur 5 et 8 respectivement?

```javascript
const a = [8, 5, 10, -3, 7];

function addNumbers() {
  const b = a.pop();

  // c = 5
  return function (c) {
    this.b = a.shift();

    // d = 8
    return function (d) {
      a[10] = d === b ? d / b : a.length;
      console.log("The sum is", a[10] + this.b + c + d);
    };
  };
}

// addNumbers();
```

code JavaScript

- a. Le programme lancera une erreur, car a[10] n'est pas un index valide puisque le tableau ne contient que 5 éléments.
- b. Le programme lancera une erreur, car certaines lignes dans le code ne sont pas valides.
- c. The sum is 25
- d. The sum is 22
- e. The sum is 31
- f. The sum is 23
- g. The sum is 30
- h. The sum is 24 ✔
- i. Il n'est pas possible de retourner des fonctions plusieurs fois.

Qu'affichera le programme ci-dessous en analysant le code Javascript suivant?

```javascript
const person = {
  age: "20",
  name: "Jean",
};

const age = person["age"];
person["age"] = age + 1;

console.log("Jean's age is", person.age);
```

code Javascript

- a. Le programme lancera une erreur « Infinity » ou « NaN (Not a Number) », car la propriété *age* de l'objet *person* est un *string*, alors que le programme tente de faire une opération arithmétique sur celle-ci et de la convertir en *number*.
- b. Jean's age is 21 puisque l'opérateur + force le type number au résultat.
- c. Jean's age is 201 puisque le type de la variable est un string. ✔
- d. L'objet *person* est défini comme *const*, ce qui signifie que l'objet *person* ainsi que ses propriétés (*age*, *name*) ne peuvent pas être réassigner ou modifier. Donc, le programme lancera une erreur.
- e. Le programme lance une erreur puisque « age » est déclaré plus qu'une fois.
- f. Jean's age is 201 puisque la variable *age* est défini comme *const* à la ligne 6, ce qui signifie que ce n'est pas possible de modifier celle-ci à la ligne 7, mais seulement l'additionner à une autre variable.

Que sera affiché par le programme ci-dessous?

```
let width = Math.round(2.49);
const height = 11.0 + 1;

width += 1;

console.log('height / width');
```

- a. NaN
- b. 4.0
- c. height / width ✔
- d. 4
- e. 3
- f. Il n'est pas possible de réassigner une valeur à une variable déclarée comme *let*.

Écrivez une fonction nommée *sumTwoArrays* qui prend 2 tableaux en paramètres et qui calcule la somme de ces deux tableaux. Retournez la somme obtenue.

Par exemple:

Test	Résultat
`console.log(sumTwoArrays([2, 4, 5], [0, 1]));`	12

Réponse : (régime de pénalités : 10, 20, ... %)

```
function sumTwoArrays(table1, table2){
    let sum = 0;

    for(let i = 0; i < table1.length; i++){
        sum += table1[i];
    }

    for(let i = 0; i < table2.length; i++){
        sum += table2[i];
    }

    return sum;
}
```

Que seront affichés par le programme ci-dessous?

```
function updateInfo(yearOfBirth, type, pet) {
  yearOfBirth = 2005;
  type = "Cat";
  pet.name = "Kitty";
}

let yearOfBirth = 2002;
let pet = {
  type: "Dog",
  name: "Enzo",
};

updateInfo(yearOfBirth, pet.type, pet);

console.table({ yearOfBirth, pet });
```

- a. ✔

(index)	type	name	Values
yearOfBirth			2002
pet	'Dog'	'Kitty'	

Quel est l'utilité de l'attribut **defer** dans le script suivant?

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

- a. Aucune des réponses.
- b. Le script sera executé de manière asynchrone avec le reste de la page, c'est-à-dire, le script sera executé pendant que la page est en train d'interprêter le DOM.
- c. Le script sera executé après que le chargement du DOM soit entièrement complété. ✔
- d. Cet attribut permet de récupérer le script qui est executé immédiatement sans bloquer le DOM.

JAVASCRIPT 2 :

Considérez la variable "button" qui contient une référence vers un bouton HTML et qui n'a aucun gestionnaire d'événement attaché au préalable.

Après l'exécution du code suivant et un "click" sur le bouton, quelle(s) alerte(s) sera(sont) affichée(s) à l'écran et pourquoi ?

```
1 button.addEventListener("click", () => alert("1"));
2
3 button.removeEventListener("click", () => alert("1"));
4
5 button.onclick = () => alert(2);
```

- a. Alerte : "2" puisque "removeEventListener" a retiré le premier gestionnaire d'événements
- b. Alertes : "1" et "2" puisque "removeEventListener" n'a pas retiré le premier gestionnaire d'événements ✔
- c. Alertes : "1" et "2" puisque "removeEventListener" n'est pas une méthode valide
- d. Alerte : aucune puisque ce code n'est valide.
- e. Alerte : "1" puisque "onclick" n'est pas un attribut valide d'un HTMLElementButton
- f. Alerte : "1" puisqu'on ne peut pas attacher plusieurs gestionnaires du même événement sur le même élément

Choisissez seulement le **premier** enfant directe (élément ***div « First »***) dont le parent est « ***.container*** ». Ce dernier est imbriqué dans un autre élément ayant la classe « ***.main*** ».

```
1 <div class="main">
2     <div class="container">
3         <div>First</div>  <---
4         <div>Second</div>
5         <div>Third</div>
6     </div>
7 </div>
```

Illustration du DOM

- a. document.querySelectorAll(".main .container > div")
- b. document.getElementsByClassName(".main .container > div")[0]
- c. document.getElementsByTagName("div")[0]
- d. document.querySelector(".main .container div") ✔

Lequel parmi les points suivants ne fait pas partie des propriétés d'un système de modularisation?

- a. Importation des éléments d'un module à dans une autre source de code
- b. Utilisation d'un module pour définir le code
- c. Utilisation des fichiers externes
- d. Exportation des éléments d'un module
- e. Aucune des réponses ✔

Quel(s) serait/seraient la/les façon(s) d'insérer l'élément à l'endroit indiqué soit à la 2e ligne à la figure 1?

```
1 <div>First</div>
2 <!-- Add Second <HTMLDivElement> here-->
3 <div>Third</div>
```

Figure 1: Représentation du DOM *(fichier HTML)*

```
1 const newlyCreatedDiv = document.createElement("div");
2 newlyCreatedDiv.setAttribute("id", "second");
3 newlyCreatedDiv.textContent = "Second";
4
5 const firstDivElement = document.getElementsByTagName("div")[0];
6 const thirdDivElement = document.getElementsByTagName("div")[1];
```

Figure 2: Code additionnel pour la création de l'élément HTMLDivElement *(fichier JavaScript)*

- [ ] a. firstDivElement.appendChild(newlyCreatedDiv);
- [x] b. thirdDivElement.parentNode.insertBefore(newlyCreatedDiv, thirdDivElement);✔
- [x] c. firstDivElement.insertAdjacentElement("afterend", newlyCreatedDiv);✔
- [ ] d. thirdDivElement.appendChild(newlyCreatedDiv);
- [ ] e. firstDivElement.parentNode.insertBefore(thirdDivElement, newlyCreatedDiv);

Comment pouvons-nous ajouter le texte « Hello World » à un élément HTML vide nommé *emptyNode*?

- ○ a. emptyNode.innerText = "Hello World"
- ○ b. emptyNode.innerHTML = "Hello World"
- ○ c. emptyNode.appendChild(document.createTextNode("Hello World"));
- ○ d. a) et b)
- ◉ e. Toutes les réponses✔

Quelle est la méthode ou le moyen pour supprimer une clé à partir d'un *Storage*?

- ◉ a. removeItem(*key*)✔
- ○ b. deleteItem(*key*)
- ○ c. delete localStorage(*key*) ou delete sessionStorage(*key*)
- ○ d. removeItem(*key*, *value*)
- ○ e. removeKey(*key*)

DOM :

Comment pouvons-nous ajouter le texte « Hello World » à un élément HTML vide nommé *emptyNode*?

- ○ a. emptyNode.innerText = "Hello World"
- ○ b. emptyNode.innerHTML = "Hello World"
- ○ c. emptyNode.appendChild(document.createTextNode("Hello World"));
- ○ d. a) et b)
- ◉ e. Toutes les réponses✔

Quelle est la méthode ou le moyen pour supprimer une clé à partir d'un *Storage*?

- a. removeKey(*key*)
- b. delete localStorage(*key*) ou delete sessionStorage(*key*)
- c. deleteItem(*key*)
- d. removeItem(*key*) ✔
- e. removeItem(*key*, *value*)

Quel(s) serait/seraient la/les façon(s) d'insérer l'élément à l'endroit indiqué soit à la 2e ligne à la figure 1?

```
1 <div>First</div>
2 <!-- Add Second <HTMLDivElement> here-->
3 <div>Third</div>
```

Figure 1: Représentation du DOM *(fichier HTML)*

```
1 const newlyCreatedDiv = document.createElement("div");
2 newlyCreatedDiv.setAttribute("id", "second");
3 newlyCreatedDiv.textContent = "Second";
4
5 const firstDivElement = document.getElementsByTagName("div")[0];
6 const thirdDivElement = document.getElementsByTagName("div")[1];
```

Figure 2: Code additionnel pour la création de l'élément HTMLDivElement *(fichier JavaScript)*

- a. firstDivElement.insertAdjacentElement("afterend", newlyCreatedDiv); ✔
- b. firstDivElement.appendChild(newlyCreatedDiv);
- c. thirdDivElement.appendChild(newlyCreatedDiv);
- d. firstDivElement.parentNode.insertBefore(thirdDivElement, newlyCreatedDiv);
- e. thirdDivElement.parentNode.insertBefore(newlyCreatedDiv, thirdDivElement); ✔

Considérez la variable "button" qui contient une référence vers un bouton HTML et qui n'a aucun gestionnaire d'événement attaché au préalable.

Après l'exécution du code suivant et un "click" sur le bouton, quelle(s) alerte(s) sera(sont) affichée(s) à l'écran et pourquoi ?

```
1 button.addEventListener("click", () => alert("1"));
2
3 button.removeEventListener("click", () => alert("1"));
4
5 button.onclick = () => alert(2);
```

- a. Alertes : "1" et "2" puisque "removeEventListener" n'est pas une méthode valide
- b. Alertes : "1" et "2" puisque "removeEventListener" n'a pas retiré le premier gestionnaire d'événements ✔
- c. Alerte : aucune puisque ce code n'est valide.
- d. Alerte : "1" puisque "onclick" n'est pas un attribut valide d'un HTMLElementButton
- e. Alerte : "1" puisqu'on ne peut pas attacher plusieurs gestionnaires du même événement sur le même élément
- f. Alerte : "2" puisque "removeEventListener" a retiré le premier gestionnaire d'événements

Lequel parmi les points suivants ne fait pas partie des propriétés d'un système de modularisation?

a. Exportation des éléments d'un module
b. Importation des éléments d'un module à dans une autre source de code
c. Utilisation d'un module pour définir le code
d. Utilisation des fichiers externes
e. Aucune des réponses ✔

Choisissez seulement le **premier** enfant directe (élément ***div*** « ***First*** ») dont le parent est « ***.container*** ». Ce dernier est imbriqué dans un autre élément ayant la classe « ***.main*** ».

```
<div class="main">
    <div class="container">
        <div>First</div>  <---
        <div>Second</div>
        <div>Third</div>
    </div>
</div>
```

Illustration du DOM

a. document.querySelectorAll(".main .container > div")
b. document.querySelector(".main .container div") ✔
c. document.getElementsByClassName(".main .container > div")[0]
d. document.getElementsByTagName("div")[0]

Validation et Verification :

Ordonnez les tests suivants selon l'ordre qu'ils doivent être exécutés.

1. Tests unitaires ✔
2. Tests d'intégration ✔
3. Test de système ✔
4. Test d'acceptation ✔

Analysez le code et les jeux de tests ci-dessous et repondez aux questions concernant la couverture des instructions?

Test1 = <{x=-1, y=1}, {return}>

Test2 = <{x=0, y=1}, {0}>

```
let x, y, sum;
if (x < 0 && x < y)
  return;
let z = 0;
while (y > x || x > y) {
  sum = 0;
  let temp = y - 1;
  while (temp > 0) {
    sum = sum + sum * temp;
    temp--;
  }
  z += sum;
  y--;
}
console.log(z);
```

1. Les lignes parcourues par Test1 sont 1, 2, 3 ✔
2. Les lignes parcourues par Test2 sont 1, 2, 4, 5, 6, 7, 8, 11, 12, 13, 14, 15 ✔
3. Un nouveau jeu de test (Test3) devra parcourir les lignes 9, 10 ✔ pour satisfaire la couverture des instructions
4. Une solution possible pour Test3 est <{x=3, y=2}, {0}> ✔

Quelle est la différence entre retester un code et les tests de régression?

- a. Retester signifie reexécuter les tests une nouvelle fois alors que les tests de regréssion permettent de détecter si une erreur inattendue s'est produite lors des changements dans notre programme. ✔
- b. Retester permet de détecter si une erreur inattendue s'est produite lors des changements dans notre programme alors que les tests de regréssion signifie reexécuter les tests une nouvelle fois.
- c. Retester utilise des environnements différents alors que les tests de regréssion utilisent le même environnement.
- d. Retester est utile après la résolution des erreurs alors que les tests de regréssion sont faits avant la détection des erreurs.
- e. Retester est fait par des développeurs alors que les tests de regréssion sont faits par des testeurs en soi.

À quoi on pourrait associer « la détection des erreurs lors du transfert des données » ?

- a. Vérification ✔
- b. Validation

Quelle(s) technique(s) de tests exécute/exécutent réellement le code afin d'analyser le comportement d'une fonction (si celle-ci a été appelée, le nombre de fois qu'elle a été appelée, ..)?

- a. Mock
- b. Stub, Mock et Spy
- c. Stub et Mock
- d. Stub
- e. Spy ✔

1. Les tests en boîte blanche ✔ permet de tester le structure interne du code ou du programme, tandis que les tests en boîte noire ✔ a pour but de cacher la structure interne du code.
2. Les critères telles que la couverture des instructions, de branches et ainsi de suite constituent des tests en boîte blanche ✔ .
3. Les tests unitaires, tests d'intégrations, test de système et tests d'acceptation appartiennent notamment aux tests en boîte noire ✔ .