## 

Test d’admission

SEMESTRE 4 IL

# Exercice 1 : Programmation Web

*L’objectif de cet exercice est d’évaluer vos connaissances en HTML, CSS, JavaScript et NodeJS.*

Une association de jeux de société souhaite proposer à ses adhérents un jeu de quiz en ligne. Elle vous demande un coup de main pour le développer.

## Préambule

Chaque fichier HTML possèdera sa propre feuille de style et son propre fichier script, même si ceux-ci sont vides.

## Bases du site Les bases d’un site web

* Créez une page *accueil.html*. Cette page devra comporter la mise en page de base HTML5 (header, main, footer, vides pour le moment). Vous pouvez mettre un titre comme « bienvenue » pour avoir meilleur rendu.
* Mettez en place un server NodeJS pour servir cette page correctement ainsi que les fichiers qui lui sont liés.

## Formulaire de création

* Mettez en place une autre page html accessible depuis l’accueil nommée *formulaire\_creation.html.* Cette page contiendra un formulaire utilisant la méthode GET et possèdera
  + 3 champs textes :
    - « Question »
    - « Réponse 1 »
    - « Réponse 2 »
  + Un champ « Réponse correcte » de type « number » pour indiquer quelle est la bonne réponse parmi les deux proposées (1 ou 2)
* Ce formulaire devra posséder son propre traitement dans le serveur NodeJS. Le traitement devra vérifier si la réponse est correcte ou non.

## Page de quizz

* Un nouveau fichier html devra être accessible depuis la page d’accueil : *affichage\_questions.html* Cette page sera l’exemple de l’affichage d’une question entrée par l’administrateur à l’aide du formulaire créé précédemment. Voici un exemple de ce à quoi l’affichage d’une question doit ressembler :

Réponse 2

Réponse 1

Question

Quelle est la couleur du cheval blanc d’Henry IV ?

Bleu

Blanc

* La gestion de la bonne réponse doit se faire via le JS à l’aide des évènements. Vous choisirez quelle réponse est juste (1 ou 2) et changerez le background de la réponse :
  + En vert si l’utilisateur clique sur la bonne.
  + En rouge si l’utilisateur clique sur la mauvaise.

# Exercice 2 : modélisation et base de données

Dans cet exercice, vous aidez un centre d’adoption d’animaux à gérer le registre des animaux et de leurs adoptions.

Le système est le suivant :

* Un client peut adopter plusieurs animaux mais un animal est adopté par un seul client à la fois.
* Pour une adoption, le centre souhaite connaitre la date de réservation de l’animal, la date effective d’adoption et si l’adoption a été payée ou non.
* Un animal appartient à une seule espèce mais pour une espèce, on peut avoir plusieurs animaux. Une espèce est représentée par un identifiant, un nom courant, un nom latin, une description et un prix.
* A partir des entités Client et Animal décrites ci-dessus, complétez le modèle conceptuel de données au format entité-association pour les besoins exprimés pour le centre d’adoption.
  + Indiquez et justifiez toutes vos hypothèses.
  + Donnez le dictionnaire de données de votre modèle (nom, type, description).
* Traduisez le modèle entité-association proposé à la question précédente en un modèle relationnel.
* En suivant le modèle relationnel proposé, écrivez les requêtes SQL suivantes :
  + Créer une requête permettant de sélectionner les nom, commentaires et date de naissance des animaux disponibles à l’adoption. (selectDispo.sql)
  + Supprimez l’adoption de l’animal dont l’id = 5. (deleteAdoption.sql)
  + Mettre à jour le statut disponible de l’animal dont l’id = 5. (updateAnimal.sql)
  + Créer une requête permettant de sélectionner les animaux de l’espèce chat. (selectChat.sql).
  + Créer une requête permettant de compter le nombre d’animaux par espèces. (nbEspece.sql).

# Exercice 3 : Programmation web et base de données

*Réalisez le site web du centre d’adoption de l’exercice 2 à l’aide du langage php et de PDO pour les interrogations vers la base de données.*

Voici les fichiers attendus pour cet exercice :

## mon-header.php

Créez un fichier php nommé mon-header.php qui sera utilisé sur toutes les pages de votre site et contiendra au minimum :

* Le logo DOGS sur la gauche
* Votre nom et prénom à droite en gras
* Un fond de couleur

## index.php

* La page index.php est composée d’un formulaire permettant au visiteur de sélectionner dans une liste l’espèce animale qu’il recherche.
* A la validation du formulaire, il sera redirigé vers la page adoption.php lui permettant d’adopter un animal.
* Gardez le choix de l’espèce dans une session ou un cookie.

## adoption.php

* La page adoption.php est composée d’un formulaire permettant au visiteur de sélectionner dans une liste l’animal qu’il souhaite adopter.
* A la validation du formulaire, il sera redirigé vers la page animal.php lui permettant d’afficher les informations l’animal choisi et son prix selon l’espèce sélectionnée précédemment.

# Exercice 4 : Récursivité

Soit un tableau de pourcentage de défauts de composants électroniques. On considère un composant comme valide lorsque son pourcentage de défauts se situe entre 0 et 2%.

## Travail à réaliser

Ecrire une fonction récursive qui compte le nombre de composants non valides.

Initialisation du tableau de pourcentage :

float[] defauts = new float[] { 0.8, 2.17, 0.02, 3.5,4.6,2.0,1.99};

Résultat attendu : 3

Vous rendrez l’algorithme de votre fonction en pseudo-code dans un fichier.

# Exercice 5 : Programmation Orientée Objet

Le trésor du Dragon est un jeu de plateau à 1 dimension, de 31 cases, où un Elfe doit conquérir un trésor gardé par un Dragon.

Une partie se déroule entre un elfe et un dragon. L’Elfe commence à la position 0, le dragon à la position 30.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | D |

*Positions de l’elfe (E) et du dragon (D) en début de partie.*

L’elfe peut recommencer la quête x nombre de fois, c’est-à-dire recommencer depuis la position 0. Ce nombre de fois est déterminé au démarrage de la partie et ne peut pas être modifié au cours de la partie.

Les deux combattants sont caractérisés par leur nom et leur position sur le plateau.

## Déroulement de la partie

1. On crée les deux personnages.
2. L’Elfe démarre en lançant le dé (6 faces). Il avance d’autant de cases que le résultat.
3. Il regarde si le Dragon est présent sur sa case.
   1. Si oui il tire une carte Chance.
      1. Si c’est une carte Sort : il lance de la poudre et fait reculer le Dragon de 3 cases
      2. Si c’est une carte Fuite : il recule d’une case
      3. Si c’est une carte Portauloin : il avance de 3 cases
   2. Sinon il attend le tour suivant
4. Le Dragon lance le dé. Il avance d’autant de cases que le résultat.
5. Il regarde si l’Elfe est présent sur sa case.
   1. Si oui : il crache du Feu
   2. L’elfe a deux options : pour déterminer l’option, on lance le dé :
      1. S’il fait un score inférieur ou égal à 3 : il subit l’attaque : reculer de 2 cases
      2. S’il fait un score supérieur à 3 : il tire une carte Chance.
6. On recommence à l’étape 1 jusqu’à que :
   1. L’Elfe a atteint le trésor à la case 31.
   2. L’Elfe a épuisé toutes ses tentatives de recommencer.

## Règles de gestion

* Les combattants ne peuvent pas de déplacer au-delà des cases 0 à 31.
* Chaque fois que l’Elfe retourne à la case 0, il épuise une tentative de conquérir le trésor.

## Travail demandé

* Lister toutes les classes nécessaires au bon fonctionnement de votre jeu.
* Pour chacune d’entre elles vous prendrez garde à spécifier la visibilité des différents membres ainsi que leur type.
* Les méthodes de chaque classe doivent être fournies avec leur signature.
* Vous devez préciser les éventuelles relations d’héritage entre les classes.
* Vous vous assurerez que vos classes respectent les règles de gestion.
* Vous implanterez le programme dans un langage objet typé tels que C# ou Java par exemple.

Vous rendrez le code source de votre programme.