

CC PHP : Session 1 - 1h30 - Sur machine

Téléchargez l'archive **cc-23-php-sujet.tgz** de l'espace Moodle. Après extraction, renommez le répertoire obtenu **nom-prenom** où **nom** et **prenom** sont vos nom et prénom écrits en minuscules et sans espaces (par ex. **delafontaine-jean**). A l'issue de l'examen, archivez ce répertoire (`tar czf nom-prenom.tgz nom-prenom`) et déposez l'archive sur Moodle.

Les exercices et la plupart des questions peuvent être traités dans l'ordre de votre choix. Pour visualiser ce qui est attendu, consultez ce [démonstrateur](#).

Exercice 1. Le fichier **meteorites.json** est un historique de chutes de météorites qui fournit pour chacune différentes informations : nom, masse, latitude et longitude du point de chute, etc. Le fichier **meteorites.php** convertit cet historique en un tableau associatif `$M0`. L'objectif est d'implémenter différents traitements sur ce tableau : transformation, filtrage, tri, calcul. Les tableaux attendus pour les questions 1 à 3 sont donnés dans le dossier **solutions** et sont importés pour que vous puissiez les comparer aux vôtres.

1. Créez le tableau `$M1` en supprimant dans chaque élément du tableau `$M0` l'élément de clé `'geometry'`.
2. Créez le tableau `$M2` en remplaçant chaque élément de `$M1` qui est un tableau associant la clé `'properties'` à un tableau associatif par ce tableau associatif.
3. A partir du tableau `$M2`, créez le tableau `$FR` ne contenant que les météorites tombées en France (clé `'country_code'` de valeur `'FR'`).
4. Triez le tableau `$M2` par masse de météorite croissante (clé `'mass_g'`) et stockez dans la variable `$maxMass` la masse de la météorite la plus lourde. Résultat attendu : 330000.
5. Calculez la masse totale en kgs, arrondie à l'entier inférieur, des météorites listées dans `$M2` et stockez le résultat dans la variable `$totalMass`. Attention, les masses sont données en grammes dans le fichier. Résultat attendu : 2223.

Exercice 2. Une ébauche de site web vous est fournie pour générer et traiter des QCM sur le langage PHP qu'il s'agit de compléter.

Le site s'appuie sur une liste de questions numérotées de 1 à 40. Le visiteur saisit d'abord un nombre n de questions sur la page d'accueil (Figure 1) qui s'afficheront au clic sur le bouton **Générer** (Figure 2). Les n questions sont tirées aléatoirement s'il a coché la case **Tirage aléatoire**, sinon les n premières questions de la liste s'affichent. Le visiteur doit ensuite cocher une réponse par question (Figure 3), chaque question admettant une unique solution. Après soumission, une réponse cochée apparaît en vert si elle est correcte, en rouge sinon (Figure 4). Le visiteur peut alors resoumettre de nouvelles réponses pour le même QCM (bouton **Vérifier**) ou bien revenir à la page d'accueil (hyperlien **Recommencer**).

Les questions sont stockées dans le tableau `$questions` du fichier **./data/questions.php** qui associe l'énoncé de chaque question à son numéro. Le fichier **./data/answers.csv** stocke les réponses associées aux questions. Il comprend 3 colonnes et utilise le caractère de tabulation `\t` comme séparateur. Chaque ligne, hormis la ligne d'en-têtes, modélise une réponse : le premier champ donne le numéro de la question correspondante, le second fournit l'énoncé de la réponse, le troisième indique s'il s'agit de la solution à la question (valeur 1), ou non (valeur 0).

Le cycle de traitement est le suivant :

- le formulaire **qcm.html** est soumis au script **qcm_generate.php**,
- **qcm_generate.php** utilise **questions.php** pour extraire les questions,
- **qcm_generate.php** utilise **questions_answers.php** pour extraire les questions-réponses,
- **qcm_generate.php** utilise **encode_statement.php** pour les afficher,
- les réponses cochées sont soumises au script **qcm_check.php** pour vérification et réaffichage.

Vous devez compléter ces fichiers PHP dans les questions qui suivent à l'exception de **qcm_check.php** qui vous est fourni (fichier obfusqué).

1. La fonction `questions($n)` déclarée dans **questions.php** doit renvoyer un tableau de questions selon le type et la valeur du paramètre `$n` :

- si `$n` est un entier, `n` questions sont tirées aléatoirement¹ dans `$questions`;
- si `$n` est un tableau d'entiers, ces entiers correspondent aux numéros des questions à extraire de `$questions`.

La fonction n'a pas à gérer les erreurs résultant d'un type ou d'une valeur incorrecte pour `n` et elle doit préserver les indices (numéros de questions) lors de l'extraction. Implémentez-la en vous conformant aux commentaires fournis dans le code.

2. La fonction `questionsAnswers($questions)` déclarée dans **questions_answers.php** prend en paramètre un tableau de questions. Elle doit extraire les réponses à ces questions à partir du fichier **answers.csv** et combiner questions et réponses sous forme de tableau. Implémentez cette fonction en vous conformant aux commentaires fournis dans le code. A titre d'exemple, le tableau attendu pour les questions 2 et 3 est donné dans le fichier **/solutions/questions_answers.php**.

3. Les énoncés des questions et des réponses sont des chaînes de caractères pouvant mélanger texte français et code PHP imbriqué dans des éléments HTML de balise `code`. Par exemple :

- Que fait l'instruction `<code>include('file.php');</code>` si `<code>file.php</code>` n'existe pas ?
- `<code>echo 234 <=> 123;</code>`

La fonction `encodeStatement(...)` définie dans **encode_statement.php** doit convertir en HTML un énoncé donné. Précisément, elle doit convertir tous les caractères réservés de HTML en entités HTML sans modifier les balises `<code>` et `</code>` qui peuvent y figurer. Par exemple, le dernier énoncé ci-dessus doit être converti en la chaîne `<code>echo 234 <=> 123;</code>`. Implémentez cette fonction en vous conformant aux commentaires fournis dans le code.

4. Le formulaire de la page d'accueil **qcm.html** est soumis au script **qcm_generate.php**. Selon les données soumises, ce dernier doit d'abord extraire un tableau de questions-réponses en utilisant les fonctions `questions($n)` et `questionsAnswers($questions)`. Implémentez cette fonctionnalité.

5. Une fois extrait le tableau de questions-réponses (voir ci-dessus), le script **qcm_generate.php** doit l'afficher sous forme de tableau HTML tel qu'illustré en Figure 2. Implémentez cette fonctionnalité en respectant les consignes suivantes :

- placez chaque question et ses réponses dans un élément HTML `div`;
- placez l'énoncé de la question dans un paragraphe HTML;
- pour chaque réponse, générez un bouton radio dont l'attribut `name` est le numéro de la question et l'attribut `value` l'indice de la réponse dans le tableau des questions-réponses, puis placez l'énoncé de la réponse dans un élément `label`;
- utilisez `encodeStatement()` pour convertir les énoncés des questions et réponses;
- pré-cochez la première réponse de chaque question.

Exemple de fragment attendu pour la question 6 et ses 2 réponses :

```
<div>
  <p>
    <code>$x; var_dump(isset($x));</code>
  </p>
  <input type="radio" name="6" value="1" checked/>
  <label><code>bool(false)</code></label>
  <br>
  <input type="radio" name="6" value="2">
  <label><code>bool(true)</code></label>
  <br>
</div>
```

1. Voir la fonction PHP `array_rand`

Nombre de questions

Tirage aléatoire ☐

FIGURE 1 – Page d'accueil (**qcm.html**)

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☒ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☒ NULL
☐ boolean
☐ undefined

FIGURE 2 – Génération de QCM (**qcm_generate.php**)

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☐ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☒ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☒ NULL
☐ boolean
☐ undefined

FIGURE 3 – Choix du visiteur

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☐ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☒ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☒ NULL
☐ boolean
☐ undefined

[Recommencer](#)

FIGURE 4 – Vérification (**qcm_check.php**)