Module 4 – Bases du langage PHP

Pour tester rapidement du code PHP : http://sandbox.onlinephpfunctions.com/

Exercice 1

Sur PHP Sandbox, exécutez le script suivant. Prenez l'habitude de retaper le code plutôt que d'en faire un copier/coller, pour vous habituer à la syntaxe de PHP.

```
$x = 0;
echo "\$x vaut $x et est de type " . gettype($x) . "\n";
echo "Var_export indique : ";
var_export($x);
echo "\nVar_dump indique : ";
var_dump($x);
echo "\n";
```

Exécutez ce code et observez comment la valeur de $x est écrite dans la première instruction « echo » ; c'est l'instruction la plus utilisée en PHP pour produire une sortie (vers le document HTML). Observez également les résultats produits par « var_export » (qui donne une écriture de la valeur correspondant à un littéral PHP) et par « var_dump » (qui précise non seulement la valeur mais également le type de celle-ci).

Plusieurs sortes de chaînes de caractères…

Dans les instructions « echo », remplacez les guillemets par des apostrophes et observez le résultat. Revenez aux guillemets avant de poursuivre.

Plusieurs types de valeurs…

Modifiez ensuite la première ligne pour qu'elle indique

```
$x = 135.2;
```

et exécutez à nouveau le script. Observez une fois encore les résultats de l'affichage via « echo », via « var_export » et via « var_dump ».

Exécutez le même code en remplaçant la ligne d'initialisation par chacun des exemples qui suivent.

```
$x = true;      // résultat surprenant ?
$x = false;     // résultat surprenant ?
$x = "hello";
$x = null;
```

Entre autres, observez bien la manière dont PHP affiche les valeurs booléennes : de manière interne (dans un code propre), on utilise les littéraux « true » et « false » mais, à l'affichage simple (via « echo »), PHP produit soit un « 1 » soit une chaîne vide !

Exercice 2

Le but de cet exercice est d'écrire un document PHP à placer sur le serveur. Ce document devra produire du code HTML générant le texte présenté ci-dessous. Pour vérifier votre travail, vous utiliserez un navigateur internet (en utilisant l'adresse vm-debian.iesn.be/…).

Je m'appelle Homer.

J'ai étudié l'énergie nucléaire.

Je voudrais gagner $3000 par mois.

Dans ce texte, les parties en italique sont variables. Leur contenu est déterminé par quatre variables :

- $nom : reprend le nom de la personne (ici, "Homer") ;
- $domaine : reprend le domaine d'étude de la personne (ici "l'énergie nucléaire") ;
- $etude : un booléen qui indique si la personne est encore aux études (selon sa valeur, on écrira « J'ai étudié » ou « J'étudie ») ;
- $salaire : reprend le salaire désiré (ici, 3000).

Ces quatre variables seront initialisées au début

Vous produirez plusieurs versions du script php. Effectuez à chaque fois plusieurs tests en modifiant les valeurs des variables.

1. Dans un premier temps, vous utiliserez des guillemets pour toutes les chaînes de caractères et vous utiliserez PHP uniquement pour produire les parties variables.
2. Dans un second temps, remplacez les guillemets par des apostrophes pour toutes les chaînes de caractères.
3. Ensuite, faites en sorte que tout le contenu de la balise <body> soit produit par du code PHP, en utilisant uniquement des apostrophes pour les chaînes de caractères.
4. Finalement, remplacez les apostrophes par des guillemets et tirez avantage de la syntaxe autorisée par PHP en incorporant les variables *dans* les chaînes de caractères.

Exercice 3

Écrivez un script PHP qui, à partir d'un entier et d'une chaîne de caractères (initialisés directement dans le programme, par exemple $x = 5 et $token="*!"), va afficher une ligne répétant la chaîne de caractères autant de fois que demandé. Par exemple, pour 5 et « *! », le script affichera

```
*!*!*!*!*!
```

Dans un premier temps, utilisez une boucle « for » avec la syntaxe commune au C, à Javascript et à Java et effectuez l'affichage au fur et à mesure.

Dans un second temps, conservez la même structure de boucle mais construisez une chaîne de caractères qui sera affichée après la boucle.

Dans un troisième temps, modifiez la syntaxe des instructions pour utiliser des « : ».

Modifiez votre script pour obtenir le résultat suivant. Le script PHP devra, à partir d'un entier (initialisé directement dans le programme, par exemple $x = 7259), afficher une ou plusieurs lignes d'étoiles : une ligne d'étoiles pour chacun des chiffres de l'entier (ici 7, 2, 5, 9). Vous pouvez utiliser l'une ou l'autre syntaxe, au choix.

Par exemple, pour l'entrée $x = 7259, le script devrait afficher quatre lignes, la première avec 9 étoiles, la seconde avec 5 étoiles, la troisième avec 2 étoiles et la dernière avec 7 étoiles :

```
*********
*****
**
*******
```

Exercice 4

Les tests isset($x) et empty($x) permettent respectivement de vérifier si une variable $x « existe » et si elle est « vide ». Mais la signification de ces termes entre guillemets n'est pas forcément très facile à définir précisément.

Le but de cet exercice est de compléter le tableau suivant en indiquant pour chacun des cas la valeur booléenne (donc vrai ou faux) renvoyée par chacun des tests.

	isset($x)	empty($x)
(Aucune déclaration)		
$x = 0;	1	1
$x = 0.0;	1	1
$x = 7;	1	1
$x = 27.4;	1	1
$x = true;	1	1
$x = false;	1	1
$x = "";	1	1
$x = "\n";	1	1
$x = "hello";		
$x = null;		
$x = array(); *(tableau vide)*		

c'est partout des 1,fais chier

Pour réaliser ces tests, on vous propose le script suivant. Qu'en pensez-vous ?

```
// Ligne à remplacer par la déclaration
echo isset($x);
echo empty($x);
```

Exercice 5 : pour s'échauffer…

Sur Sandbox, copiez le code suivant

```
$x = 76;
// définition de inc
// définition de dec
inc();
dec();
inc();
var_dump($x);
```

puis remplacez les deux commentaires par des définitions pour les fonctions inc et dec censées incrémenter et décrémenter la variable $x.

Vérifiez votre code en vous assurant que le script affiche bien la bonne valeur finale (ici, 77).

Dans un second temps, modifiez la définition de inc et de dec pour que ces deux fonctions prennent comme argument la variable à incrémenter/décrémenter. Modifiez également les appels pour indiquer que les fonctions doivent agir sur $x et vérifiez le résultat.

Exercice 6 : démographie !

Le but final de cet exercice est de produire un tableau reprenant les informations démographiques citées ci-dessous (il s'agit de la répartition de la population belge entre femmes et hommes au 1er janvier 2015).

Catégorie	Femmes	Hommes	Graphique
Moins de 18 ans	1112811 (48.87%)	1164347 (51.13%)	$$$$$$$$$$$$$$$$$$$$
De 18 à 64 ans	3438304 (49.82%)	3462994 (50.18%)	$$$$$$$$$$$$$$$$$$$$
Plus de 64 ans	1152835 (56.77%)	877753 (43.23%)	$$$$$$$$$$$$$$$$$$$$
Total	5703950 (50.89%)	5505094 (49.11%)	$$$$$$$$$$$$$$$$$$$$

Le résultat sera construit pas par pas en PHP. Pour cet exercice, on entrera les données (à savoir les nombres de femmes et d'hommes pour chaque ligne du tableau) en « brut », mais on aurait également pu les obtenir à partir d'une base de données.

Étape 1

Créez un fichier demo.php contenant le code suivant puis placez-le sur le serveur.

```
<html>
  <head>
    <?php
      $coulH = "blue";
      $coulF = "red";
      function ligne($ctg,$val1,$val2) {
        echo "<tr><td>$ctg</td><td>$val1</td><td>$val2</td><td>.</td></tr>";
      }
      include_once("code.php");
```

```
    ?>
  </head>
  <body>
    <table id="pop">
      <tr>
        <th>Catégorie</th><th>Femmes</th><th>Hommes</th><th>Graphique</th>
      </tr>
      <?php
        ligne("Moins de 18 ans", 1112811, 1164347);
        ligne("De 18 à 64 ans", 3438304, 3462994);
        ligne("Plus de 64 ans", 1152835, 877753);
        ligne("Total", 5703950, 5505094);
      ?>
    </table>
  </body>
</html>
```

Ouvrez-le via Firefox (en passant via Internet, pour que le code PHP soit exécuté) et observez le résultat. Notez particulièrement l'utilisation de la fonction « ligne » pour rédiger chacune des lignes du tableau. Bien sûr, il s'agit ici d'une première version, que vous devrez ensuite raffiner.

Dans l'en-tête du document HTML/PHP, ou dans un fichier séparé, incluez les définitions de style suivantes pour obtenir l'apparence désirée du tableau (en même temps, révisez votre CSS : assurez-vous que vous comprenez bien chacune des règles).

```
#pop {
  font-family: "comic sans ms";
  border: 2px solid gray;
  border-collapse: collapse;
}
#pop tr:last-child {
  background-color: beige;
  border-top: 1px solid gray;
}
#pop th {
  border-bottom: 2px solid gray;
}
#pop th, #pop td {
  padding: 4px;
}
#pop tr td:nth-child(2) {
  color: blue;
}
#pop tr td:nth-child(3) {
  color: red;
}
```

Étape 2 : programmation modulaire

Dans le document demo.php, remplacez la ligne « include_once » par « require_once ». Mettez à jour la version sur le serveur puis chargez à nouveau la page web. Expliquez le résultat que vous obtenez…

Étape 3

Créez un nouveau fichier texte que vous nommerez code.php. Copiez dans ce fichier la définition de la fonction ligne. Comme il s'agit d'un module php qui sera inclus dans le

document principal, n'oubliez pas d'ajouter la balise <?php au début (la balise fermante ?> est, elle, optionnelle).

Placez le fichier code.php sur le serveur et rechargez la page web. Si vous avez suivi les instructions à la lettre, rien ne devrait s'afficher. Déterminer pourquoi c'est le cas et corrigez l'erreur !

Étape 4 : pourcentages

À ce moment, votre page web devrait afficher un tableau avec les données brutes. Il reste à modifier la définition de la fonction « ligne » qui se trouve dans le module code.php et à définir quelques fonctions annexes pour faciliter son implémentation.

Dans un premier temps, concentrez-vous sur le calcul des pourcentages affichés dans les colonnes « Femmes » et « Hommes » (voir le modèle en début d'énoncé). Pour ce faire, le premier objectif sera d'écrire une fonction PHP répondant à la spécification suivante.

`pourcents($val1,$val2)`

Précondition. $val1 et $val2 sont des valeurs numériques positives

Postcondition. La fonction renvoie des chaînes de caractères représentant le ratio entre $val1 et $val2 sous forme de pourcentages avec 2 décimales. Par exemple, pour les arguments $val1 = 2 et $val2 = 3, la fonction devrait renvoyer les chaînes de caractères « 40.00% » et « 60.00% » car, sur le total de 2+3 = 5, la valeur 2 représente 40% et la valeur 3 représente 60%.

Quelques conseils :

- Pour pouvoir plus facilement tester votre fonction, rédigez son code sur Sandbox.
- Dans un premier temps, vous pouvez tenter de réaliser une fonction qui renvoie simplement les deux pourcentages (40 et 60) sans vous préoccuper du format d'écriture. Référez-vous aux slides théoriques pour voir comment le faire.
- Pour transformer une valeur numérique en une chaîne de caractères écrite dans un format particulier, utilisez sprintf. Plutôt que d'écrire deux fois la transformation à appliquer, vous pouvez définir une fonction « formatPourcents($val) » qui transforme une valeur numérique en une chaîne de caractères au format désiré.

Une fois que votre fonction est prête, incorporez-la dans le fichier code.php et modifiez la définition de la fonction ligne pour (a) faire appel à pourcents et (b) incorporer les résultats, entre parenthèses, dans les colonnes adéquates.

Étape 5 : des graphiques en $

Il faut encore s'occuper des barres en $ de la dernière colonne. L'objectif de cette étape est de construire (proprement) une fonction répondant à la spécification suivante.

`barre($val1,$val2,$coul1,$coul2,$taille)`

Précondition. $val1 et $val2 sont des valeurs numériques positives ; $coul1 et $coul2 sont des couleurs web/css ; et $taille est un entier strictement positif

Postcondition. La fonction doit afficher (via echo par exemple) au format HTML une séquence composée d'exactement $taille caractères « $ ». Les premiers « $ » étant affichés dans la couleur $coul1 et les suivants, dans la couleur $coul2. Le rapport entre le nombre de $ en $coul1 et ceux en $coul2 doit être le même qu'entre $val1 et $val2.

Note. Si la taille n'est pas spécifiée, on supposera qu'on désire un total de 20 « $ ».

Par exemple, pour l'appel barre(2,3,"green","orange",10), on désire afficher une suite de 10 caractères « $ » constitué de 40% de « $ » verts et de 60% de « $ » oranges, c'est-à-dire de 4 « $ » verts suivis de 6 « $ » oranges : $$$$$$$$$$.

Quelques conseils :

- Une fois encore, testez tout d'abord votre code sur Sandbox (attention : le résultat affiché sera le code HTML produit, pas son interprétation par un navigateur).
- Pour produire le résultat demandé, vous allez devoir afficher un certain nombre de « $ » de suite. Commencez par définir une fonction repete($nb,$txt) qui affiche $nb fois le texte $txt. Et, tant que vous y êtes, faites en sorte que, par défaut, si on ne donne qu'un seul argument à cette fonction, on suppose que $txt est un caractère « $ ».
- Dans la fonction barre, pour être certain de bien avoir un total de $taille caractères, calculez le nombre de $ à afficher pour $val1 et, pour $val2, assurez-vous d'afficher ($taille – ce nombre) fois le caractère $.
- En HTML, pour produire une suite de $ en couleur, le plus simple est sans doute d'utiliser la balise « span ». Par exemple, on peut produire $$$$ via le code HTML `<span style='color:green'>$$$$</span>`. Et, comme vous aurez à faire ce genre de choses deux fois, pourquoi ne pas définir une fonction annexe ?

Étape 6 : la finale

Mettez à jour le fichier code.php et modifiez la fonction ligne pour qu'elle fasse appel à barre en utilisant les couleurs $coulH et $coulF définies dans demo.php ! Assurez-vous que vous obtenez un résultat identique à celui présenté en début d'énoncé.

Exercice 7 : parcours de tableaux multidimensionnels

Le but de cet exercice est de créer du code PHP pour afficher le contenu d'un tableau bidimensionnel sous la forme d'une liste de liste (liste au sens HTML).

Ainsi, à partir du tableau multidimensionnel suivant (dont la définition sera incorporée dans le script PHP)…

```
$tab = array (
    'Simpson' => array (
        'père' => 'Homer',
        'mère' => 'Marge'),
    'Nahasapeemapetilon' => array (
        'père' => 'Apu',
        'mère' => 'Manjula'),
    'van Houten' => array (
```

```
        'père' => 'Kirk',
        'mère' => 'Luann')
);
```

… votre script devrait produire du code HTML correspondant à l'affichage ci-contre.

Dans le fichier HTML produit, assurez-vous d'utiliser au moins 2 styles CSS : un style pour les titres de famille (avec la règle text-transform:capitalize dans l'exemple ci-contre) et un style pour les champs cités (« père », « mère » en gras dans l'exemple ci-contre).

Votre script devra fonctionner même après que vous ayez apporté les modifications suivantes au tableau (autrement dit, il doit parcourir les tableaux pour connaître leur structure exacte).

- SIMPSON
  - **père** : Homer
  - **mère** : Marge
- NAHASAPEEMAPETILON
  - **père** : Apu
  - **mère** : Manjula
- VAN HOUTEN
  - **père** : Kirk
  - **mère** : Luann

1. Modifiez la définition du tableau pour ajouter la famille « Wiggum », dont le père se prénomme « Clancy » et la mère, « Sarah ».
2. Modifiez la définition du tableau pour ajouter les informations sur les enfants de chacune des familles (voir le tableau suivant) ; les listes intérieures n'auront donc pas toutes la même longueur.

Simpsons		Nahasapeemapetilon		van Houten		Wiggum	
enfant1	Bart	nbEnfants	8	enfant	Milhouse	enfant	Ralph
enfant2	Lisa						
enfant3	Maggie						