

## Développement *Web*

### Partie 1 : schémas algorithmiques

La division euclidienne de 2020 par 26 s'écrit  $2020 = \mathbf{77} \times 26 + 18$ . La division euclidienne de **77** par 26 s'écrit  $77 = 2 \times 26 + 25$ . On en déduit que l'écriture en base 26 de 2020 est  $(2 \times 26 + 25) \times 26 + 18 = \mathbf{2} \times 26^2 + \mathbf{25} \times 26 + \mathbf{18}$ . Comme les lettres correspondant à 2, 25 et 18 sont respectivement B, Y et R on peut affirmer que sous Excel la colonne 2020 se nomme **BYR**.

En utilisant la même démarche algorithmique, donner le nom sous Excel des colonnes numéros 1939 et 1945. Vous commencerez par fournir les divisions euclidiennes utilisées et l'écriture en base 26 associée.

On fournit, pour mémoire, les associations de lettres et de nombres suivantes :

A=1 B=2 O=15 R=18 U=21 V=22 Y=25 Z=26.

### Partie 2 : expressions régulières

A quoi correspond une chaîne de caractères qui est définie par l'expression régulière  $\wedge [1-9] [0-9] * \$ ?$

A quoi correspond une chaîne de caractères qui est définie par l'expression régulière  $\wedge [A-Z] [a-z] * \$ ?$

Dans les deux cas, vous détaillerez chaque composante des expressions régulières afin de justifier votre réponse.

Est-ce que les quatre chaînes "A", "0" (zéro), "non" et "123456799" correspondent à la première expression régulière ? Et à la seconde ? Pourquoi ?

## Partie 3 : du PHP conceptuel

Traduire le fragment XHTML strict suivant en PHP conceptuel à l'aide des fonctions de `std.php`. On supposera que les instructions `include("std.php")`; et `debutPage(...)`; ont été exécutées.

```
<table border="1" cellpadding="10" class='collapse'>

  <tr>
    <th> <p>  Aliment </p>  </th>
    <th> <p>  Photo   </p>  </th>
  </tr>

  <tr>
    <td> Oeuf                                     </td>
    <td> <img src='oeuf.png' alt='oeuf' width='300' /> </td>
  </tr>

</table>
```

On ne cherchera pas à reproduire exactement le formatage (espaces, retours à la ligne, indentation...).

## Partie 4 : un peu de SQL

Pour ce qui suit, on admet qu'il existe – ce qui n'est pas le cas physiquement – une base de données au sens de MySQL nommée **ECOLES** qui contient une table de données nommée **CE1**.

Voici un exemple **fictif** d'une telle table **CE1** :

| idEcole | Enfant    | Prenom  | classe  |
|---------|-----------|---------|---------|
| 217     | DUPOND    | Line    | CE1     |
| 125     | BEN GAZYH | Mohamed | CE1-CE2 |
| 5129    | TRENCH    | John    | CE1     |
| 312     | DIETTER   | Gudrun  | CE1     |
| [...]   |           |         |         |

### Question 4.1

Quelle instruction MySQL permet de savoir combien il y a d'enfants en tout dans la table CE1 ?

### Question 4.2

Quelle instruction MySQL permet de savoir combien il y a d'enfants dans la table CE1 pour l'école dont l'identifiant est le nombre 312 ?

### Question 4.3

Quelle instruction MySQL permet de savoir combien il y a d'enfants dans la table CE1 école par école ? On essaiera d'afficher les écoles via leur identifiant et leur nombre d'enfants par école, par ordre décroissant d'enfants par école et, en cas d'*ex aequo*, on triera par identifiant d'école. On se limitera aux cinq premiers résultats.

## Partie 5 : fonctions PHP

On voudrait faire utiliser à des enfants de petites classes ce qu'on appelle des *petits nombres*. Pour cela on invente les règles suivantes :

- un *mini entier* est un nombre entre 1 et 5 ;
- un *petit entier pair* est le double d'un *mini entier* ;
- un *petit entier impair* s'obtient en retranchant 1 à un *petit entier pair*.

### Question 5.1

Donner la valeur de tous les *petits entiers pairs* et de tous les *petits entiers impairs* selon ces définitions.

### Question 5.2

Donner le code d'une fonction PHP nommée `miniEntier()` sans paramètre qui renvoie un *mini entier* au hasard au sens des définitions précédentes. On utilisera la fonction PHP nommée `rand()` vue en cours.

### Question 5.3

Donner le code d'une fonction PHP nommée `petitEntierPair()` sans paramètre qui renvoie un *petit entier pair* au sens des définitions précédentes en faisant appel à la fonction `miniEntier()`.

### Question 5.4

Est-il plus intéressant d'écrire deux fonctions PHP nommées respectivement `petitEntierPair()` et `petitEntierImpair()`, toutes deux sans paramètre ou d'écrire une seule fonction PHP nommée `petitEntier()` avec un paramètre numérique, telle que l'appel de `petitEntier(0)` renvoie un *petitEntierPair* et celui de `petitEntier(1)` renvoie un *petitEntierImpair* ?

### Question 5.5

Serait-il pédagogique, selon vous, d'inclure la valeur zéro dans la définition de *mini entier* ? Pourquoi ? Si on décidait de le faire, y aurait-il beaucoup de changements à effectuer dans le code informatique des fonctions PHP précédentes ?

### Question 5.6 (attention : question longue )

Essayer de donner les grandes lignes d'un programme PHP qui réalise l'affichage de la question 4.3 dans un élément `<table>` via du PHP. Essayer ensuite de donner le code PHP conceptuel correspondant, même incomplet, pour montrer que vous avez compris comment on fait dialoguer PHP et MySQL.

## Partie 6 : culture WEB

Discuter en au moins 10 lignes et 5 phrases au minimum l'affirmation suivante à l'aide d'au moins 3 mots de 3 syllabes ou plus afin de « transmettre un contenu rédactionnel fort » :

*"L'utilisation des tablettes et des ordinateurs en classes est une aide très importante pour l'apprentissage de la langue française, aussi bien à l'oral qu'à l'écrit."*

Vous essaieriez de fournir au moins trois arguments POUR et trois arguments CONTRE avant de donner et d'argumenter votre avis sur le sujet.

# CORRIGÉ RAPIDE

## Partie 1 : schémas algorithmiques

La division euclidienne de 1939 par 26 est  $74 \times 26 + 15$ . La division euclidienne de 74 par 26 est  $2 \times 26 + 22$ . On en déduit que l'écriture en base 26 de 1939 est  $(2 \times 26 + 22) \times 26 + 15 = \mathbf{2} \times 26^2 + \mathbf{22} \times 26 + \mathbf{15}$ . Comme les lettres correspondant à 2, 22 et 15 sont respectivement B, V et O on peut affirmer que sous Excel la colonne 1939 se nomme **BVO**.

La division euclidienne de 1945 par 26 est  $74 \times 26 + 21$ . La division euclidienne de 74 par 26 est  $2 \times 26 + 22$ . On en déduit que l'écriture en base 26 de 1945 est  $(2 \times 26 + 22) \times 26 + 21 = \mathbf{2} \times 26^2 + \mathbf{22} \times 26 + \mathbf{21}$ . Comme les lettres correspondant à 2, 22 et 21 sont respectivement B, V et U on peut affirmer que sous Excel la colonne 1945 se nomme **BVU**.

## Partie 4 : expressions régulières

Il y a quatre parties dans l'expression régulière  $^{\wedge}[1-9][0-9]^*\$$ .

- la première,  $^{\wedge}$  signifie *qui commence par* ;
- la deuxième,  $[1-9]$  correspond à *un chiffre de 1 à 9* ;
- la troisième,  $[0-9]^*$  correspond à *une suite éventuellement vide de chiffres de 0 à 9* ;
- la quatrième,  $\$$  signifie *qui se termine* ;

On a donc une chaîne qui commence par un chiffre de 1 à 9, suivie d'un ou plusieurs chiffres de 0 à 9 puis qui se termine.

Il s'agit donc d'une chaîne qui correspond à **un seul nombre entier strictement positif**.

Il y a aussi quatre parties dans l'expression régulière  $\wedge[A-Z][a-z]*\$$ .

- la première,  $\wedge$  signifie *qui commence par* ;
- la deuxième,  $[A-Z]$  correspond à *une lettre majuscule non accentuée* ;
- la troisième,  $[a-z]*$  correspond à *une suite éventuelle vide de lettres minuscules non accentuées* ;
- la quatrième,  $\$$  signifie *qui se termine*.

Il s'agit donc d'une chaîne qui correspond à **un seul mot** dont l'initiale est une majuscule (non accentuée) et le reste du mot est en minuscules (non accentuées).

Compte-tenu de ces explications,

- la chaîne "A" ne correspond pas à l'expression régulière numéro 1 mais elle correspond à l'expression numéro 2 ;
- les chaînes "0" (zéro) et "non" ne correspondent à aucune des deux expressions régulières proposées ;
- la chaîne "123456799" correspond à l'expression régulière numéro 1 mais pas à la numéro 2.

## Partie 3 : du PHP conceptuel

Voici le code demandé :

```
table(1,10,"collapse") ;

tr() ;
  th() ; p() ; echo "Aliment" ; finp() ; finth() ;
  th() ; p() ; echo "Photo" ; finp() ; finth() ;
fintr() ;
tr() ;
  td() ; echo "Oeuf" ; fintd() ;
  td() ; echo img("oeuf.png","oeuf",300) ; fintd() ;
fintr() ;

fintable() ;
```

## Partie 4 : un peu de SQL

### Question 4.1

```
SELECT COUNT(Enfant) FROM CE1 ;
```

### Question 4.2

```
SELECT COUNT(Enfant) FROM CE1 WHERE idEcole=312 ;
```

### Question 4.3

```
SELECT idEcole, COUNT(Enfant) as nbEnfants
FROM      CE1
GROUP BY idEcole
ORDER BY nbEnfants DESC, idEcole ASC
LIMIT    5 ;
```

## Partie 5 : fonctions PHP

### Question 5.1

Les mini entiers sont les valeurs 1, 2, 3, 4, 5. Si on les multiplie par deux on a tous les *petits entiers pairs*, soit 2, 4, 6, 8, 10. Si on retranche 1 à ces *petits entiers pairs*, on obtient tous les *petits entiers impairs* soit 1, 3, 5, 7, 9.

### Question 5.2 à 5.4

```
1 <?php
2
3     function miniEntier() {
4         return( rand(1,5) ) ;
5     } # fin de fonction miniEntier
6
7     function petitEntierPair() {
8         return( 2*miniEntier() ) ;
9     } # fin de fonction petitEntierPair
10
11    function petitEntierImpair() {
12        return( petitEntierPair() - 1 ) ;
```

```

13     } # fin de fonction petitEntierImpair
14
15     function petitEntier($pair=0) {
16         if ($pair==0) {
17             return( 2*miniEntier() ) ;
18         } else {
19             return( 2*miniEntier() - 1 ) ;
20         } ; # fin si
21     } # fin de fonction petitEntier
22
23     ?>

```

Il est certainement plus intéressant d'écrire deux fonctions PHP nommées respectivement `petitEntierPair()` et `petitEntierImpair()`, parce que c'est plus lisible et beaucoup plus explicite. Compte-tenu de la simplicité de ces deux fonctions, c'est largement suffisant. Rien ne justifie de regrouper ces deux définitions.

### Question 5.5

Ce n'est pas pédagogique pour des enfants d'inclure la valeur zéro dans la définition de *mini entier*. Si on rajoute zéro comme *mini entier*, cela rajoute 0 comme *petit entier pair* et les enfants risquent de ne pas penser que 0 est pair car ils ne connaissent peut être pas ce qu'est la multiplication par zéro. De plus on a -1 qui devient un *petit entier impair* et les nombres négatifs sont à éviter dans les petites classes.

Il n'y aurait pratiquement rien à changer pour effectuer ce changement : il suffit d'écrire `rand(0,5)` au lieu de `rand(1,5)` dans la définition de la fonction `miniEntier()`.