ANNEXE I

Les fonctions relatives aux chaînes de caractères

1. La fonction chr

Elle retourne le caractère dont le code ASCII est donné.
Syntaxe : string chr (int ascii**)**

2. La fonction ord

Elle retourne la valeur ASCII du premier caractère d'une chaîne.
Syntaxe : int ord (string string)

3. La fonction strlen

Elle retourne la longueur d'une chaîne.
Syntaxe : int strlen (string str)

4. La fonction strpos

Elle recherche la première occurrence d'un caractère dans une chaîne et retourne la position numérique de la première occurrence de ch1 dans la chaîne ch à partir d'une position donnée.
Syntaxe : int strpos (string ch, string ch1, int position)

5. La fonction trim

Elle efface les espaces de début et de fin de chaîne.
Syntaxe : string trim (string str, " ")

6. La fonction strcmp

Elle compare en binaire des chaînes. Elle retourne < 0 si str1 est plus petit que str2 ; > 0 si str1 est plus grand que str2, et 0 s'ils sont égaux.
Syntaxe : int strcmp **(string** str1**, string** str2**)**

N.B. : *La fonction* ***strcasecmp*** *est identique à la fonction* ***strcmp****, elle effectue une comparaison insensible à la casse.*

7. La fonction strtolower

Elle convertit en minuscule tous les caractères d'une chaîne.
Syntaxe : string strtolower (string ch)

8. La fonction strtoupper

Elle convertit en majuscule tous les caractères d'une chaîne.
Syntaxe : string strtoupper (string ch)

9. La fonction str_replace

Elle remplace toutes les occurrences d'une chaîne par une autre.
Syntaxe : string str_replace (string search, string replace, string subject)

str_replace() remplace toutes les occurrences de search dans subject par la chaîne replace.

10. La fonction substr

Elle extrait une partie d'une chaîne donnée.
Syntaxe : string substr (string ch, int pos, int nbre)

11. La concaténation

La concaténation des chaînes de caractères se fait avec l'opérateur ● (Point).

ANNEXE II

Les fonctions relatives aux types date et heure en php

Dans certaines sites Web dynamiques, on se trouve obligé de :

- affecter la date et/ou l'heure système à une variable ou à un champ,
- ajouter une période à un champ du type date,
- changer le format d'affichage d'une date,
- traiter des champs contenant des dates,
- etc.

1. La fonction checkdate

Elle retourne **TRUE** si la date représentée par le jour **j**, le mois **m** et l'année **a** est valide et sinon **FALSE**.
N.B. **:** L'ordre des arguments n'est pas l'ordre français. La date est considérée comme valide si :

- L'année est comprise entre 1 et 32767 inclus.
- Le mois est compris entre 1 et 12 inclus
- Le jour est compris dans l'intervalle de dates du mois.
  Les années bissextiles sont prises en compte.

Syntaxe : **int checkdate (int m, int j, int a)**

Exemple :

```
<?Php
$j=30;   $m=11;  $a=2012;
if (checkdate($m,$j,$a)==true) $d=$j."/".$m."/".$a;
else $d="données invalide ";
echo($d);
?>
```

2. La fonction date

Elle retourne une date sous forme d'une chaîne, au format donné par le paramètre format, la date utilisée est fournie par le paramètre timestamp ou la date et l'heure courantes si aucun timestamp n'est fourni.
Syntaxe : string date (string format, [int timestamp])

Exemple :

```
<?Php
$d=date("Y-m-d H:i:s");
echo("Nous sommes le : ".$d);
?>
```

3. La fonction time

Elle retourne le timestamp qui est un entier long représentant l'heure courante, mesurée en secondes.

Syntaxe : int time (void)

Exemple :

```
<?Php
$a=time();
$b=date("d/m/Y H:i:s",$a);
echo($a." = ".$b);
?>
```

4. La fonction strtotime

Elle essaye de lire une date au format anglais et de la transformer en timestamp, relativement au timestamp now, ou à la date courante si ce dernier est omis.

Syntaxe : int strtotime (string time int now)

Exemple :

```
<?php
$a=strtotime("now");
$b=strtotime("10 September 2000");
$c=strtotime("+1 day");
$d=strtotime("+1 week");
$e=strtotime("+1 week 2 days 4 hours 2 seconds");
echo (date("Y-d-m H:i:s",$a)." = ".$a." secondes<br>") ;
echo (date("Y-d-m H:i:s",$b)." = ".$b." secondes<br>") ;
echo (date("Y-d-m H:i:s",$c)." = ".$c." secondes<br>") ;
echo (date("Y-d-m H:i:s",$d)." = ".$d." secondes<br>") ;
echo (date("Y-d-m H:i:s",$e)." = ".$e." secondes<br>") ;
  ?>
```

5. La définition d'un timpestamp :

Un timestamp est un entier long, contenant le nombre de secondes entre le début de l'époque UNIX (1er Janvier 1970 00:00:00 GMT).

ANNEXE III

Le type Tableau en PHP (Les fonctions prédéfinies)

1. Déclaration d'un tableau

Il existe plusieurs façons de déclarer un tableau en PHP :

```
// Méthode 1 - avec array()
    $tableau1 = array();
// Méthode 2 - avec la syntaxe courte []
    $tableau2 = [];
```

2. Remplissage d'un tableau

2.1. Remplissage manuel

```
$tableau = [10, 20, 30, 40, 50];
```

2.2. Remplissage automatique avec une boucle for

```
$tableau = [];
for ($i = 0; $i < 5; $i++) {
   $tableau[$i] = ($i + 1) * 10; // Remplit avec 10, 20, 30, 40, 50
}
```

3. Parcours d'un tableau avec une boucle for

```
$tableau = [10, 20, 30, 40, 50];
// On utilise count() pour obtenir la taille du tableau
$taille = count($tableau);
for ($i = 0; $i < $taille; $i++) {
   echo "Index: $i, Valeur: " . $tableau[$i] . "<br>";}
```

4. Remarques

- Les tableaux en PHP sont dynamiques - pas besoin de préciser la taille à l'avance
- La fonction count() retourne le nombre d'éléments
- Les indices commencent à 0 par défaut

ANNEXE IV

Les fonctions PHP pour MySQL

1. mysqli_connect()

```
$con = mysqli_connect("localhost", "root", "", "ecole");
```

La fonction mysqli_connect() est utilisée pour établir une connexion à une base de données MySQL en utilisant l'extension MySQLi.

Paramètres :

- Serveur : "localhost"
- Utilisateur : "root"
- Mot de passe : "" (vide par défaut)
- Nom de la base : "ecole"

2. mysqli_query()

```
mysqli_query($connexion, $requete);
```

La fonction mysqli_query() permet d'exécuter une requête SQL sur une base de données MySQL via l'extension MySQLi.

3. mysqli_num_rows()

```
mysqli_num_rows($resultat);
```

La fonction mysqli_num_rows() permet de compter le nombre de lignes retournées par une requête SELECT en MySQLi.

Retourne :

> 0	Un entier représentant le nombre de lignes trouvées.
0	Aucune ligne n'est retournée.
FALSE	En cas d'erreur.

Exemple :

```
$sql = "SELECT * FROM etudiants WHERE email = 'ali@mail.com'";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    echo "L'étudiant existe.";
} else {
    echo "Aucun étudiant trouvé avec cet email.";
}
```

⚠ mysqli_num_rows() est utilisé uniquement pour les requêtes SELECT.

4. mysqli_fetch_array()

mysqli_fetch_array($resultat);

La fonction mysqli_fetch_array() est utilisée pour récupérer une ligne de résultat d'une requête SELECT sous forme de tableau. Elle permet d'accéder aux données à la fois avec des index numériques et des noms de colonnes.

Exemple :

```
$conn = mysqli_connect("localhost", "root", "", "ecole");
$sql = "SELECT id, nom, prenom FROM etudiants";
$result = mysqli_query($conn, $sql);
while ($t = mysqli_fetch_array($result)) {
    echo "ID: " . $t[0] . " | Nom: " . $t["nom"] . " | Prénom: " . $t[2] . "<br>";
}
```

Explication :

- $t[0] → Accès via index numérique (id). (tableau indexé)
- $t["nom"] → Accès via nom de colonne. (tableau associatif)
- $t[2] → Accès via index numérique (prenom).

mysqli_fetch_array() permet de récupérer une ligne sous forme de tableau mixte (associatif + numérique).

5. mysqli_fetch_row()

mysqli_fetch_row($resultat);

La fonction mysqli_fetch_row() récupère une ligne de résultat sous forme de tableau indexé numériquement (contrairement à mysqli_fetch_array(), qui utilise tableau indexé et tableau associatif).

Exemple :

```
$conn = mysqli_connect("localhost", "root", "", "ecole");
$sql = "SELECT id, nom, prenom FROM etudiants";
$result = mysqli_query($conn, $sql);
while ($t = mysqli_fetch_row($result)) {
    echo "ID: " . $t[0] . " | Nom: " . $t[1] . " | Prénom: " . $t[2] . "<br>";
}
```

Explication :

- $t[0] → Correspond à id
- $t[1] → Correspond à nom
- $t[2] → Correspond à prenom

mysqli_fetch_row() récupère une ligne sous forme de tableau indexé numériquement.
Plus rapide que mysqli_fetch_array() car il n'a pas besoin de créer des clés associatives.
Retourne NULL si plus aucune ligne n'est disponible.

6. mysqli_affected_rows()

mysqli_affected_rows($connexion);

La fonction mysqli_affected_rows() permet d'obtenir le nombre de lignes affectées par une requête INSERT, UPDATE ou DELETE dans une base de données MySQL.

Retourne :

> 0	Nombre de lignes réellement affectées.
0	Aucune ligne n'a été affectée.
-1	En cas d'erreur.

Exemple 1 : Mise à jour avec UPDATE

```
$conn = mysqli_connect("localhost", "root", "", "ecole");
$sql = "UPDATE etudiants SET email = 'nouveau@mail.com' WHERE niveau = '4eme'";
mysqli_query($conn, $sql);
$nb = mysqli_affected_rows($conn);
echo "Nombre de lignes modifiées : " . $nb;
```

On met à jour les emails des étudiants de 4ème niveau.
mysqli_affected_rows($conn) retourne le nombre de lignes réellement modifiées.

Exemple 2 : Suppression avec DELETE

```
$sql = "DELETE FROM etudiants WHERE moyenne < 10";
mysqli_query($conn, $sql);
if (mysqli_affected_rows($conn) > 0) {
    echo "Étudiants supprimés : " . mysqli_affected_rows($conn);
} else {
    echo "Aucune suppression effectuée.";
}
```

Exemple 3 : Insertion avec INSERT

```
$sql = "INSERT INTO etudiants (nom, email) VALUES ('Sami', 'sami@mail.com')";
mysqli_query($conn, $sql);
if (mysqli_affected_rows($conn) > 0) {
    echo "Étudiant ajouté avec succès.";
} else {
    echo "Échec de l'insertion.";
}
```

mysqli_affected_rows() retourne 1 si l'insertion réussit.

7. mysqli_close()

mysqli_close($connexion);

La fonction mysqli_close() permet de fermer une connexion à une base de données MySQL ouverte avec mysqli_connect().

Retourne :

TRUE	La connexion est fermée avec succès.
FALSE	En cas d'échec (rare, sauf problème système).

Exemple :

```
$conn = mysqli_connect("localhost", "root", "", "ecole");
echo "Connexion réussie !";
mysqli_close($conn);
```

Explication :

- On ouvre une connexion avec mysqli_connect().
- On effectue les opérations nécessaires.
- On ferme la connexion avec mysqli_close($conn).

8. mysqli_error()

mysqli_error($connexion);

La fonction mysqli_error() permet d'obtenir le dernier message d'erreur généré par MySQL après une requête exécutée avec MySQLi.

Exemple :

```
$sql = "UPDATE etudiants SET email = 'sami@mail.com' WHERE id = 999";
mysqli_query($conn, $sql);
if (mysqli_affected_rows($conn) == -1) {
    echo "Erreur SQL : " . mysqli_error($conn);
} else {
    echo "Mise à jour réussie.";
}
```

mysqli_affected_rows($conn) == -1 signifie qu'une erreur s'est produite.