

Examen

Durée 2h — Documents autorisés: supports de cours distribués
Le barème est donné à titre indicatif et est proportionnel au temps à passer sur l'exercice.

Le partiel comprend 6 pages dont un aide mémoire HTML et PHP à la page 5 (il n'y a pas besoin de CSS).

- Numérotez les copies (1/3, 2/3, 3/3, ...);
- Inscrivez votre nom et prénom lisiblement sur chaque copie;
- Ne pas oublier **de cacher** la copie **après signature de la liste d'émargement**;
- Les téléphones portables, calculatrices, ordinateurs portable sont interdits, ainsi que tout document ne figurant pas dans la liste des documents autorisés. Les dictionnaires et appareils de traductions sont autorisés.

1 Questions de cours (2 points)

1. (1 point) En quoi consiste une attaque par « injection de code » ?
2. (1 point) Donner un exemple d'évènement qui peut faire passer un processus de l'état « en exécution » à « en attente » ?

2 Chemins Unix, permissions, expressions régulières (5 points)

Pour chacune des séquences de commandes suivantes, décrire précisément le comportement de chacune d'elle (expansion des arguments, effet de la commande, affichage). Si une commande échoue, vous ne traitez pas les commandes suivantes. On se replace entre chaque **question** (pas entre chaque commande) dans la situation initiale suivante : l'utilisateur est lambda, il est dans son répertoire personnel /home/lambda/ qui est vide et dont les permissions sont 755 (1 point pour les questions 1-3, 2 points pour la 4).

1. `ls | wc -l`
2. `echo -e '#/bin/bash \n ls -l ' > fichier.txt`
`chmod +x fichier.txt`
`./fichier.txt`
(le \n dans la première ligne représente un retour à la ligne se trouvant dans la chaîne de caractères).
3. `ls *.jpg`
`echo TOTO`
4. `for i in $(seq 1 100); do echo "$i" > "$i".txt; done`
`rm -f +(1|2|3).txt`
`ls *.txt | wc -l`

Réponse:

5. `ls | wc -l` : le répertoire est vide, la commande `ls` n'affiche rien (0 ligne) et n'écrit donc rien sur l'entrée standard de la commande `wc`. Cette dernière compte le nombre de ligne et affiche donc 0.
6. La première ligne affiche la chaîne de caractères `'#/bin/. . .'` sur la sortie standard qui est immédiatement redirigée dans le fichier `fichier.txt` (pas d'erreur). La deuxième ligne donne les droits en exécution à l'utilisateur courant sur ce fichier (pas d'erreur). La troisième ligne exécute le

fichier comme un script. La commande `ls -l` s'exécute et elle affiche la liste détaillée des fichiers se trouvant dans le répertoire (il y a le fichier `fichier.txt` nouvellement créé, pas d'erreur).

7. Pour la première ligne, le *shell* tente de faire une expansion sur l'expression régulière `*.jpg`. Comme aucun fichier ne correspond, la chaîne `'*.jpg'` est passée en argument à `ls` qui affiche un message d'erreur car aucun fichier de ce nom n'existe dans le répertoire courant. Échec.
8. La première ligne est une boucle `for` qui va itérer pour `$i` entre 1 et 100 (compris) et créer des fichiers `1.txt`, `2.txt`, ..., `100.txt` contenant respectivement 1, 2, ..., 100 (pas d'erreur). Pour la deuxième ligne, l'expression régulière est expansée. Elle sélectionne tous les fichiers donc les noms sont des répétitions de 1, 2 ou 3, de longueur au moins 1 et finissant par `.txt`. Elle est donc expansée en : `1.txt`, `2.txt`, `3.txt`, `11.txt`, `12.txt`, `13.txt`, `21.txt`, `22.txt`, `23.txt`, `31.txt`, `32.txt`, `33.txt` qui sont passés à `ls` (pas d'échec). La commande `wc -l` compte le nombre de lignes affichées par `ls` et affiche donc 12 (pas d'erreur).

3 Réseaux (3 points)

1. (1 point) On considère un adressage par sous-réseaux. Combien de machines sont adressables en utilisant le masque `255.255.254.0` ? (on suppose que toutes les adresses du sous-réseau peuvent être affectées à une machine, i.e. on ne tient pas compte des adresses spéciales : diffusion, etc.)
2. (2 points) De combien de sous-réseau au minimum a-t-on besoin, et avec quels masques pour pouvoir adresser exactement 800 machines ? (i.e. on ne veut pas d'adresses inutilisées). Dans ce cas, combien de machines doivent faire office de routeurs pour que toutes les machines soient atteignables ?

4 Programmation PHP (10 points)

Remarque : Il est important de lire attentivement l'énoncé du problème. À chaque question, Vous avez le droit de ré-utiliser les fonctions des questions précédentes, **même si vous n'avez pas réussi à les écrire**.

On souhaite réaliser en PHP une version simple du jeu du démineur. Dans ce jeu, un *tableau* de jeu contient des cases dont le contenu est initialement dissimulé. Certaines cases sont vides et d'autres peuvent contenir *des bombes*. Le joueur sélectionne une case non-découverte. Le contenu est alors révélé. Si la case contient une bombe, le joueur a perdu. Si la case est vide, deux situations peuvent se produire :

- s'il y a des bombes dans les cases *adjacentes*, alors leur nombre est inscrit dans la case révélée.
- s'il n'y a pas de bombe alors les cases adjacentes sont elles-même révélées, sauf si elles contiennent une bombe.

Le joueur sélectionne ensuite une nouvelle case jusqu'à ce qu'il tombe sur une bombe (partie perdue) ou qu'il ne reste que des bombes dans les cases non-découvertes (partie gagnée).

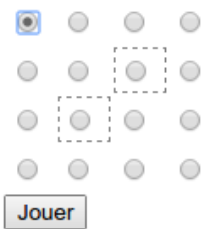
Le tableau est représenté par une table HTML. Les cases non-découvertes sont composées de boutons « radio ». Pour découvrir une case, il faut sélectionner le bouton correspondant puis cliquer sur le bouton « Jouer ». La procédure de découverte d'une case ainsi que l'interface sont illustrée à la figure 1. Les cases encadrées en pointillées contiennent les bombes (ces pointillés ne sont là que pour les besoins de l'explication et n'apparaissent pas lors d'une véritable partie !). Le tableau de jeu est représenté en PHP par un *tableau de tableaux*, indicés par des entiers consécutifs commençant à 0. Les cases d'un tel tableau peuvent contenir les chaînes de caractères suivantes :

- `"?"` si la case n'a pas encore été découverte et ne contient pas de bombe
- `"B"` si la case contient une bombe
- `"0"`, ..., `"8"` si la case est découverte. Le nombre est le nombre de bombes dans les cases adjacentes.

Par exemple la configuration du point (2.) de la figure 1 est représentée par le tableau :

```
Array ( 0 => Array ( 0 => "0", 1 => "1", 2 => "?", 3 => "?" ),
        1 => Array ( 0 => "1", 1 => "2", 2 => "B", 3 => "?" ),
        2 => Array ( 0 => "?", 1 => "B", 2 => "?", 3 => "?" ),
        3 => Array ( 0 => "?", 1 => "?", 2 => "?", 3 => "?" ))
```

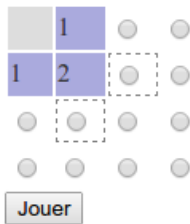
1.



[Recommencer](#)

Le tableau initial. Le joueur sélectionne la case (0,0) (en haut à gauche) avant de cliquer sur le bouton « Jouer ».

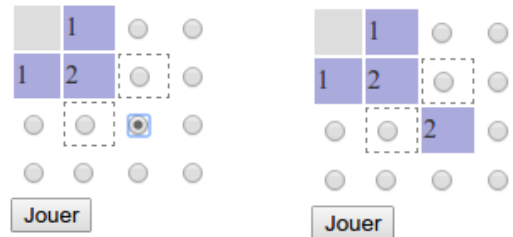
2.



[Recommencer](#)

Les cases *adjacentes* à la case sélectionnée ne contiennent aucune bombe. La case est grisée puis les cases voisines ((0,1), (1,1) et (1,0)) sont découvertes. La case (0,1) contient une bombe dans ses voisins, donc on y affiche un 1. Idem pour la case (1,0). La case (1,1) contient deux bombes parmi les voisins on y affiche donc un 2.

3.



[Recommencer](#)

[Recommencer](#)

Le joueur sélectionne la case (2,2). Cette dernière a deux bombes dans ses voisins, donc on y affiche 2 et on s'arrête.

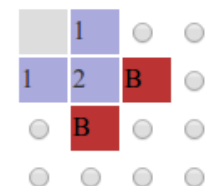
4.

GAGNÉ



[Recommencer](#)

PERDU



[Recommencer](#)

Le tableau en fin de partie (victoire à gauche, défaite à droite).

FIGURE 1 – Une partie de démineur

questions (on ne cherche, dans cet exercice, qu'à écrire les fonctions principales du jeu, pas le jeu en entier).

1. (1.5 points) Écrire une fonction `init($m, $n)` prenant en argument deux entiers `$m` et `$n` et renvoyant un tableau tel que celui décrit ci-dessus. Le tableau extérieur est de longueur `$m` et les tableaux intérieurs de longueur `$n`. Les cases contiennent soit "?" soit "B" avec une probabilité de 1/10. On utilisera le code PHP suivant pour tirer un nombre aléatoirement :

```
if (rand(0,9) === 0) {  
    // 1 fois sur 10  
    ...  
} else {  
    // 9 fois sur 10  
  
}
```

(la fonction `rand(i, j)` renvoie un nombre aléatoirement choisi entre `i` et `j` inclus).

2. (1 point) écrire un bout de code PHP qui initialise une *session*, puis teste si la variable de session "tab" n'est pas déjà définie. Si elle n'est pas définie, alors elle est initialisée avec la valeur de retour de `init(10,10)`.
3. (1 point) écrire une fonction `case_valide` prenant en arguments un tableau de jeu (tel que renvoyé par `init`) *par référence*, et deux entiers `$x` et `$y` et qui renvoie vrai si les coordonnées sont dans les bornes du tableau passé en argument (et faux sinon).
4. (1.5 points) écrire une fonction `bombes_adj` prenant en arguments un tableau de jeu *par référence*, et deux entiers `$x` et `$y` et qui renvoie le nombre de bombes présentes dans les cases adjacentes à la case `$x, $y`. Les cases adjacentes sont les 8 cases se trouvant autour de la case donnée (si elles existent, on pourra réutiliser la fonction `case_valide` pour vérifier cela).
5. (2 points) écrire une fonction `dessine` prenant un tableau de jeu en argument *par référence* et un booléen `$bombe` et qui produit la table HTML correspondant. Pour chaque élément `td` de la table on mettra l'attribut `class` et le contenu comme suit :
 - si la case contient "0" l'élément `td` a la classe vide et ne contient rien
 - si la case contient ? l'élément `td` contient un élément `input` de type `radio` dont le nom est `case` et dont la valeur la chaîne de caractères formée des deux coordonnées courantes séparée par une virgule (par exemple `value="3,2"`).
 - si la case contient "B" et que `$bombe` vaut faux, on affiche la case comme une case "?". Si `$bombe` vaut vrai, on affiche un B dans l'élément `td` et on lui donne la classe "bombe".
 - si la case contient un nombre entre 1 et 8, ce nombre est placé dans l'élément `td` et la classe de l'élément est `num`.
6. (2 points) écrire une fonction `clicque` prenant en arguments un tableau de jeu *par référence* et deux entiers `$x` et `$y` et qui implémente le découverture d'une case :
 - si la case `$x, $y` ne contient pas "?" ou "B", on ne fait rien.
 - sinon calculer le nombre de bombes adjacentes.
 - si la case contient "?", placer le nombre dans la case
 - si le nombre vaut zéro, rappeler la fonction `clicque` sur toutes les cases adjacentes.
7. (1 point) écrire une fonction `victoire` prenant un tableau de jeu *par référence* et renvoyant vrai si la partie est gagnée et faux sinon. La partie est gagnée si le tableau ne contient plus de "?".

Aide-mémoire HTML

Balise	Description
<code>...</code>	Lien hypertexte ayant pour cible l'url <i>s</i>
<code>...</code>	Mise en gras du texte
<code>
</code>	Forcer un retour à la ligne
<code><body>...</body></code>	Corps de la page Web
<code><button type="submit"> l </button></code>	Bouton de soumission de formulaire, avec le texte <i>l</i> . Doit apparaître sous un form
<code><div>...</div></code>	Définit une section
<code><form action="a" method="m"> ... </form></code>	Formulaire ayant pour cible l'url <i>a</i> et utilisant la méthode <i>m</i> comme passage de paramètres
<code><h1>...</h1></code>	Titre de sections de niveau 1. Il existe h2, h3, ... pour des titre de sous-sections, sous-sous-sections etc.
<code><head>...</head></code>	En-tête de la page Web
<code><html>...</html></code>	Balise principale de la page Web
<code><i>...</i></code>	Mise en italique du texte
<code></code>	Affiche l'image se trouvant à l'url <i>u</i>
<code><input type="t" name="n" value="v"> ... </input></code>	Affiche un élément graphique de type <i>t</i> qui permet de saisir une valeur stockée dans le paramètre de nom <i>n</i> . Doit apparaître sous un form. Les différents types pour <i>t</i> sont text (champ de texte), file (sélecteur de fichier), button (bouton), radio (un choix parmi plusieurs), checkbox (case à cocher)...
<code>...</code>	Élément d'une liste. Doit apparaître sous un ul ou ol
<code>...</code>	Liste énumérée
<code><option value="v">t</option></code>	Une des valeurs possibles pour une liste énumérée. <i>v</i> est la valeur stockée dans le paramètre, <i>t</i> est le texte correspondant, affiché par le navigateur. Doit apparaître sous un select
<code><p>...</p></code>	Paragraphe
<code><select name="n">...</select></code>	Affiche une liste permettant de choisir une valeur parmi plusieurs (les valeurs sont données par l'élément option). La valeur choisie est associée au paramètre de nom <i>n</i> . Doit apparaître sous un form.
<code><table>...</table></code>	Définit une table HTML
<code><td>...</td></code>	Définit une cellule d'une table HTML. Doit apparaître sous un tr ou th
<code><th>...</th></code>	Définit une ligne d'en-tête de colonnes d'une table HTML. Doit apparaître sous un table
<code><title>...</title></code>	Titre de la page (se trouve dans la balise head
<code><tr>...</tr></code>	Définit une ligne d'une table HTML. Doit apparaître sous un table
<code>...</code>	Liste non ordonnée

Conventions : On demande de respecter les conventions XHTML :

- les balises sont toujours refermées
- les noms de balises sont en minuscules

Aide-mémoire PHP

Variables globales

Nom	Description
<code>\$_GET</code>	Tableau dont l'indice <i>n</i> contient la valeur du paramètre get de nom <i>n</i>
<code>\$_POST</code>	Tableau dont l'indice <i>n</i> contient la valeur du paramètre post de nom <i>n</i>
<code>\$_SESSION</code>	Tableau dont l'indice <i>n</i> contient la valeur de la variable de session de nom <i>n</i>
<code>\$_COOKIE</code>	Tableau dont l'indice <i>n</i> contient la valeur du cookie de nom <i>n</i>

Expressions de bases

Nom	Description
NULL	Valeur nulle (renvoyée pour les variables non déclarées, les indices de tableaux inexistantes, ...)
TRUE, FALSE	Booléens vrai et faux
isset(<i>e</i>)	<i>e</i> peut être une variable ou un accès à un tableau. Renvoie vrai si <i>e</i> est défini et faux sinon
+, -, *, /, %	Opérations arithmétiques addition, soustraction, multiplication, division, modulo
.	Concaténation de chaînes de caractères
echo <i>e</i>	Affiche le résultat de l'expression <i>e</i> sur la sortie renvoyée au client

Tableaux

Nom	Description
array (<i>x</i> ₀ => <i>y</i> ₀ , ..., <i>x</i> _{<i>n</i>} => <i>y</i> _{<i>n</i>})	Création d'un tableau et initialisation de l'indice <i>x_i</i> avec la valeur <i>y_i</i>
count(\$t)	Renvoie la taille du tableau \$t
\$t[<i>v</i>]	Accès à la case d'indice <i>v</i> dans le tableau \$t. Renvoie NULL si l'indice n'est pas dans la tableau
asort(\$t)	Trie le tableau \$t par ordre de valeurs
ksort(\$t)	Trie le tableau \$t par ordre d'indices

Lecture de fichiers

Nom	Description
file(\$f)	Prend en argument une chaîne représentant un nom de fichier et renvoie un tableau indicé par des entiers et dont la case <i>i</i> contient la <i>i</i> ^{ème} ligne du fichier. Les retours à la ligne éventuels sont conservés en fin de chaîne
file_get_contents(\$f)	Prend en argument une chaîne représentant un nom de fichier et renvoie le contenu du fichier sous forme d'une chaîne de caractères

Manipulation de chaînes de caractères

Nom	Description
trim(\$s)	Supprime les caractères blancs (espaces, retour à la ligne, tabulation, ...) en début et fin de la chaîne passée en argument et renvoie la nouvelle chaîne
preg_split(\$r, \$s)	Découpe la chaîne de caractères \$s en utilisant comme séparateur les suites de caractères correspondant à l'expression régulière \$r et renvoie le résultat sous forme d'un tableau indicé par des entiers. Les séparateurs sont supprimés. Exemple : preg_split('/', '+', 'a,,b,cd') renvoie un tableau (0 => 'a', 1 => 'b', 2 => 'cd')
preg_match(\$r, \$s)	Renvoie 1 si une sous chaîne de la chaîne \$s correspond à l'expression régulière \$r et 0 sinon

Les expressions régulières PHP sont des chaînes de caractères commençant et finissant par « / » et utilisant les opérateurs :

.	n'importe quel car.	<i>e</i> ? Rep. 0 ou 1 fois	[<i>c</i> ₁ ... <i>c_n</i>] 1 car. dans <i>c</i> ₁ ... <i>c_n</i>
<i>e</i> ₁ <i>e</i> ₂	<i>e</i> ₁ ou <i>e</i> ₂	<i>e</i> * Rep. ≥ 0	[^ <i>c</i> ₁ ... <i>c_n</i>] 1 car. diff. de <i>c</i> ₁ ... <i>c_n</i>
(<i>e</i>)	<i>e</i>	<i>e</i> + Rep. ≥ 1	[<i>c</i> ₁ - <i>c_n</i>] 1 car. dans <i>c</i> ₁ ... <i>c_n</i>
^	Début de texte	\$	Fin de texte

Gestion des cookies et des sessions

Nom	Description
setcookie(\$n, \$v, \$t)	Stocke dans le cookie de nom \$n, la valeur \$v avec la date d'expiration \$t (qui est un nombre de secondes depuis 1/1/1970)
session_start()	Démarre une session