Classe:..... Nom:.....
Groupe: ..... Prénom:.....

CONTRÔLE : SYSTÈME D'EXPLOITATION 4
3IIR

Rappels:
- L'option -v de grep permet d'inverser la recherche

1ère partie : compréhension

A- Peut-on accéder au 15ème argument d'un script à l'aide des variables positionnelles? Si oui, comment? Sinon pourquoi? (2pt)

1. ⊗ OUI / ○ NON :JUSTIFICATION :……**${15}**…….

B- Soit un script monscript.sh. Quelle est la différence entre les syntaxes d'invocation suivantes : (4pt)

1. $ ./monscript.sh : **doit être executable et spécifier le shell en ligne shebang**
2. $. monscript.sh : **sourcing, ça veut dire inclure et intérpréter le script ds le shell courant**
3. $ bash monscript.sh : **interprétation explicite en sous shell bash**
4. $ source monscript.sh : **sourcing comme . monscript.sh**

C- Quelle est l'utilité du fichier ~/.profile ? (2pt)
Script d'initialisation exécutée à chaque ouverture d'un nouveau bash

D- Soit la commande suivante dans un script bash :
getopts ":la:-" opt
1- Quelles sont les options qu'on peut lire dans ce script? *(4pt)*
- **l'option simple -l sans valeur**
- **l'option simple -a avec valeur**
- **toute option étendue qui commence par -- (ex. --help)**

2- C'est quoi "opt" ? *(1pt)*
c'est la variable qui va recevoir les options lues par getopts

2ème partie : interprétation

A- Quel serait le résultat à l'affichage après l'exécution du script suivant: (3pt)

```
#!/bin/bash
var1=`(var2=101; echo "5*$var2")`
var3=${var2:-"$var1.$var1"}
echo "${var1}.$var3"
```

résultat : 5*101.5*101.5*101

B- Nous avons un dossier "rep" qui contient 14 fichiers nommés fich01, fich02, … ,fich13, fich14; Nous avons aussi un script shell test.sh executable et contient le code suivant :

```
#!/bin/bash
echo "$1--$2--$3"
```

- Quel serait le contenu du fichier f après l'exécution de la ligne suivante : (4pt)

```
$ ls rep | grep -v "1" | tee f | xargs -n 3 test.sh  >> f
```

explication: on liste les 14 fichiers, on ignore ceux qui contiennent le chiffre "1", il nous reste de fich02 à fich09 qui seront enregistrés par tee dans le fichier f, ensuite ils seront transmis à xargs qui va les donner 3 par 3 au script test.sh. Ce dernier va les afficher dans le format "$1--$2--$3" et le résultat sera ajouté à la fin du fichier f. puisque le nombre de fichiers n'est pas un multiple de 3, xargs donnera seulement 2 arguments à test.sh à la dernière itération. Voici le résultat final dans le fichier f :

```
fich02
fich03
fich04
fich05
fich06
fich07
fich08
fich09
fich02--fich03--fich04
fich05--fich06--fich07
fich08--fich09--
```