TD2 - ISS

Exercice 1.

1.1

Ce programme renvoie le résultat de la somme de deux entiers. Dans le cas de 22 et 34, l'affichage résultant est "Le résultat est : 56".

1.2

Parce que le nom du programme compte comme paramètre et est sauvegardé dans la variable `argv[0]`.

1.3

- Le code source est le fichier .c sans avoir être compilé.
- L'exécutable est le fichier resultant de la compilation du code
- Un processus est le programme en cours d'execution.

Exercice 2.

2.1

1. Bash découpe les mots en utilisant la variable système $IFS (l'espace par défaut).
2. Bash effectue les substitutions (alias, ...).
3. Bash exécute les commandes qui sont dans la forme $() dans les paramètres.
4. Expansion des métacaractères.
5. Effacement des guillemets des expressions.
6. Pipelines.
7. Exécution de la commande.

2.2

Le processus init\systemd, les processus intermediaires, le processus terminal, le processus shell et finalement le processus correspondant au éxécutable.

S'on commence à conter depuis le processus de l'emulator de terminal alors on à 3.

2.3

En utilisant les PID, Process ID, qui sont uniques en space mais pas en temps (ils peuvent être reutilises.).

2.4

2.5

La commande qui permet de lancer un script est `bash nom_du_script` ou tout simplement `./script.sh` dans ce dernier cas seulement si le script à un shebang.

13 processus sont alors crées, 1 pour le script lui-même, 1 pour le mkdir exterieur à la boucle et 11 dans la boucle.

2.6

Oui, tout à fait avec l'option `-p` de mkdir, juste par simplicité.

2.7

Étant donné que les built-in ne spawnent pas de nouveaux processus, et seuelement `ls` et `cat` ne sont pas des built-in, alors 22 processus + le script lui même = 23.

Exercice 3.

3.1

```
$ bash /home/moi/monProg
```

3.2

`$ bash ../monProg`

3.3

Si on explicite la commande `./monProg`, alors Bash essaie d'exécuter l'instance située dans le répertoire courant. Mais si seulement `monProg` est spécifié, alors Bash l'ignore et commence à chercher dans le PATH. L'instance qui va s'exécuter sera donc celle située dans `/sbin`.

3.4

Comme les répertoires sont sépares en utilisant `:` alors on a que:

1. Uniquement dans la session courante du shell.

`$ PATH=/path/to/insert:$PATH`

2. Pour tous les shell.

`$ export PATH=/path/to/insert:$PATH`

3.5

Non, parce que on essaie de lancer uniquement `monProg` donc faut ajouter un `./` pour que ça soit celui du répertoire actuel.

Exercice 4.

4.1

Il y a trois erreurs dans ce script, d'abord, manque les double crochets dans la condition du if, faut capturer le retour du programme et en plus faut signaler que c'est dans le répertoire courant, donc, le `./`

Le script correct est:

```
for num in {1..100} ; do
    if [[ $(./testPair $num) ]] ; then
        echo -n "$num "
    fi
done
echo
```

4.2

Le script affiche tous les numéros pairs de 1 à 100 compris.

4.3

///

4.4

```
for num in {1..100} ; do
    res=$(./testPair $num)
    if test $res -eq 0; then
        echo -n $num
    fi
done
```

4.5

Déjà fait mais bon, `[ exp ]` est obsolette et est mieux d'utiliser `[[ exp ]]`

```
for num in {1..100} ; do
    res=$(./testPair $num)
    if [ $res -eq 0 ]; then
        echo -n $num
    fi
done
```