PROGRAMMATION CONCURRENTE

CPE Lyon – 3ETI

TRAVAUX PRATIQUES SEANCE 1 – REVISION PROGRAMMATION PYTHON

EXERCICE 1 - ARGUMENTS de la ligne de commande – chaînes de caractères

❶ La variable **`sys.argv`** contient les arguments de la ligne de commande, sous forme d'une liste dont le premier élément est le nom du script lancé. La valeur **`sys.argv[0]`** est une chaine de caractères contenant le nom du script, **`sys.argv[1]`** est une chaine de caractères contenant le premier argument, … etc.

Recopiez le script suivant et testez-le avec la ligne de commande suivante :

```
$ python exercice1.py python un deux 3
import sys
print("Nom du programme : ", sys.argv[0])
print("Nombre d'arguments : ", len(sys.argv)-1)
print("Les arguments sont : ")
for arg in sys.argv[1:] :
    print(arg)
```

Dans cet exemple, le script **`exercice1.py`** est lancé avec **4** arguments.
Les **5** paramètres sont reçus à partir de la ligne de commande (les arguments + le nom du script).

❷ Ecrivez un script (**`miroir.py`**) qui prend en paramètre une chaîne de caractères et l'affiche à l'envers.

Par exemple :

```
$ python miroir.py trace
> ecart
```

❸ Modifiez le script **`miroir.py`** (créez le fichier **`miroir2.py`**) pour qu'il traite plusieurs arguments. Par exemple :

```
$ python miroir2.py ecart DNA
> trace AND
```

Exercice 2 – utilisation des arguments de la ligne de commande

Ecrire un programme (**`moyenne.py`**) qui calcule et affiche la moyenne d'un ensemble de notes (nombres entiers) passées en arguments sur la ligne de commande. Le résultat doit être affiché sous la forme :

Moyenne = résultat

La moyenne sera affichée tronquée à **2** décimales de précision.

Cahier des charges complet

Vérifier que :

- Au moins une note est passée en argument. Si cette première vérification échoue, on affichera le message « **`Aucune moyenne à calculer`** ».
- Chaque note doit être comprise entre **0** et **20** (bornes incluses). Si cette vérification échoue, on affichera à l'écran : **Note(s) non valide(s)**
  - Si toutes les notes sont valides, on affichera sur la ligne de commande :

    `Moyenne = <valeur>`

    <valeur> est la valeur de la moyenne.

Exemples de sorties attendues :

```
$ python moyenne.py 10 15 15
> Moyenne est : 13.33

$ python moyenne.py 8 7 12 15 3
> Moyenne est : 9.00

$ python moyenne.py 8 maison 4
> Note non valide

$ python moyenne.py 4 8 -1 7
> Note non valide
```

Note : le symbole **$** indique une entrée utilisateur sur la ligne de commande. Le symbole **>** indique un affichage du programme [Ce ne sont pas des caractères à afficher].

Aide

- Avant de développer le code, réfléchissez aux types des arguments de la ligne de commande. Quelle conversion est nécessaire ?
- Pour convertir une chaine de caractères représentant un nombre vers un type entier (ou réel), vous pouvez utiliser les fonctions de conversion suivantes : **`int()`** ou **`float()`**.
- Pour afficher **2** décimales d'un nombre flottant **N**, on pourra utiliser la syntaxe suivante :

```
print("%.2f" %N)
```

Initiation appel fork()

Testez et commenter les deux programmes suivants :

```
import os,sys
N = 10
i=1
while os.fork()==0 and i<=N :
      i += 1
print(i)
sys.exit(0)
```

Combien de « **Bonjour** ! » et de « **Ok** ! » affiche le programme suivant ?
Testez et commenter ce programme.

```
for i in range(4) :
       pid = os.fork()
       if pid != 0 :
           print("Ok !")
       print("Bonjour !")
sys.exit(0)
```

processus en cascade

Écrire deux programmes qui créent **N** processus P_i tels que :

Dans le 1er programme : pour tout **i** entre **2** et **N**, P_i soit le fils de P_{i-1}

Dans le 2ème programme : pour tout **i** entre **2** et **N**, P_i soit le fils de P_1

Chaque processus devra afficher son identifiant et celui de son père.