Exercice 1 – utilisation des arguments de la ligne de commande [Introduction à Python]

Ecrire un script **python** (**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
- à l'écran : **Aucune moyenne à calculer**
- Chaque note est comprise entre **0** et **20** (bornes inclues). 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 script. 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)**, il est possible d'utiliser les fonctions suivantes : **int()**, ou **float()**.
- Pour afficher **2** décimales d'un nombre flottant **x**, on pourra utiliser la syntaxe suivante : **print ("%.2f" % x)**

Exercice 1 – Somme des éléments d'une liste [**Processus** et **tube anonymes**]

Pour calculer la somme des éléments d'une liste **L** à **N** entiers (initialisée aléatoirement) on utilise deux processus qui s'exécutent en parallèle. Le processus **P1** parcourt les éléments d'indice impair et le processus **P2** parcourt les éléments d'indice pair. Le processus père lance les 2 processus **P1** et **P2**. A la fin d'exécution de **P1** et **P2**, le processus père récupère les 2 résultats **R1** et **R2** déposés dans un tube **T** par les deux processus et affiche la somme de R1 et R2.

#Processus P1	#Processus P2
i = 1 SommeImpairs = 0 Tant que (i ≤ N) faire SommeImpairs=SommeImpairs+L[i] i = i + 2 FinTantque Déposer SommeImpairs dans le tube T	i = 0 SommePairs = 0 Tant que (i ≤ N) faire SommePairs := Somme SommePairs + L[i] i = i + 2 FinTantque Déposer SommePairs dans le tube T

Exercice 3 – os.fork() & os.exec()

Ecrire un script **python** qui prend en paramètre une série de fichiers source **.c**, les compile chacun séparément et simultanément puis édite les liens pour produire un exécutable. Ce programme doit :

- Lancer un **processus fils** pour chacun des noms de fichiers passés en paramètre;
- Chaque fils doit exécuter le programme **gcc -c** sur le fichier dont il s'occupe;
- Le père doit **attendre la terminaison** de tous ses fils;
- Si l'ensemble des fils ont terminés sans erreur, le père réalise l'édition de liens en exécutant **gcc** sur les **fichiers .o** produits par les fils.

Exercice 4

Considérons le programme Python suivant :

```
1  import os, time, random, sys
2  for i in range(4) :
3      if os.fork() != 0 :
4          break
5  random.seed()
6  delai = random.randint(0,4)
7  time.sleep(delai)
8  print("Mon nom est " + chr(ord('A')+i) + " j ai dormi "
   + str(delai) +  " secondes")
9  os._exit(0)
```

1. Donnez l'arbre généalogique des processus engendrés par ce programme.
2. Quels sont les affichages possibles à l'écran.
3. Sans modifier les lignes de *2* à *7*, modifiez ce programme de façon à ce que les processus fassent leur affichage par ordre alphabétique inversé du nom.