UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

Info. Soutien1 : fonctions, paramètres

Rendu: (en fin de séance) pour faire une **archive zip** de votre TP pour le rendu, exécutez, dans le terminal :

```
cd ~/info-spichi/
zip -r tpsoutien1.zip tpsoutien1/
```

Mise en place:

Travaillez dans un dossier dédié au TP, et créez un fichier `compte-rendu.txt` pour écrire la date et vos noms, donc par exemple :

```
mkdir ~/info-spichi/tpsoutien1
cd ~/info-spichi/tpsoutien1
emacs compte-rendu.txt &
mkdir ex1
```

NB : si vous écrivez un programme qui prend trop de temps, vous pouvez l'interrompre avec `Ctrl+C`.

Exercice 1 – Parcour des paramètres du programme

Pour cet exercice, travaillez dans un fichier `ex1.py`, directement dans le dossier `tpsoutien1`.

Important Quand on lance `python3 leprogramme.py bla bla2 bla3`, nous lançons `leprogramme.py` en lui passant 3 paramètres. Le programme peut accéder à ces paramètres (qui sont des chaînes de caractères) en utilisant la variable `argv` du module `sys`. Cette variable « `sys.argv` » est une liste de chaînes de caractères qui contient le nom du programme et ses paramètres. Dans l'exemple donné, « `sys.argv` » vaudra `["leprogramme.py", "bla", "bla2", "bla3"]`.

Attention, il faut bien distinguer « paramètres de programme » et « paramètres d'une fonction ».

Q1) Écrivez un programme qui crée une variable `param` et lui affecte comme valeur la liste des paramètres passés en ligne de commande. Par exemple, en exécutant `python3 ex1.py toto 1 3.5` la valeur de `param` doit être `["ex1.py","toto","1","3.5"]`.

'Q2) Quelle est le type des éléments de la liste `param` ?

Q3) Modifiez le programme `ex1.py` pour qu'il affiche ligne par ligne ses paramètres, en parcourant avec une boucle `for` les éléments de la liste `param`.

Q4) Rajoutez au programme `ex1.py` un deuxième affichage ligne par ligne des paramètres, cette fois-ci en parcourant avec une boucle `for` les indices possibles de la liste `param`. Aide : servez-vous des fonctions `range` et `len`

Q5) Rajoutez au programme `ex1.py` un troixième affichage ligne par ligne des paramètres, cette fois-ci en parcourant avec une boucle `while` les indices possibles de la liste `param`.

Q6) (Optionnel) Avec la méthode de votre choix, rajoutez un quatrième affichage des paramètres du programme cette fois-ci dans l'ordre inverse. Aide : utilisez la fonction `range` avec le pas approprié.

Exercice 2 – Fonctions et Conditions

Pour cet exercice, travaillez dans un fichier `ex2.py`, directement dans le dossier `tpsoutien1`. Attention : Ne confondez pas les paramètres du programme avec les paramètres des fonctions.

Q7) Définissez une fonction `produit(a,b)` qui affiche `Appel de la fonction produit` et renvoit le produit de ses paramètres.

'Q8) Que se passe-t-il en exécutant `python3 ex2.py` ?

Q9) Définissez une fonction `somme(a,b)` qui affiche `Appel de la fonction somme` et renvoit la somme de ses paramètres.

'Q10) Le résultat de l'exécution `python3 ex2.py` a-t-il changé ?

Q11) Dans le programme principal, affichez maintenant `Programme principal`.

'Q12) Que se passe-t-il en exécutant `python3 ex2.py` ?

Q13) Dans le programme principal, définissez maintenant deux variables `var1` et `var2` et initialisez-les respectivement aux valeurs `10` et `1.5`.

Q14) Modifiez le programme principal pour qu'il calcule et affiche le produit et la somme des deux variables, en utilisant les fonctions définies dans les questions précendentes.

'Q15) Que se passe-t-il en exécutant `python3 ex2.py` ?

Q16) Toujours dans le programme principal, définissez et initialisez deux nouvelles variables et affichez leur somme et leur produit.

Vous allez maintenant modifier le programme pour qu'il affiche le produit ou la somme des valeurs passées en paramètre en ligne de commande. Par exemple, si on exécute `python3 ex2.py produit 2.2 3`, le programme doit afficher le produit `6.6` et si on exécute `python3 ex2.py somme 2.2 1`, le programme doit afficher la somme `3.2`.

Q17) Modifiez le programme principale pour qu'il récupère les paramètres passés en ligne de commande. Ensuite, faites en sorte qu'il affiche `multiplication` si le premier paramètre est `produit` ou `addition` si le premier paramètre est `somme`.

Q18) Modifiez le programme principale pour qu'il affiche le produit des paramètres si le premier paramètre est `produit` ou la somme si le premier paramètre est `somme`.

'Q19) Qu'est-ce qu'affiche le programme en exécutant `python3 ex2.py somme 0 100` ?

Exercice 3 – Optionnel

Pour cet exercice, travaillez dans un fichier `ex3.py`, directement dans le dossier `tpsoutien1`.

Q20) Définissez la fonction `liste(n)` qui crée et renvoit une liste de taille `n` avec que des `0` dedans.

Q21) Modifiez la fonction `liste(n)` pour que ses éléments ne soient pas des `0`, mais la chaîne de caractères `pair` si l'élément est à une place paire dans la liste ou `impair` sinon.

Q22) Dans le programme principal, créez une liste `liste1` de taille `7` contenant comme éléments la châine de caractères `pair` si l'élément est à une place paire dans la liste ou `impair` sinon.

Q23) Dans le programme principal, créez une deuxième liste `liste2` de taille `7` contenant aussi comme éléments la chaîne de caractères `pair` si l'élément est à une place paire dans la liste ou `impair` sinon. Attention : si on modifie la liste `liste1`, la liste `liste2` ne doit pas changer.

Q24) Définissez une fonction `uns(l)` qui prend en paramètre une liste `l` et qui remplace les éléments qui valent `pair` avec des `1`.

Q25) Dans le programme principal, appelez `uns(liste1)` et affichez ensuite les listes `liste1` et `liste2`.

'Q26) Qu'est-ce qu'il affiche le programme si on execute `python3 ex3.py` ?