UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

Outils Info. TP6: Boucles, Fonctions, Figures

Rendu : sous forme d'archive

À la fin, vos réponses seront rendues dans une archive zip : cette archive contiendra un fichier `compte-rendu.txt` (avec votre **nom**, **groupe**, et les réponses aux questions précédées d'une apostrophe) et vos fichiers pythons.

Pour faire une **archive zip** de votre TP pour le rendu, exécutez, dans le terminal :

```
cd ~/outils-info/
zip -r tp6.zip tp6/
```

Important : mise en place

- travaillez dans un dossier dédié au TP, lui même dans `~/outils-info`,
- vous devez aussi télécharger depuis le site du cours le fichier nommé `ressources-tp6.zip` et le décompresser dans le dossier `tp6` (avec `unzip` ou un outil graphique).

Exercice 1 –Échauffement, tracé et fonction losange

Q1) Copier le programme d'exemple avec `cp debut.py figures.py` et ouvrez `figures.py` pour commencer à en comprendre le contenu.

'Q2) Lancez le programme. Quelle commande avez-vous utilisé ?

'Q3) Que fait ce programme ?

'Q4) D'après le fichier `.py`, que fait le programme qui n'est pas directement visible à l'écran à son exécution ?

Q5) Dans le fichier `figures.py`, écrire une fonction `losange(..)` qui prend en paramètre une fenetre, les coordonnées du centre et deux rayons (horizontal et vertical), et qui trace un losange ayant les diagonales horizontales et verticales, et les dimensions données par les « rayons ».

Q6) Dans la fonction `principale`, (sans enlever ce qui existe déjà), appelez plusieurs fois la fonction losange et faites que le programme sauve l'image dans un fichier `question-losange.png`.

Q7) Modifiez/exécutez votre programme jusque obtenir une image `question-losange.png` qui vous convient. Cette image (ainsi que les autres) fera partie de votre rendu de TP. Au besoin, demandez comment visualiser un fichier image.

Exercice 2 – Paramètres de programme

Q11) Copier le second programme d'exemple avec `cp debut2.py param.py`.

'Q12) Que se passe-t-il quand vous lancez `python3 param.py` ?

'Q13) Que se passe-t-il quand vous lancez `python3 param.py TOTO` ?

Q14) Ouvrez `param.py` et lisez l'explication donnée ci-dessous pour comprendre chaque ligne de `param.py`.

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 ».

'Q15) Quelle commande utiliser pour lancer le programme pour qu'il affiche aussi 5 fois `cool` ?

Exercice 3 – magic

Q16) Créez un fichier `magic-spir.py` pour taper le programme à venir. Ouvrez l'image `modele-magic-spir.png` et considérez la comme un modèle (ou un objectif ou une spécification) pour cet exercice.

Q17) Écrivez dans le fichier python (ici en utilisant la tortue), un programme qui reçoit un nombre entier `n` et un nom de fichier `sortie` (reçus en paramètres dans `argv`) et affiche `n` traits (suivant le modèle) et sauve une image du nom de sortie. Le modèle donné a par exemple été généré avec `python3 magic-spir.py 8 magic-spir.png`.

'Q18) Donnez les commandes que vous utilisez pour lancer votre programme. **Important:** exécutez votre programme avec différents paramètres (en sauvant des images avec des *noms différents*). Les commandes (dans le compte-rendu) et les images constituerons votre réponse.

Exercice 3 (suite, sans la tortue)

'Q19) Répétez les questions « magic », dans un fichier `magic-1.py`, avec comme modèle `modele-magic-1.png` obtenus avec `python3 magic-1.py 4 magic-1.png` … et donnez les commandes utilisées pour générez vos images.

'Q20) Répétez les questions « magic », dans un fichier `magic-2.py`, avec comme modèle `modele-magic-2.png` obtenus avec `python3 magic-2.py 4 magic-2.png` … et donnez les commandes utilisées pour générez vos images.

'Q21) Répétez les questions « magic », dans un fichier `magic-3.py`, avec comme modèle `modele-magic-3.png` obtenus avec `python3 magic-3.py 4 magic-3.png` … et donnez les commandes utilisées pour générez vos images.

'Q22) Répétez les questions « magic », dans un fichier `magic-4.py`, avec comme modèle `modele-magic-4.png` obtenus avec `python3 magic-4.py 4 magic-4.png` … et donnez les commandes utilisées pour générez vos images.

'Q23) Répétez les questions « magic », dans un fichier `magic-5.py`, avec comme modèle `modele-magic-5.png` obtenus avec `python3 magic-5.py 6 10 magic-5.png` (NB: le programme utilise 2 paramètres entiers) … et donnez les commandes utilisées pour générez vos images.

'Q24) Répétez les questions « magic », dans un fichier `magic-6.py`, avec comme modèle `modele-magic-6.png` obtenus avec `python3 magic-6.py 6 10 magic-6.png` … et donnez les commandes utilisées pour générez vos images.

Exercice 4 ...

Q25) Dans un fichier `smiley.py`, en vous inspirant de `figures.py`, créez une fonction `smiley` qui trace un smiley. La fonction doit accepter 5 paramètres : une fenêtre, une couleur (liste de 3 éléments), les coordonnées du centre et un rayon. Si vous êtes en manque d'inspiration, inspirez vous de `modele-smiley.png`.

Q26) Dans la fonction `principale`, appelez plusieurs fois la fonction smiley et faites que le programme sauve l'image dans un fichier `question-smiley.png`.

Q27) Modifiez/exécutez votre programme jusque obtenir une image `question-smiley.png` qui vous convient.

'Q28) Répétez les questions « magic », dans un fichier `magic-smiley.py`, avec comme modèle `modele-magic-smiley.png` obtenus avec `python3 magic-smiley.py 6 10 magic-smiley.png` … et donnez les commandes utilisées pour générez vos images.