

**Boucles for (1)****TP4****Exercice 1 La fonction range**

⌚ 15 min

- Q1** Écrire un programme (pas une fonction...) affichant tous les nombres entiers de 6 à 11 inclus.
- Q2** Écrire un programme affichant tous les nombres impairs entre 101 et 121 (inclus!).
- Q3** Écrire un programme affichant tous les nombres multiples de 5 de 20 à -20 (inclus!). On respectera bien ici le sens de l'affichage souhaité : on part de 20 pour aller à -20!

**Exercice 2 Proche du cours**

⌚ 20 min

- Q1** Écrire une fonction permettant de calculer pour  $n \in \mathbb{N}$  la valeur de  $S_n = \sum_{k=3}^{n+1} \sqrt{2k+1}$ . Vérifiez le résultat de votre fonction en calculant  $S_4$  à la main.
- Q2** Écrire une fonction `fact` qui prend en argument un entier naturel  $n$  et qui renvoie la valeur de la *factorielle* de  $n$  c'est-à-dire le nombre  $1 \times 2 \times \dots \times (n-1) \times n$ . On note ce nombre  $n!$ ; par exemple  $3! = 1 \times 2 \times 3 = 6$ . Testez votre fonction.
- Q3** Écrire une fonction prenant en argument un réel  $x$  et un entier naturel  $n$  et renvoyant la valeur de  $S(x, n) = \sum_{k=0}^n \frac{x^k}{k!}$ . On réutilisera la fonction `fact`. Vérifier que  $S(7, 4)$  vaut environ 189,7.
- Q4** Écrire une fonction prenant en argument un entier naturel  $n$  et renvoyant le terme  $u_n$  de la suite définie par :  $u_0 = 2$  et  $\forall n \in \mathbb{N}, u_{n+1} = 2u_n + \frac{1}{u_n}$ . On vérifiera que  $u_{10}$  vaut environ 2379,37.
- Q5** Même question pour la suite  $(v_n)$  définie par  $v_1 = 1,4$  et :  $\forall n \in \mathbb{N}^*, v_{n+1} = v_n(3 - v_n)$ . On vérifiera que  $v_{10}$  vaut environ 2,1626.

**Exercice 3 Suite et somme**

⌚ 20 min

On considère la suite  $(u_n)_{n \geq 0}$  définie par  $u_0 = 1$  et  $\forall n \in \mathbb{N}, u_{n+1} = 3\sqrt{u_n} + 1$ .

- Q1** Écrire une fonction `suite` qui prend en argument un entier naturel  $n$  et qui renvoie la valeur de  $u_n$ . (Tester votre fonction sur de petites valeurs de  $n$ ).
- Q2** En déduire une fonction `somme`, utilisant la fonction `suite`, qui prend en argument un entier naturel  $n$  et qui renvoie la valeur de  $\sum_{k=0}^n u_k$ . (Tester votre fonction sur de petites valeurs de  $n$ ).
- Q3** Combien de fois Python calcule-t-il la valeur de  $u_1$  pour évaluer `somme(5)` ?
- Q4** Écrire une fonction `somme_bis` qui renvoie le même résultat que `somme` mais sans le défaut évoqué à la question précédente.
- Q5** La fonction `somme_bis` devrait être plus "efficace" que la fonction `somme`. Pour le constater, on peut chronométrer le temps que met Python à faire un calcul. Pour chronométrer le temps d'exécution d'une commande on utilisera la syntaxe suivante :

```
1 import time # À positionner en début de script (une seule fois)
2 T0 = time.time() # stocke dans T0 l'heure de début du calcul
3 # compléter cette ligne avec le calcul à effectuer
4 T1 = time.time() # stocke dans T1 l'heure de fin du calcul
5 print(T1-T0) # affiche la durée du calcul
```

Constater que `somme_bis` est plus rapide que `somme` sur le calcul de  $u_{100}$  ou  $u_{1000}$ .

**Exercice 4** Primalité (temps restant)

Un entier naturel supérieur ou égal à 2 est dit premier lorsque ses seuls diviseurs positifs sont 1 et lui-même. On rappelle que la commande `a%b` renvoie le reste dans la division euclidienne de `a` par `b`, c'est-à-dire que  $r = a \% b$  lorsque  $r \in \llbracket 0, b - 1 \rrbracket$  et que  $a = db + r$  pour un certain entier  $d$ .

**Q1** Écrire une fonction `divisible` prenant en arguments deux entiers  $a$  et  $b$  et renvoyant `True` si  $a$  est divisible par  $b$  et `False` sinon. On prendra soin d'écrire la version la plus simple possible de la fonction.

**Q2** Écrire une fonction `premier` prenant en argument un entier et renvoyant `True` s'il est premier et `False` sinon. Testez votre fonction, en particulier avec la valeur 25.

**Q3** Écrire un programme Python affichant tous les nombres premiers inférieurs à 100.

**Consignes rendu de TP :**

Vous devez rendre ce TP sous **format PDF** dans le dossier de **restitution de devoirs** de votre session personnelle : merci de suivre les indications ci-dessous. Aucun document d'un autre format que PDF et aucun document envoyé par mail ne sera accepté.

Étapes à suivre :

- Dans Spyder, cliquez sur “Tools” dans la barre de menu en haut,
- Choisissez “Preferences”, une boîte de dialogue s'ouvre,
- Dans cette boîte de dialogue, changez le “syntax highlighting theme” en “Minimal”,
- Cliquez sur “OK”,
- Spyder vous demande si vous souhaitez redémarrer le logiciel : cliquez sur “No”,
- Cliquez sur “File”, puis “Print” et choisir “Microsoft print to PDF”,
- Cliquez sur “Imprimer”, une boîte de dialogue s'ouvre pour choisir l'emplacement et le nom du fichier à enregistrer,
- Sélectionnez le dossier “Restitution de devoirs” à gauche de la boîte de dialogue, puis le dossier “TP4 info”
- Modifiez le nom de votre fichier en : “Nom-de-famille\_Prénom\_TP4” et enregistrez-le!