

1 Introduction

Ouvrir le fichier `R1.06-TP1-part1.ipynb`. Lire attentivement les explications et exécuter les cellules en prenant soin de bien comprendre les résultats qui s'affichent.

N'hésitez pas à poser des questions lors de cette prise en main !

2 Structures de contrôle

Ouvrir le fichier `R1.06-TP1-part2.ipynb` et répondre aux questions posées directement dans le notebook.

3 À vous de jouer !

Commencer par créer un nouveau notebook en prenant soin de choisir un noyau (kernel) *sagemath* puis résoudre les exercices suivants en écrivant du code dans les cellules de type `code` et les éventuels commentaires dans les cellules de type `markdown`.

3.1 Divination

Vous allez devoir écrire un programme proposant à l'utilisateur de deviner un entier entre 1 et 10 en trois essais.

1. Commencez par choisir un nombre au hasard. Pour cela, vous pouvez utiliser la fonction `randint()`. Par exemple, la commande `randint(0, n)` retourne un entier choisi aléatoirement compris entre 0 et n (inclus).
2. Puis demandez un entier entre 1 et 10 à l'utilisateur et comparez-le à l'entier aléatoire choisi.
3. Faites en sorte que l'utilisateur ait trois essais. Le programme s'arrête quand il a trouvé ou quand son nombre d'essais est épuisé. Sinon, afficher à chaque fois un message approprié (« trop grand », « trop petit », etc.).
4. Testez votre programme avec diverses valeurs.
5. Que se passe-t-il si l'utilisateur rentre un flottant ?
6. Que se passe-t-il si l'utilisateur se trompe et rentre un nombre qui n'est pas entre 1 et 10 ?
Modifiez votre programme pour qu'il redonne sa chance à l'utilisateur dans ce cas-là.
7. *Bonus* : Et si l'utilisateur se trompe encore ?

3.2 Trois mais pas cinq

Écrire un programme qui prend un entier positif n en entrée, et affiche tous les nombres de 1 à n qui sont divisibles par 3 mais pas divisibles par 5.

3.3 De plus en plus grand

Écrire un programme qui laisse l'utilisateur rentrer autant de nombres positifs qu'il veut, tant que chaque nombre est strictement supérieur à celui d'avant. Si un nombre rentré ne vérifie pas la règle, le programme s'arrête et indique combien de nombres comporte la suite strictement croissante rentrée par l'utilisateur.

3.4 Moyenne

1. Écrire un programme qui permet à l'utilisateur de saisir autant de nombres entiers qu'il veut et qui calcule la moyenne des nombres rentrés. Quand l'utilisateur a terminé, il écrit `stop` au lieu d'un nombre.
2. Modifier le programme pour qu'il redemande les nombres tant qu'ils ne sont pas compris entre 0 et 20.