TP Python N°2 – "Geany" et les instructions

1. Prise en main de l'environnement de programmation “geany”

Jusqu'à présent, nous avons utilisé Python en mode interactif. Ce mode est pratique pour découvrir Python mais rapidement on ressent le besoin de sauvegarder les lignes de code (les instructions) qu'on a saisies dans l'interpréteur de manière à ne pas avoir à tout retaper d'une fois sur l'autre.

Pour cela, on va utiliser l'éditeur de programme *“geany”*.
Avant toute chose, il faut lancer *geany* (soit vous cliquez sur l’icône “ geany ” représentée par une “ petite lampe ”, soit vous tapez la commande suivante dans un terminal geany&). Une fenêtre s'ouvre correspondant à *“geany”*. Dans la figure 1 nous vous expliquons les commandes essentielles que vous utiliserez.

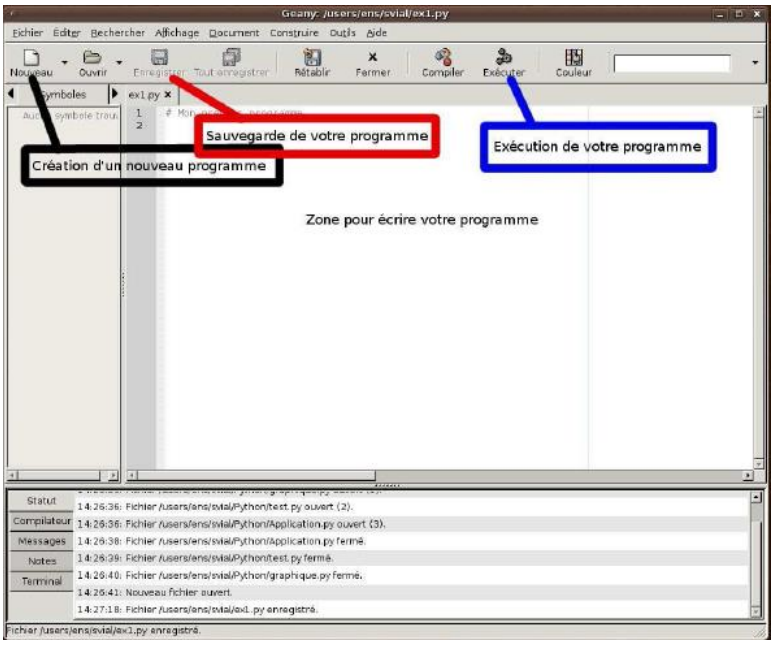

FIG. 1 - L'environnement *Geany*.

Pour créer un nouveau fichier (un nouveau programme), vous pouvez cliquer sur "**Nouveau**" dans la barre de menu. Vous pouvez taper votre programme dans l'éditeur. Pour sauvegarder votre programme, il vous suffit de cliquer sur "**Enregistrer"**. Une fenêtre apparaît alors et vous devez saisir le nom de votre fichier (votre programme) dans l'espace réservé à cet effet. **Pour que *geany* comprenne que vous travaillez dans le langage Python, il faut que le nom de votre fichier se termine par .py**. Par exemple, vous pouvez appeler le premier exercice « ex1.py ». Pour exécuter votre programme, il vous suffit de cliquer sur "**Exécuter"**. Vous êtes maintenant prêts à écrire votre premier programme!

Note: à partir de maintenant vous allez écrire des programmes dans lesquels vous demanderez à l'utilisateur de rentrer des valeurs ou des chaînes de caractères. Pour cela vous allez inviter l'utilisateur à rentrer des caractères au clavier en utilisant la fonction **input**. Cette fonction renvoie une valeur dont le type correspond à ce que l'utilisateur a rentré (entier, réel ou chaîne de caractères entre guillemets, …). Cette valeur sera affectée à la variable que vous aurez choisie.

2. Instructions input et raw_input

Dans certains cas le programme demande à l'utilisateur de répondre à des questions, à des choix … L'utilisateur doit donc rentrer des valeurs ou des chaînes de caractères lors de l'exécution du programme. Cette interaction utilisateur/programme se fait au moyen des instructions **input** et **raw_input**.

Exercice 1 - input

Faire un programme permettant de calculer l'aire d'un rectangle étant données sa largeur et sa longueur, ces deux dernières valeurs étant entrées par l'utilisateur sur la ligne de commande en utilisant ici l'instruction **input**.

Exercice 2 – raw_input

Faire un programme permettant à un utilisateur de saisir deux chaînes de caractères utilisant maintenant l'instruction **raw_input** et affichant à l'écran :

- la première chaîne et sa longueur,
- la seconde chaîne et sa longueur,
- le nombre de fois où la seconde chaîne apparaît dans la première.

Exercice 3 – raw_input et révisions sur les chaînes de caractères

Ecrire un programme permettant à un utilisateur de saisir une séquence nucléotidique et d'afficher :

- le pourcentage en A, en T, en G et en C,
- ainsi que le pourcentage de caractères qui ne font pas partie de l'alphabet nucléotidique (qui correspondent soit à des erreurs de saisie, soit à des indéterminations après séquençage).

On fera en sorte que le programme fonctionne quelque-soit la casse utilisée pour saisir a, t, g, ou c (**la casse** correspond à la façon dont les mots sont écrits : majuscule ou minuscule).

3. Instruction if et blocs d'instructions

L'instruction conditionnelle est l'un des deux fondements de la programmation. En python cette instruction se note **if**. On va introduire les if via une série d'exercices qu'on sauvegardera dans autant de fichiers différents.

Note : A partir de maintenant, on se servira du mode interactif exclusivement pour tester des instructions qu'on ne maîtrise pas. Dans tous les autres cas, on sauvegardera la suite des instructions dans un fichier. Comme on va dès à présent écrire des programmes de plus en plus longs et complexes, il est utile de commenter son code, ne serait-ce que pour savoir, sans lire les instructions, ce qu'il est censé faire. On utilisera pour cela le **caractère #** qui indique à l'interpréteur Python d'ignorer (de ne pas considérer) tout ce qui le suit (jusqu'au saut de ligne).

Exercice 4 – if:

Ecrire un programme permettant à un utilisateur de saisir deux chaînes caractères et de savoir si la seconde est contenue dans la première.
On fera deux versions de ce programme : l'une avec **count**, l'autre avec **in**.

Exercice 5 - if: ... else:

Ecrire un programme permettant de savoir si un nombre entier saisi par l'utilisateur est pair ou impair.

Aide : % est l'opérateur modulo (le reste de la division entière).

Exercice 6 - if: ... elif: ... else:

Ecrire un programme permettant à un utilisateur de saisir un nombre et indiquant si ce nombre est positif, négatif ou égal à zéro.

Exercice 7 - Extraction d'une partie d'une chaîne

7.1. Ecrire un programme permettant à un utilisateur de saisir une chaîne ainsi que d'extraire de cette chaîne les *n* premiers caractères (encore une fois la valeur de *n* doit être saisie par l'utilisateur),

7.2. Ecrire un programme permettant à un utilisateur de saisir une chaîne ainsi que d'extraire de cette chaîne les *n* derniers caractères de la chaîne saisie (via la saisie de *n* par l'utilisateur)

7.3. En utilisant les exercices précédents écrire un programme **donnant le choix à l'utilisateur** de traiter un des deux cas précédents. On utilisera dans ce cas une instruction du type "**if: … elif: … else:**"