TP Info 3 – Initiation à Python et IPE

D.Malka – MPSI 2017-2018 – Lycée Saint-Exupéry

En classe préparatoire, les algorithmes seront implémentés en langage Python 3. Python est un langage de haut-niveau à la syntaxe très légère et au typage dynamique ce qui le rend idéal pour une initiation à la programmation. Nous utilisons la distribution Pyzo qui inclut, entre autres :

— Python 3
— numpy : bibliothèque de fonctions pour le calcul numérique
— scipy : bibliothèque de fonctions pour le calcul scientifique
— matplotlib : bibliothèque de fonctions graphiques
— l'environnement de développement intégré (l'IDE) IPE

Ce premier TP consiste en une familiarisation avec IPE et un premier contact avec le langage Python.

1 Vue d'ensemble de IPE

Voir fig.1
On peut écrire et exécuter du code via :

— la console interactive : le code est écrit à l'entrée `[IN]` ; le résultat du code est affiché à la sortie `[OUT]`. Le script ne peut pas être sauvegardé.
— l'éditeur : il permet d'écrire des programmes et de les enregistrer dans un fichier. Le résultat de l'exécution est affiché dans la console interactive.

2 Utilisation via la console interactive

Dans un premier temps, nous n'utilisons que la console interactive.

2.1 Python comme calculatrice

2.2 Opérations simples

Python peut réaliser des calculs simples avec les opérateurs classiques suivant :

+	addition
-	soustraction
*	multiplication
/	division décimale
//	division euclidienne
%	modulo (reste de la division euclidienne)
**	exponentiation

Tester différents calculs mettant en jeu ces opérateurs. Les priorités des opérateurs sont celles des mathématiques.

2.2.1 Fonctions mathématiques prédéfinies

Pour pouvoir utiliser les fonctions mathématiques, il faut importer la bibliothèque `math` :

```
1 import math as m
```

On peut alors appeler une fonction mathématique de la façon suivante :

```
1 m.cos(3.14)
```

Les noms des fonctions sont intuitifs : `log`, `exp`, `sin`, `sqrt`...

Réaliser différents calculs mettant en jeu des fonctions.

2.3 Python comme grapheur

Python peut être utiliser comme grapheur.

Pour pouvoir définir des listes de points puis des fonctions s'appliquant à chaque point de la liste, il faut importer la bibliothèque `numpy` :

```
1 import numpy as np
```

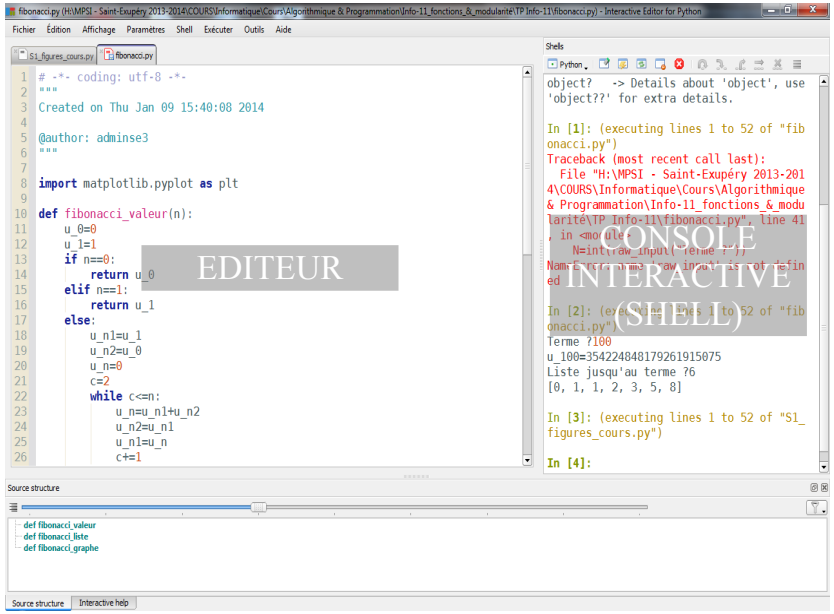

FIGURE 1 – L'environnement de développement IPE. Il comprend entre autres une console interactive et un éditeur de script.

Pour pouvoir définir des listes de points, on peut utiliser la fonction `linspace` de la bibliothèque `numpy` :

```
x=np.linspace(0,1,101)#liste de 101 nombres equireparties entre 0 et 1
```

ou bien `arange` :

```
x=np.arange(0,100,2)#liste de nombres entre 0 et 100 (exclu) par pas de 2
```

Pour calculer les images de x par une fonction :

```
y=np.sqrt(x)#liste des 100 images de x par la fonction racine
```

Pour créer et afficher le graphe, il faut d'abord importer le module `pyplot` de la bibliothèque `matplotlib` :

```
import matplotlib.pyplot as plt
```

Pour créer et afficher le graphe de `y` en ordonnée de `x`, on utilise la fonction `plot` :

```
plt.plot(x,y)
```

Afficher la courbe représentative de la fonction exponentielle sur l'intervalle $[1, 10]$.

3 L'éditeur de script

L'éditeur de script permet d'écrire des programmes, de les exécuter et de les sauvegarder dans des fichiers d'extension `.py`. Il est indispensable pour les programmes longs.

```
def exp_rapide(k,n):
    if n==0:
        return 1
    elif n==1:
        return k
    else:
        if n%2==0:
            return exp_rapide(k*k,n/2)
        else: #n est impair
            return k*exp_rapide(k*k,n/2)#n/2 division euclidienne
                donc tronque n/2


k=float(input("Entrez un nombre : "))
n=int(input("Entrez un entier naturel positif : ") )

puissance=exp_rapide(k,n)
print("{}^{}={}".format(k,n,puissance))
```

FIGURE 2 – Un exemple de programme

Une ligne commençant par une dièse # est ignorée. Ces lignes servent à commenter le code.

```
File "H:\MPSI - Saint-Exupery 2014-2015\Cours\Informatique\
    Architecture des Machines\Info3 - Environnement de
    developpement\exponentiation_rapide_recursive.py", line 15
    return exp_rapide(k*k,n/2)
         ^
IndentationError: expected an indented block
```

FIGURE 3 – Un exemple de message d'erreur

3.1 Programme "Hello world"

Ecrire un programme qui affiche *Hello world* à l'aide de la fonction `print`. Les chaînes de caractères s'écrivent entre guillemets "".

A l'aide des fonctions `input` et `print`, écrire un programme qui demande le prénom `prenom` de l'utilisateur et qui affiche un message lui souhaitant le bonjour. Sauvegarder ce programme.

3.2 Graphe des fonctions trigonométiques

Ecrire un programme qui affiche sur le même graphe les fonctions cosinus et sinus sur l'intervalle $[-\pi, \pi]$. Sauvegarder ce programme. Pour créer une interface graphique on utilise la fonction `figure()` du module bibliothèque `matplotlib.pyplot`. Pour afficher le graphe, on utilise la fonction `show()` du module `matplotlib.pyplot`.

4 Apprendre à utiliser la documentation Python

On trouvera sur le web les documentations de n'importe quelle bibliothèque. On trouve également des cours de Python tel que le livre de Gérard Swinnen dont le format pdf est disponible sur le site de la classe.

La fonction `help` renseigne sur la syntaxe et l'implémentation d'une fonction.

Rechercher dans la documentation de `matplotlib.pyplot` comment modifier l'intervalle de représentation axes d'un graphe (fonction `xlim`), comme ajouter des étiquettes à l'axe des ordonnées (`ylabel` et à l'axe des abscisses (`xlabel`). Modifier le programme précédent afin qu'il affiche les courbes avec $y_{max} = 2$ et $y_{min} = -2$, la courbe représentative du cosinus en trait bleu continu, la courbe représentative du sinus en tirets rouges, x en abscisse, y en ordonnée.

```
Help on built-in function min in module builtins:

min(...)
    min(iterable[, key=func]) -> value
    min(a, b, c, ...[, key=func]) -> value

    With a single iterable argument, return its smallest item.
    With two or more arguments, return the smallest argument.
```

FIGURE 4 – Retour de la commande `help` pour la fonction `min`

5 Déboguer les programmes

5.1 Le débogueur de l'IDE

Les IDE contiennent aussi des outils permettant de deboguer les programmes c'est-à-dire rechercher les erreurs. Nous apprendrons à les utiliser au fur et à mesure de l'année.

5.2 Python Tutor

`www.pythontutor.com` (fig.5) est une application en ligne exécutant du code Python pas à pas en montrant schématiquement l'évolution de l'état du programme pendant son déroulement. Il est très pratique pour comprendre le fonctionnement d'un programme et éventuellement rechercher une erreur plus subtile qu'une simple syntaxique (qui serait renvoyée par l'interpréteur).

En exécutant le programme fig.6 sur `www.pythontutor.com`, chercher à comprendre ce qu'il fait et comment il le fait.

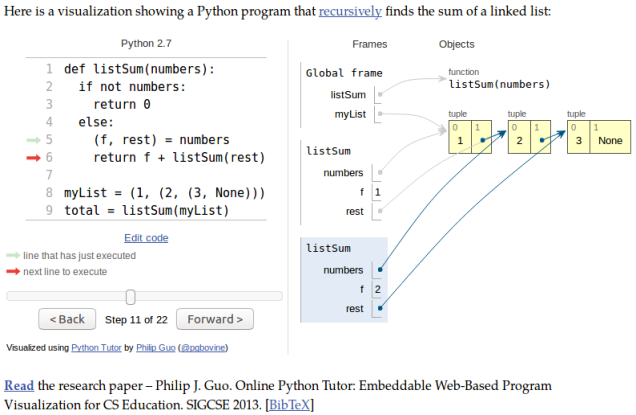

FIGURE 5 – Python Tutor

```
1  def rech_min(T,x):
2      n=len(T)
3      m=T[0]
4      for i in range(1,n):
5          if T[i]<m:
6              m=T[i]
7      return m
8
9  tab=[2,8,9,10,25,30,6,14,0,5]
10 mini=rech_min(tab)
11 print(mini)
```

FIGURE 6 – Que fait ce programme ?