Faculté des Sciences
Aix*Marseille Université

Master de mathématiques, 1re année

Premier TP

vendredi 19 septembre 2013
T. Coulbois

Documentation :
`http://www.sagemath.org/help.html`
En français :
`http://www.sagemath.org/fr/html/a_tour_of_sage/`
`http://www.sagemath.org/fr/html/tutorial/`
`http://sagebook.gforge.inria.fr/`

1 Avec les nombres (entiers, rationnels, et réels)

♣ `n.factorial()`, `n.factor()`, `n.digits()`, `n.numerical_approx()` ♣

1. Calculer et comparer $\frac{271828182,84}{314159265,35}$ et $\frac{27182818284}{31415926535}$.
2. Calculer 200! puis factoriser en nombres premiers.
3. Calculer les 10000 décimales de π.
4. Donner une valeur approchée de la constante d'Euler à 10 décimales près en utilisant la formule d'Euler-Maclaurin.

2 Arithmétique symbolique et polynômes

♣ `var('i,n')`, `sum`, `var('x')` vs `ZZ['x']` ♣

1. Écrire puis simplifier $1 + 4 + \cdots + n^2$.
2. Calculer
$$\sum_{n=1}^{\infty} \frac{1}{n^2}$$
3. Développer $(1 + x)^{15}$ puis dériver et factoriser le résultat obtenu.
4. Développer $(x^2 + ax + b)^5$. Organiser les termes suivant les puisssances de a. Remplacer la valuer de a par 5. Extraire le coefficient en x^3.

3 Dérivation et intégration

♣ `var('x')`, `f.derivative()`, `f.integral()` ♣

1. Dériver x^{x^x} deux fois puis intégrer deux fois.
2. Calculer $\int_0^{+\infty} e^{-x^2} dx$.

4 Résolution de systèmes et d'équations

♣ `x,y,z = var('x,y,z'), solve` ♣

♠ `P.<x,y,z> = QQ[], P.ideal, I.elimination_ideal` ♠

Soit le système :

$$\begin{cases} x+y+z & = & 1 \\ x-y+z & = & 2 \\ x^2+1/y-1/z & = & 2 \end{cases}$$

1. Le résoudre en utilisant `solve`.
2. Le résoudre en utilisant `P.ideal` et `P.quo` pour un anneaux de polynômes.

5 Listes et tuples

5.1 Les tuples

Définition : un tuple est une suite entourée de (et de).
Syntaxe :`(1,2,3)`, `(f(i) for i in range(n))`
Construire les tuples :

1. S_1 des 20 premiers nombres entiers.
2. S_2 des 20 premiers nombres pairs.
3. S_3 des 20 premiers nombres impairs.
4. S_4 des 20 premiers nombres premiers.
5. S_5 des 20 premiers nombres carrés.
6. S_6 des 20 premiers nombres congrus à 1 modulo 4.

5.2 Les listes

Définition : une liste est une suite entourée de [et de].
Syntaxe :`[1,2,3]`, `[f(i) for i in range(n)]`

♣ `nops`, `op`,`map` ♣

1. Transformer les tuples S_4 et S_5 en des listes L_4 et L_5.
2. Additionner le premier et le troisième élément de la liste L_5.
3. Afficher le nombre d'éléments de la liste L_5.
4. Afficher les trois premiers éléments de L_5 puis les 3 derniers.
5. Construire la liste $L_4 reste$ des restes modulo 4 des éléments de L_4
6. Construire la liste L_1 des éléments b, d, e, f
7. Ajouter l'élément a en tête de liste de L_1. Appeler la nouvelle liste L_2.
8. Ajouter l'élément g en fin de liste de L_2. Appeler la nouvelle liste L_3.
9. Ajouter l'élément c entre le $2^{\text{ème}}$ et le $3^{\text{ème}}$ élément de L_3. Appeler cette liste L_4.
10. Donner le dernier terme de L_4.
11. Inverser les éléments de la liste L_4.

6 Programmation

6.1 Fonctions et boucles

♣ `def`, `if`, `for`, `while` ♣

```
a = 0
for i in range(n):
    a += f(i)
```

```
if bool:
    line a
else:
    line b
```

```
n = 7
while n < 16:
    print n
    n += 1
```

```
def f(x,y):
    if x < y:
        return x
    else:
        return y
```

6.2 Exercises de programmation

1. Les variables a, b, c contiennent trois nombres distincts. Écrire une procédure qui affiche celui des trois nombres qui est compris entre les deux autres.
2. Écrire une procédure qui, étant donnés un objet x et une liste l, teste la présence de x dans la liste l.
3. Un nombre est parfait s'il est égal à la somme de ses diviseurs lui-même exclu. Écrire un algorithme permettant de trouver les nombres parfaits figurant parmi les 500 premiers entiers. [1]

6.3 Utilisation des classes en Python

On donne un exemple plus avancé de l'utilisation des classes d'objets en Python.

Exercise. La classe `NombreMagique` (ci-dessous) est une classe d'éléments d'un anneau. Décrire l'anneau. Changer la classe (en changeant la fonction `__mul__`), pour produire des nombres complexes tel que (a, b) répresent $a + bi$. Changer la fonction `__repr__` pour avoir cette répresentation standard des nombres.

```
class NombreMagique():
    """
    Une classe de tuples (a,b) avec addition et multiplication.
    """
    def __init__(self,ab):
        try:
            ab = tuple(ab)
        except:
            raise ValueError, "ab doit être un tuple."
        if len(ab) != 2:
            raise ValueError, "ab doit être un tuple de longuer 2."
        self._ab = ab
```

1. on connaît beaucoup de nombres parfaits pairs que l'on sait caractériser à partir des nombres premiers de Mersenne premiers par $2^{n-1}(2^n - 1)$ ssi $2^n - 1$ est premier (voir ci-dessous). Par contre personne ne sait s'il existe des nombres parfaits impairs !

```
    def __repr__(self):
        return "(%s,%s)"%self._ab
    def __add__(self,other):
        if isinstance(other,NombreMagique):
            a1,b1 = self._ab
            a2,b2 = other._ab
            return NombreMagique([a1+a2,b1+b2])
        else:
            raise ValueError, "other doit être un NombreMagique"
    def __mul__(self,other):
        if isinstance(other,NombreMagique):
            a1,b1 = self._ab
            a2,b2 = other._ab
            return NombreMagique([a1*a2,a1*b2+a2*b1])
        else:
            raise ValueError, "other doit être un NombreMagique"

x = NombreMagique((7,1)) # utilisant __init__
y = NombreMagique([3,4]) # utilisant __init__
print x + y # ultilisant __add__ et __repr__
print x * y # ultilisant __mul__ et __repr__
```