Faculté des Sciences
Aix*Marseille Université

Master de mathématiques, 1re année

Premier TP

vendredi 4 novembre 2015
T. Coulbois, B. Cadorel

Les nombres de Sage

1 Avec quel précision est codé un nombre réel dans Sage ?

1. Testez les instructions suivantes

```
sage: a=1; a
sage: a.parent()
sage: type(a)
sage: b=1/2+1; b
sage: print b, b.parent()
sage: print type(b)
sage: print a in QQ, b in ZZ
```

```
sage: x=1.0
sage: x.parent()
sage: x.precision()
sage: y=1.0e-20
sage: x+y==x
sage: 1.0+1/3
```

```
sage: phi=(1+sqrt(5))/2
sage: phi.parent()
sage: phi.precision()
sage: phi in RR
sage: (phi+1.0).parent()
sage: phi+y==phi
sage: if phi+y==phi: print "=="
```

2. Décrire l'ensemble `RR` de Sage. Préciser les nombres qu'il contient (vous réfléchirez à la notion de mantisse et d'exposant).

3. Mettre en lumière les limites de cet ensemble en faisant apparaître des résultats paradoxaux.

2 Travailler avec des nombres à précision donnée

```
sage: x=sqrt(3)
sage: x.numerical_approx(100)
sage: R100=RealField(prec=100)
sage: y=R100(x)
sage: x+1/3
sage: y+1/3
```

4. Donner les 20 premières décimales de $\sqrt{2}$ et π

3 Des complexes, des nombres algébriques, etc.

```
sage: z=1/(1+I); z
sage: X=QQ['X'].gen()
sage: P=X^8 - X^7 + X^5 -X^4 + X^3 - X + 1
sage: P.roots(QQbar)
```

5. Calculer dans $\mathbb{C}$: i^i.

4 Programmer : écriture décimale, binaire, etc. des nombres

6. Rappeler la définition de l'écriture binaire (respectivement décimale) d'un nombre entier.

Soit `n` un entier qui s'écrit $\overline{b_n b_{n-1} \cdots b_0}$ en binaire (par exemple 13 s'écrit $\overline{1101}$). Nous utilisons une liste `[1,0,1,1]` pour manipuler cette écriture.

7. Écrire une fonction `bits_to_integer()` qui étant donnée une liste de chiffres binaires renvoie le nombre entier qui admet cette écriture.

```
sage: bits_to_integer([1,0,1,1])
13
```

8. Écrire une fonction `integer_to_bits()` qui étant donné un nombre entier renvoie son écriture binaire.

```
sage: integer_to_bits(13)
[1,0,1,1]
```

9. Définir l'écriture d'un nombre réel en binaire.

10. Comme précédemment écrire des fonctions `bits_to_real(l,m)` où `l` et `m` sont des listes de chiffres binaires respectivement avant et après la virgule et `real_to_bits(x,digits)`, le paramètre `digits` donnant le nombre de chiffres binaires après la virgule souhaités.

11. Étendre votre programme pour écrire un nombre dans n'importe quelle base et réciproquement.

5 Programmation objet

12. Taper le programme ci-contre, l'éxécuter, puis tester les commandes :

```
sage: a=BinaryInteger([1,1,0,1])
sage: b=a+a; b
sage: print a+b
```

13. Écrire une méthode `to_integer(self)` qui renvoie l'entier dont l'écriture binaire est `self`.

14. Écrire une méthode `__mul__(self,other)` qui multiplie deux écritures binaires.

15. Écrire une méthode statique `from_integer()` qui construit l'écriture binaire d'un entier.

```
class BinaryInteger:
     def __init__(self,l):
          self._l=l

     def __repr__(self):
          return str(self._l)

     def __getitem__(self,i):
          if i<len(self._l):
             return self._l[i]
          else:
             return 0

     def __add__(self,other):
          result=[]
          i=0; carry=0
          while i<len(self._l) or i<len(other._l):
              d=self[i]+other[i]+carry
              if d>1:
                  d=d-2; carry=1
              else:
                  carry=0
              result.append(d)
              i+=1
          if carry>0:
              result.append(carry)
          return BinaryInteger(result)
```