```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

#
# fichier: expression_tests.py
# version: 0.5.0
#  auteur: Pascal CHAUVIN
#    date: 2014/10/28
#
# (tous les symboles non internationaux sont volontairement omis)
#

import sys
sys.path.append('../calc_mod')
sys.path.append('../calc_mod/math_mod')
sys.path.append('../calc_mod/math_mod/entier_mod')
sys.path.append('../calc_mod/math_mod/rationnel_mod')
sys.path.append('../calc_mod/math_mod/monome_mod')
sys.path.append('../calc_mod/math_mod/polynome_mod')
sys.path.append('../calc_mod/math_mod/fraction_mod')
sys.path.append('../calc_mod/math_mod/utile_mod')
sys.path.append('../calc_mod/expression_mod')
sys.path.append('../calc_mod/calcul_mod')

import expression as ex

import fraction as frac
import rationnel as ra

def test_unitaire_0(visible =False):
  print("*** expression: test_unitaire_0 ***")

  e = ex.expression("{5 * {x / 3} + [x - 1]}")
  if visible:
    print(e)
    print(repr(e))

  f = frac.fraction()
  if visible: print(f)

  ok = e.est_valide() and f.est_valide()
  return ok



def test_unitaire_1(visible =False):
  print("*** expression: test_unitaire_1 ***")

  a = frac.fraction_depuis_lettre('a')
  if visible: print(a)

  b = frac.fraction_depuis_lettre("b")
  if visible: print(b)

  ok = (b.lire_num().valuation() == ra.rationnel(1))
  return ok



def test_unitaire_2(visible =False):
  print("*** expression: test_unitaire_2 ***")

  n = frac.fraction_depuis_naturel(20 + 1)
  if visible: print(n)

  ok = (n.lire_num().valuation() == ra.rationnel(21))
  return ok
```

```python
def test_unitaire_3(visible =False):
  print("*** expression: test_unitaire_3 ***")

  e = ex.expression("3")
  if visible: print(e)

  ok = (e.lire_valeur().lire_num().valuation() == ra.rationnel(3))
  return ok



def test_unitaire_4(visible =False):
  print("*** expression: test_unitaire_4 ***")

  e = ex.expression("3 * ( 1/2 + 1 )")
  if visible: print(e)

  ok = e.est_valide()
  return ok



def test_unitaire_5(visible =False):
  print("*** expression: test_unitaire_5 ***")

  e = ex.expression("a + b")
  if visible: print(e)

  ok = e.est_valide()
  return ok



def test_unitaire_6(visible =False):
  print("*** expression: test_unitaire_6 ***")

  e = ex.expression("(a + b)/(3-1)")
  if visible: print(e)

  ok = e.est_valide()
  return ok



def test_unitaire_7(visible =False):
  print("*** expression: test_unitaire_7 ***")

  e = ex.expression("(a + b)/(x+x)")
  if visible: print(e)

  ok = e.est_valide()
  return ok



def test_unitaire_8(visible =False):
  print("*** expression: test_unitaire_8 ***")

  e = ex.expression("(a + b)/(x+1)")
  if visible: print(e)

  ok = e.est_valide()
  return ok
```

```
def test_unitaire_9(visible =False):
  print("*** expression: test_unitaire_9 ***")

  e = ex.expression("(a + b)/(x)")
  if visible: print(e)

  ok = e.est_valide()
  return ok



def test_unitaire_10(visible =False):
  print("*** expression: test_unitaire_10 ***")

  a = ex.expression("3 * ( x + 1 ) / (x - 20)")
  if visible: print("a =", a)

  b = ex.expression("x - 1")
  if visible: print("b =", b)

  x = a + b
  if visible: print("a + b = {}\n".format(x))

  x = a - b
  if visible: print("a - b = {}\n".format(x))

  x = a * b
  if visible: print("a * b = {}\n".format(x))

  x = a / b
  if visible: print("a / b = {}\n".format(x))

  a = ex.expression("x + 1")
  if visible: print("a =", a)

  n = ex.expression("4")
  if visible: print("n =", n)

  x = a ** n
  if visible: print("a ** n = {}\n".format(x))

  ok = True
  return ok



def test_unitaire_11(visible =False):
  print("*** expression: test_unitaire_11 ***")

  e = ex.expression("3 / ( (19 + 1) - 20 )")
  if visible:
    print("e =", e)
    e.afficher_erreur()

  ok = (not e.est_valide())
  return ok



def test_unitaire_12(visible =False):
  print("*** expression: test_unitaire_12 ***")

  e = ex.expression("(2+1)/(x*x*x*x*x+1)")
  if visible: print(e)

  ok = e.est_valide()
  return ok
```

```
def test_unitaire_13(visible =False):
  print("*** expression: test_unitaire_13 ***")

  ok = True
  return ok



def test_unitaire_14(visible =False):
  print("*** expression: test_unitaire_14 ***")

  ok = True
  return ok



def test_unitaire_15(visible =False):
  print("*** expression: test_unitaire_15 ***")

  ok = True
  return ok



def test_unitaire_16(visible =False):
  print("*** expression: test_unitaire_16 ***")

  ok = True
  return ok



def test_unitaire_17(visible =False):
  print("*** expression: test_unitaire_17 ***")

  ok = True
  return ok



def test_unitaire_18(visible =False):
  print("*** expression: test_unitaire_18 ***")

  ok = True
  return ok



def test_unitaire_19(visible =False):
  print("*** expression: test_unitaire_19 ***")

  ok = True
  return ok



def test_unitaire_(visible =False):
  print("*** expression: test_unitaire_ ***")

  ok = True
  return ok



def tests_unitaires():
  return (
    test_unitaire_0() and \
    test_unitaire_1() and \
    test_unitaire_2() and \
    test_unitaire_3() and \
    test_unitaire_4() and \
    test_unitaire_5(True) and \
    test_unitaire_6(True) and \
    test_unitaire_7(True) and \
    test_unitaire_8(True) and \
    test_unitaire_9(True) and \
    test_unitaire_10(True) and \
    test_unitaire_11(True) and \
    test_unitaire_12(True) and \
    test_unitaire_13() and \
    test_unitaire_14() and \
    test_unitaire_15() and \
    test_unitaire_16() and \
    test_unitaire_17() and \
    test_unitaire_18() and \
    test_unitaire_19()
  )



if __name__ == "__main__":
  ok = tests_unitaires()
  if ok:
    print("*** expression: tests unitaires OK ***")
```