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

#
# fichier: calcul.py
# version: 0.5.0
#  auteur: Pascal CHAUVIN
#    date: 2014/10/29
#
# (tous les symboles non internationaux sont volontairement omis)
#

import string

import sys
sys.path.append('../expression_mod')

import expression_mod.expression as ex

class calcul(object):

  def __init__(self):
    """ constructeur """
    self.__fichier = open("calculs.txt", "w")
    self.__compteur = 1
    self.__valide = True



  def __str__(self):
    """ - """
    return "(calcul)"



  def __repr__(self):
    """ - """
    return "[calcul:\n__valide={}\n]\n".\
      format(self.__valide)



  def est_valide(self):
    """ accesseur """
    return self.__valide



  def exemples(self):
    """ tests automatiques """
    print("*** quelques exemples ***\n")

    e = ex.expression("(3 + 4) * 5 ^ (1 + 1) - 7")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne 168

    e = ex.expression("2 ^ 3 ^ 2 ^ 2")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne 24178516392292583494123 52

    e = ex.expression("-(5 - 1) : 5^3 + ( 9 + 1 ) * ( 7 + 2 * 5 )")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne 169.968

    e = ex.expression("3/7 - 2/7 : ( 5 : 14 )")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne -13/35
```

```
    e = ex.expression("((a + b)/2)^2 - ((a - b)/2)^2")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne a*b

    e = ex.expression("(x+2/3)^(-3)")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne 27/(27*x^3 + 54*x^2 + 36*x + 8)

    e = ex.expression("(8*(16/10)*10^8)/((4/10)*10^10)")
    print("exemple de calcul :", e.lire_formule())
    print(e, '\n') # donne 0.32



  def notice(self):
    """ notice """
    print()
    print("*** calc3: programme de calcul formel (corps de fractions) ***")
    print()
    print("                \"Python3 pour le lyc\\'{e}e et la pr\\'{e}pa\"")
    print("                              Alexandre CASAMAYOU-BOUCAU")
    print("                                        Pascal CHAUVIN")
    print("                                     Guillaume CONNAN")
    print()
    print("    (version 0.5.0)")
    print()



  def remarque(self):
    """ remarque """
    print("*** remarque importante ***")
    print()
    print("    Les calculs sont enregistres dans un fichier (en format texte pur)")
    print("    nomme \"calculs.txt\" dans le repertoire courant d'execution du pro-")
    print("    gramme. Il est donc indispensable d'executer le programme \"calc3\"")
    print("    depuis un repertoire ou l'utilisateur possede le droit d'ecriture.")
    print()
    print("    (tous les symboles non internationaux -i.e. non ASCII- sont volon-")
    print("    tairement omis)")
    print()

  def lecture(self):
    """ saisie d'une expression """
    invite = "calc3: {}> ".format(self.__compteur)
    s = str(input(invite))
    self.__fichier.write(invite + s + "\n")
    if len(s) > 0:
      self.__compteur += 1
    return s



  def boucle(self):
    """ boucle d'evaluation """
    print("*** boucle interactive ***")
    print()
    print("    Entrer une expression mathematique a evaluer (ou laisser vide")
    print("    pour finir) puis valider.")
    print()
    entree = self.lecture()
    while len(entree) > 0:
      e = ex.expression(entree)
      print()
      self.__fichier.write(str(e) + "\n\n")
      print(e.lire_valeur().joli())
      print()
      entree = self.lecture()
```

```
    print()
    self.__fichier.close()


  def executer(self):
    """ execution du programme """
    self.notice()
    self.exemples()
    self.remarque()
    self.boucle()


if __name__ == "__main__":
  c = calcul()
  c.executer()
```

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

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

import calcul as calc

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

  c = calc.calcul()
  c.executer()
  if visible: print(c)

  ok = c.est_valide()
  return ok



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

  ok = True
  return ok



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

  ok = True
  return ok



def test_unitaire_3(visible =False):
  print("*** calcul: test_unitaire_3 ***")

  ok = True
  return ok



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

  ok = True
  return ok



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

  ok = True
  return ok



def test_unitaire_6(visible =False):
  print("*** calcul: test_unitaire_6 ***")
```

```
  ok = True
  return ok


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

  ok = True
  return ok


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

  ok = True
  return ok


def test_unitaire_9(visible =False):
  print("*** calcul: test_unitaire_9 ***")

  ok = True
  return ok


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

  ok = True
  return ok


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

  ok = True
  return ok


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

  ok = True
  return ok


def test_unitaire_13(visible =False):
  print("*** calcul: test_unitaire_13 ***")

  ok = True
  return ok


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

  ok = True
  return ok
```

```
def test_unitaire_15(visible =False):
  print("*** calcul: test_unitaire_15 ***")

  ok = True
  return ok



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

  ok = True
  return ok



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

  ok = True
  return ok



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

  ok = True
  return ok



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

  ok = True
  return ok



def test_unitaire_(visible =False):
  print("*** calcul: 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() and \
    test_unitaire_6() and \
    test_unitaire_7() and \
    test_unitaire_8() and \
    test_unitaire_9() and \
    test_unitaire_10() and \
    test_unitaire_11() and \
    test_unitaire_12() 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("*** calcul: tests unitaires OK ***")
```