nphullmen_12xa.py

```
#!/usr/bin/env python3.6
# -*- coding: utf-8 -*-
# Série première. En Général Python
# Objet : Outil pratique
# MusicAtoumic nphullmen_12xa Ce samedi 20 février 2018

"""Pyhton Compulsion Commune
Programme évoluant sur une table de travail
L'histoire des premiers dividendes aux nombres communs
Produisant des multiples à structures uniques
Associant les unités en une ligne composée:
    Structure commune ' Nombre Premier = Premier multiple
    (Un nombre peut composer plusieurs premiers communs)'
La première finalité est alors, liée à une nouvelle fin
    Un nombre peut avoir plusieurs structures communes
    Les structures se chevauchent ou se suivent
Ce qui détermine la fin, c'est l'(ip.sup & dv.inf)
    (dv.inf): L'unité terminale proche du carré
        Soit, elle est un nombre premier
        Soit, elle multiplie les premiers
        Soit, elle voit (Batterie à essais:)
"""

import time

time_ = time.time()


def wifi():
    print('nphullmen_12xa.py En: \n', time.time() - time_)


def zero(z):
    if z == 1:
        z = 0
        print('Premiers (', z, ')ex\n', '[0]')
        print('{} * {} Types {}&{}'
              .format(z, z, z % 6, z % 6))
        wifi()
    else:
        wifi()


""" Batterie à essais:
    Ces nombres sur un alignement de mesures:
        Ligne(#): Entier Dv.inf Break(Type) Temps(s) Multi(|*|)
        Dv.inf premier est signé (°)
        Cette signature (²) indicque sa divisibilité
        |°*°|: Les nombres premiers côtés (ip*dv)"""
# 9325651468789787465281041 4 302367527315767°
# 9137468251595753963214 78 865157831052031219°
# 9878987654565432123 21 34584163467²
#   En Cours ..T loop sqr2/3: 98789876545654321 2321
# 1234567898765432112345678987654321 359223300970873 79²
#   En Cours ..T loop sqr3/6: 1234567898765432112345678987654321
# 99988877766655566677788899 9 107407407407411²
# 22255713014674722255713014674 7 476012969567993²
#   En Cours ..T loop sqr: 22255713014674722255713014674 7
```

```
# 11122233322211100011222333222111 11210988789011211²
#   En Cours ..T loop sqr: 11122233322211100011222333222111
# 555555555515555555555 43944547535²
# 99988877766655544433322211 37207794324679900879 81°
#   En Cours ..T loop sqr3/3: 99988877766655544433322211
# 99988877766655500011122233344455 13045629262860246199858078 21°
#   En Cours ..T loop sqr: 9998887776665550001112223334445 55
# 22255713012225571301 10000000001²
# 9878987654565432123 17 2536415380787°
# 6532987845121422 1088831307520237°
"""
Chaque nombre a sa finalité
Break ? ; En cours
Break om1>q1 ; Stop des lecteurs (om1&q1)
"""
# 9325651652810414 8426829197°
# 9137468251596321478 1696522141031623°
# 98789875654321232 6174367228395077°
# 123456788987654320 355659398²
# 888777666666777888 945612278²
# 22255301467146747 76478699199817°
# 9998887776665554 105684254²
# 999888777665 11255129²
# 11122233110001321 4331042123°
#   En Cours ..T loop sqr: 11122233110001321
# En Cours 55555555551555555 1278297169° break p1>borne (0s)
# ____________________
# :nombre: objet unique
nombre = 456789215975346
print('Nombre =', nombre, '; typ =', nombre % 6)
# :breakComm: printage communs
breakComm = 0
# :zero: instruction zéro
if nombre == 0:
    zero(breakComm)
elif nombre < 0:
    nombre = abs(nombre)
# :carre: service axial
carre = int(nombre ** .5)
print('carre =', carre)
cartyp6 = [carre]
borne = [0]

hautniveau = []  # Liste premiers
horscourse = []  # Liste communs ip&dv
hautmulti_ = [1]  # Mult.hautniveau
hautcumul_ = []  # Cum.Mult.hautniveau
horspluri_ = [0]  # Plus.horscourse
horsmulti_ = [0]  # Mult.horscourse
horscumul_ = []  # Cum.Mult.horscourse
c12tableau = []  # Liste communs dv
dvinftable = []  # Liste tranches dv
ipsuptable = []  # Liste tranches ip
ipsupcumul = [1]  # Mult.ipsuptable
rapcartime = [0]  # Indice niveau dv

# Lecteurs initiaux
p1 = [7]  # Intro +6 type 1
p5 = [11]

ed1 = [0]  # Mi inf -6 type 1
```

```
ed5 = [0]
em1 = [0]  # Mi inf +6 type 1
em5 = [0]

d1 = [0]  # Mi carre -6 type 1
d5 = [0]
m1 = [0]  # Mi carre +6 type 1
m5 = [0]

od1 = [0]  # Mi sup -6 type 1
od5 = [0]
om1 = [0]  # Mi sup +6 type 1
om5 = [0]

q1 = [0]  # Carre -6 type 1
q5 = [0]

p0 = [0]  # Portion vide


def forme15(rng_, r6_, mng_, m6_):
    """Positionnement des Lecteurs (bas/hauts)
        P1: Point (+6) Type (1)
            Position inférieure :ip: Inf.carre
        Q1: Point (-6) Type (5)
            Position supérieure :dv: Sup.carre"""

    # Max = rng_ | Axe = mng_
    dom_ = mng_ // 2  # Coaxe = dom_
    edm_ = mng_ - dom_  # Report inf = edm_
    odm_ = mng_ + dom_  # Report sup = odm_
    e6_ = edm_ % 6  # Type inf e6_
    o6_ = odm_ % 6  # Type sup o6_

    if r6_ == 1:
        q1[0] = rng_
    else:
        dfo_ = r6_ - 1
        q1[0] = rng_ - dfo_

    if m6_ == 1:
        m1[0] = mng_
        d1[0] = mng_
    else:
        dfo_ = m6_ - 1
        m1[0] = mng_ - dfo_
        d1[0] = mng_ - dfo_
    if e6_ == 1:
        ed1[0] = edm_
        em1[0] = edm_
    else:
        dfo_ = e6_ - 1
        ed1[0] = edm_ - dfo_
        em1[0] = edm_ - dfo_
    if o6_ == 1:
        od1[0] = odm_
        om1[0] = odm_
    else:
        dfo_ = o6_ - 1
        od1[0] = odm_ - dfo_
        om1[0] = odm_ - dfo_
```

```
181
182     # Position P5 (début)
183     p5[0] = p1[0] - 2
184     # Position E5 (bas)
185     ed5[0] = ed1[0] - 2
186     em5[0] = em1[0] - 2
187     # Position M5 (milieu)
188     m5[0] = m1[0] - 2
189     d5[0] = d1[0] - 2
190     # Position O5 (haut)
191     od5[0] = od1[0] - 2
192     om5[0] = om1[0] - 2
193     # Position Q5 (fin)
194     q5[0] = q1[0] - 2
195     print(' f15=0; p1=%s; ed1=%s; q1=%s' % (p1[0], ed1[0], q1[0]))
196     print(' f15=0; d1=%s; od1=%s' % (d1[0], od1[0]))
197
198
199 def compare(c):
200     """Réception et traitement (c)"""
201     if c < 1:
202         return
203     hautmulti_[0] = 1
204     didi = [c]
205     print('intro DIDI', didi)
206     con = 1
207     if c not in horscourse:
208         if carre >= c > 1:
209             horscourse.append(c)
210             horscourse.sort()
211         sqc = int(c ** .501)
212         # Démultiplications
213         for dd in range(sqc, max(hautniveau), -1):
214             con += 1
215             if not c % dd or not c % con:
216                 # Identifier (dd&con)
217                 if not c % dd:
218                     dddv = c // dd
219                     print('**dd', dd, dddv)
220                 else:
221                     dddv = c // con
222                     dd = con
223                     print('**con', con, dddv)
224                 # :dd: Explore
225                 if dd not in horscourse \
226                         and dd % 6 in (1, 5) \
227                         and dd not in didi:
228                     print('**_____C app(dd)', dd)
229                     didi.append(dd)
230                     didi.append(dddv)
231                     print('DIDI dd', didi)
232                     print('DD DDDV', dd, dddv)
233                     break
234                 # :dddv: Explore
235                 if dddv >= carre:
236                     sqdd = int(dddv ** .4)
237                     noc = 1
238                     for fd in range(sqdd, max(hautniveau), -1):
239                         noc += 1
240                         # Condition de fin (noc&fd)
241                         if noc > fd:
```

```
                                print('boucle: dddv>carre:', noc)
                                break

                            elif not c % fd or not c % noc:
                                # Identifier (fd&noc)
                                if not c % fd:
                                    dddv = c // fd
                                    print('**fd', fd, dddv, 'sqdd', sqdd)
                                else:
                                    dddv = c // noc
                                    fd = noc
                                    print('**noc', noc, dddv)
                                # :fd: Explore
                                if fd not in horscourse \
                                        and fd % 6 in (1, 5) \
                                        and fd not in didi:
                                    print('**_____C app(fd)', fd, 'sqdd',
sqdd)
                                    didi.append(fd)
                                    print('DIDI fd', didi)
                                    break
                        if dddv not in horscourse \
                                and dddv % 6 in (1, 5) \
                                and dddv <= carre \
                                and dddv not in didi:
                            print('**_____C app(dddv)', dddv)
                            didi.append(dddv)
                        sqd = int(dddv ** .501)
                        if dddv > carre:
                            sqd = int(sqd ** .6)
                            # print('C sqd', sqc)
                        for vv in range(2, sqd):
                            if not dddv % vv:
                                vvd = dddv // vv
                                if vv not in horscourse \
                                        and vv % 6 in (1, 5) \
                                        and vv not in didi:
                                    print('**_____C app(vv)', vv)
                                    didi.append(vv)
                                    break
                                if vvd not in horscourse \
                                        and vvd % 6 in (1, 5) \
                                        and vvd not in didi:
                                    sqv = int(vvd ** .5)
                                    ffd = []
                                    for q in range(2, sqv):
                                        if not vvd % q and vvd not in didi:
                                            veq = vvd // q
                                            ffd.append(q)
                                            ffd.append(veq)
                                            print('** veq =', veq, q)
                                            break
                                    if ffd:
                                        print('** ffd =', ffd)
                                        for f in ffd:
                                            vvd = f
                                            if vvd not in dvinftable and vvd
not in didi:
                                                print('**_____C app(vvd1)',
vvd)
                                                didi.append(vvd)
```

```
                                        break
                                else:
                                    if vvd not in dvinftable:
                                        dvinftable.append(vvd)
                                        print('**_____C app(vvd0)', vvd)
                                        didi.append(vvd)
                                        break
                        didi.sort()
                        print('fin DIDI', didi)
                        # Priorité :didi:
                        break

            for di in sorted(didi):
                c = di
                for haut in hautniveau[1:]:
                    if not c % haut:
                        break
                else:
                    print('****DIDIDI', c)
                    if c <= carre:
                        hautniveau.append(c)
                    elif c not in dvinftable:
                        print('****DIDIDI dvinftable', c)
                        dvinftable.append(c)
            # Production haut niveau
            for hha in hautniveau:
                hautmulti_[0] *= hha
                if hautmulti_[0] not in hautcumul_:
                    hautcumul_.append(hautmulti_[0])

            if hautmulti_[0] != 0:
                print('  Oniveau =', sorted(hautniveau))
                print('  Omulti_ =', hautmulti_[0])
                for c1 in sorted(hautniveau):
                    for c2 in horscourse:
                        c12 = c1 * c2
                        # Séparation :carre:
                        if not nombre % c12 \
                                and c12 not in horscourse:
                            if c12 <= carre:
                                horscourse.append(c12)
                                horscourse.sort()
                            elif c12 not in c12tableau:
                                c12tableau.append(c12)
                                c12tableau.sort()
                print('C Orscourse =', max(horscourse))
                if c12tableau:
                    for ctab in c12tableau:
                        iptab = nombre // ctab
                        # :ctab: Occurence :c12:
                        if iptab not in horscourse:
                            for hh in hautniveau[1:]:
                                if not iptab % hh:
                                    break
                            else:
                                print('C c12 compare: iptab =', iptab)
                                compare(iptab)

                if hautmulti_[0] > 0:
                    horspluri_[0] = 0
                    horsmulti_[0] = 1
```

```
                # Production hors course
                for hho in horscourse:
                    if horspluri_[0] <= nombre:
                        horspluri_[0] += hho
                        horsmulti_[0] *= hho
                horscumul_.append(max(horscourse))
                cartyp6[0] = int((nombre // max(horscourse)) ** .5)
                brn = int(cartyp6[0] ** .9478)
                borne[0] = brn

            # Terme commun
            if max(horscourse) <= carre:
                if hautmulti_[0]:
                    hdv = hautmulti_[0] * hautniveau[1]
                    # :hdv: Son.carre(dv) par Min.hautniveau
                    if hdv >= carre and not nombre % hdv:
                        hip = nombre // hdv
                        # :hip: Son(ip) du Son.carre(dv)
                        if hip not in horscourse:
                            print('Cm compare: hip =', hip)
                            compare(hip)

            if hautmulti_[0]:
                # Termes racine
                if hautmulti_[0] == nombre:
                    hautmulti_[0] = 0
                    print('E BREAK multi_==', nombre)
                elif hautmulti_[0] * hautniveau[1] >= carre \
                        and hautmulti_[0] == max(hautniveau):
                    hautmulti_[0] = 0
                    print('E BREAK multi*min(c)', hautmulti_)
                elif hautmulti_[0] * hautniveau[1] >= nombre:
                    hautmulti_[0] = 0
                    print('E BREAK multi*min(n)', hautmulti_)
                elif (nombre // hautmulti_[0]) <= max(hautniveau):
                    hautmulti_[0] = 0
                    print('E BREAK n/multi <= max(haut)', hautmulti_)
                elif max(horscourse) == nombre:
                    hautmulti_[0] = 0
                    print('E BREAK hors(nombre)', nombre)
                elif max(horscourse) ** 2 > nombre:
                    hautmulti_[0] = 0
                    print('E BREAK hors**2', nombre)
                elif hautmulti_[0]:
                    # Différentiel comparatif
                    dvinf = nombre // max(horscourse)
                    ipsup = dvinf // max(horscourse)
                    sqinf = int(dvinf ** .501)
                    if max(horscourse) not in ipsuptable:
                        ipsuptable.append(max(horscourse))
                        ipsupcumul[0] *= max(horscourse)
                        print('E ipsuptable =', ipsuptable)
                        if ipsupcumul[0] > carre:
                            print('E ipsupcumul =',
                                  nombre // ipsupcumul[0])
                    print('E dvinf//max(hors) ipsup =', ipsup)
                    print('*****************************')
                    print('*****************************')
                    if dvinf not in dvinftable:
                        dvinftable.append(dvinf)
                        if ipsuptable:
```

```
                                rapcartime[0] = 0
                                rapips = max(ipsuptable)
                                rapdvs = min(dvinftable)
                                sqrdvs = int(rapdvs ** .5)
                                print('.T sqrdvs =', sqrdvs)
                                rapnbr = nombre // (rapdvs // rapips)
                                print('.T rapnbr =', rapnbr)
                                if rapnbr > max(ipsuptable):
                                    print('..T rapips =', rapips)
                                    print('..T rapdvs =', rapdvs)
                                    raprap = rapdvs // rapips
                                    if raprap >= carre:
                                        print('..T rapsup =', raprap)
                                        rapcar = raprap // carre
                                    else:
                                        print('..T rapinf =', raprap)
                                        rapcar = carre // raprap
                                    print('..T rapcar =', rapcar)
                                    if rapcar > sqrdvs:
                                        print('.._T loop sqrdvs', sqrdvs)
                                        rapcartime[0] = sqrdvs
                                    elif max(hautniveau) > rapcar > 1:
                                        if rapnbr > sqrdvs:
                                            rapcartime[0] =
int(sqrdvs**.8)
                                            print('.._T loop rapnbr',
                                                  rapcartime[0])
                                    elif rapcar < 2:
                                        sqrap = int(raprap ** .5)
                                        if sqrap > max(hautniveau):
                                            rapcartime[0] = sqrap
                                            print('.. .T loop sqrap',
                                                  sqrap)
                                        else:
                                            rapcartime[0] = raprap
                                            print('.. .T loop raprap')
                                        print('.._.T rapcartime',
                                              rapcartime[0])
                                    else:
                                        if raprap > carre:
                                            if rapcar < int(sqrdvs ** .5):
                                                rapcartime[0] = int(sqrdvs
** .6)
                                                print('..T loop sqr',
                                                      rapcartime[0])
                                            else:
                                                rapcartime[0] = int(sqrdvs
** .948)
                                                print('..T loop 94',
                                                      rapcartime[0])
                                        else:
                                            rapcartime[0] = int(sqrdvs **
.987)
                                            print('..T else r<c', )
                                        print('.._T else rapcartime[0]',
                                              rapcartime[0])
                                else:
                                    rapcartime[0] = rapnbr
                                    print('...T for/else rapcartime',
                                          rapcartime[0])
```

```
                        if len(hautniveau) > 2:
                            if hautmulti_[0]:
                                if ipsup >= carre:
                                    print('E ipsup dvinf:', dvinf)
                                    if rapcartime[0] > 1:
                                        sqinf = rapcartime[0]
                                        print('.E if rapcartime[0]',
                                              rapcartime[0])
                                    else:
                                        print('.E el rapcartime[0]',
                                              rapcartime[0])
                                        sqinf = int(dvinf ** .5)
                                    rng = sqinf
                                    cartyp6[0] = rng
                                    brn = int(cartyp6[0])
                                    borne[0] = brn
                                    r6 = rng % 6
                                    mib1 = (rng - max(hautniveau)) // 2
                                    print('E ipsup forme1 =', rng)
                                    forme15(rng, r6, mib1, mib1 % 6)
                                else:
                                    if not nombre % sqinf \
                                            and sqinf % 6 in (1, 5):
                                        print(
                                            'E if(compare): sqinf =',
sqinf)
                                        compare(sqinf)
                                    if ipsup < max(horscourse) and
c12tableau:
                                        print('.E ipsup',
                                              ipsup, max(c12tableau))
                                        # hautmulti_[0] = 0
                                    else:
                                        if rapcartime[0] > 1:
                                            sqinf = rapcartime[0]
                                            print('.E if rapcartime[0]',
                                                  rapcartime[0])
                                        else:
                                            print('.E el rapcartime[0]',
                                                  rapcartime[0])
                                            sqinf = int(dvinf ** .5)
                                        rng = sqinf
                                        r6 = rng % 6
                                        mib1 = (rng - max(hautniveau)) //
2
                                        if mib1 < p1[0]:
                                            mib1 = p1[0]
                                            print('E if mib1', p1[0])
                                        print('.E else ipsup forme2 =',
rng)
                                        print('c12tableau', c12tableau,
max(horscourse))
                                        forme15(rng, r6, mib1, mib1 % 6)
                                    for hoip in horscourse[1:]:
                                        if not dvinf % hoip:
                                            hodv = dvinf // hoip
                                            ho6 = hodv % 6
                                            if ho6 in (1, 5) \
                                                    and hodv not in
horscourse:
                                                print(
```

```
                                                    'E (compare): hodv =',
hodv)
                                                compare(hodv)
                                                break
                                            elif hoip > sqinf:
                                                break


    """Bas niveau des premiers:
    Création du haut niveau des nombres premiers
    Lié au bas niveau des nombres premiers
    Composition des hors course"""
    basniveau = [1, 2, 3, 5]
    for i in range(1, 7):
        if not nombre % i and i <= carre:
            def rang(ti, i_):
                if ti == 1:  # Nombre Premier(basniveau)
                    hautniveau.append(i_)
                    horscourse.append(i_)
                else:  # Nombre Commun(basniveau)
                    horscourse.append(i_)


            if i in basniveau:
                rang(1, i)
            else:
                rang(2, i)
    if horscourse:
        # Traitement basique
        for h in horscourse:
            for o in horscourse:
                oh = h * o
                if not nombre % oh and oh not in horscourse:
                    if oh > carre and not nombre % (nombre // oh):
                        if (nombre // oh) not in horscourse:
                            print('Basic compare: n>carre =', oh)
                            compare(nombre // oh)
                            break
                    elif oh <= carre:
                        print('Basic compare: n<carre =', oh)
                        compare(oh)
                        break
        dvhors = nombre // max(horscourse)
        borne[0] = int(dvhors ** .5)
        print('Basic borne[0]', borne[0])

    """Borne & cartyp6
    Borne : Limite en pleine forêt
    Cartyp6 : Limite en plein ciel"""
    if not borne:
        borne = cartyp6
    print('cartyp6=', cartyp6[0], ';borne=', borne[0])
    print('horscourse', horscourse)
    if nombre != 0:
        mib0 = borne[0] // 2
        forme15(borne[0], borne[0] % 6, mib0, mib0 % 6)

    """Hauts niveaux premiers: cartyp6
    Itérer dans l'alignement des nombres premiers
    Condition évolutive des lecteurs"""
    while p1[0] <= cartyp6[0]:
```

```
594
595         """if p1[0] > int(borne[0]):
596             print('    W break p1>borne', p1[0], int(borne[0]))
597             break"""
598         if hautmulti_[0] == 0:
599             print('    W break hautmulti_', hautmulti_)
600             break
601         elif om1[0] > q1[0]:
602             print('    W break om1>q1', om1[0], q1[0])
603             break
604 
605         # Lecteurs en lecture
606         # Lecteurs P: Début (P)
607         if od1[0] == 0:  # Révision nombre
608             print('    W OD1', p1[0], int(borne[0] ** .916))
609             print('    W n%OD1', nombre % p1[0])
610         if not nombre % p1[0] and p1[0] not in hautniveau \
611                 and p1[0] not in horscourse:
612             print('  W____P1', p1[0])
613             compare(p1[0])
614         if not nombre % p5[0] and p5[0] not in hautniveau \
615                 and p5[0] not in horscourse:
616             print('  W____P5', p5[0])
617             compare(p5[0])
618         # Lecteurs
619         if not nombre % q1[0] and q1[0] not in hautniveau \
620                 and q1[0] not in horscourse:
621             print('  W____Q1', q1[0])
622             compare(q1[0])
623         if not nombre % q5[0] and q5[0] not in hautniveau \
624                 and q5[0] not in horscourse:
625             print('  W____Q5', q5[0])
626             compare(q5[0])
627         # Lecteurs M: Entre carre (Q) et début (P)
628         if not nombre % m1[0] and m1[0] not in hautniveau \
629                 and m1[0] not in horscourse:
630             print('  W____M1', m1[0])
631             compare(m1[0])
632         if not nombre % m5[0] and m5[0] not in hautniveau \
633                 and m5[0] not in horscourse:
634             print('  W____M5', m5[0])
635             compare(m5[0])
636         # Lecteurs
637         if not nombre % d1[0] and d1[0] not in hautniveau \
638                 and d1[0] not in horscourse:
639             print('  W____D1', d1[0])
640             compare(d1[0])
641         if not nombre % d5[0] and d5[0] not in hautniveau \
642                 and d5[0] not in horscourse:
643             print('  W____D5', d5[0])
644             compare(d5[0])
645         # Lecteurs E: Entre début (P) et mi course (M)
646         if not nombre % ed1[0] and ed1[0] not in hautniveau \
647                 and ed1[0] not in horscourse:
648             print('  W____ED1', ed1[0])
649             compare(ed1[0])
650         if not nombre % ed5[0] and ed5[0] not in hautniveau \
651                 and ed5[0] not in horscourse:
652             print('  W____ED5', ed5[0])
653             compare(ed5[0])
654         if not nombre % em1[0] and em1[0] not in hautniveau \
```

```
            and em1[0] not in horscourse:
        print('  W____EM1', em1[0])
        compare(em1[0])
    if not nombre % em5[0] and em5[0] not in hautniveau \
            and em5[0] not in horscourse:
        print('  W____EM5', em5[0])
        compare(em5[0])
    # Lecteurs O: Entre carre (Q) et mi course (M)
    if not nombre % od1[0] and od1[0] not in hautniveau \
            and od1[0] not in horscourse:
        print('  W____OD1', od1[0])
        compare(od1[0])
    if not nombre % od5[0] and od5[0] not in hautniveau \
            and od5[0] not in horscourse:
        print('  W____OD5', od5[0])
        compare(od5[0])
    if not nombre % om1[0] and om1[0] not in hautniveau \
            and om1[0] not in horscourse:
        print('  W____OM1', om1[0])
        compare(om1[0])
    if not nombre % om5[0] and om5[0] not in hautniveau \
            and om5[0] not in horscourse:
        print('  W____OM5', om5[0])
        compare(om5[0])

    p1[0] += 6
    p5[0] += 6
    q1[0] -= 6
    q5[0] -= 6
    m1[0] += 6
    m5[0] += 6
    d1[0] -= 6
    d5[0] -= 6
    # 3ème niveau de profondeur
    ed1[0] -= 6
    ed5[0] -= 6
    em1[0] += 6
    em5[0] += 6
    od1[0] -= 6
    od5[0] -= 6
    om1[0] += 6
    om5[0] += 6

if nombre != 0:
    """Désigne la communauté:
    Les nombres premiers associés :hautniveau:
    Produisent la communauté des multiples communs"""

    hi = [u for u in horscourse if u <= carre]
    hi = len(hi)
    hautniveau.sort()
    """print('car %s car6 %s bor %s \n D1 %s OM1 %s Q1=%s'
          % (carre, cartyp6, borne, d1[0], om1[0], q1[0]))"""
    print('Premiers (', hi, ')ex\n', hautniveau, carre)
    for i in horscourse[:hi]:
        if breakComm == 0:
            iii = -1
            while horscourse[iii] >= carre:
                iii -= 1
                pass
            else:
```

```
716                     ha = horscourse[iii]
717                 hu = horscourse[0]
718                 print("Duos (inf)*(sup)|(sup)*(inf) \n  {} * {} "
719                       "Types {}&{}".format(hu, nombre // hu, hu % 6,
(nombre // hu) % 6))
720                 print(
721                     '  {} * {} Types {}&{}'.format(
722                         ha, nombre // ha, ha % 6, (nombre // ha) % 6))
723                 break
724             else:
725                 print(
726                     '{} * {} Types {}&{}'.format(
727                         i, nombre // i, i % 6, (nombre // i) % 6))
728     wifi()
729
730 #
731
```