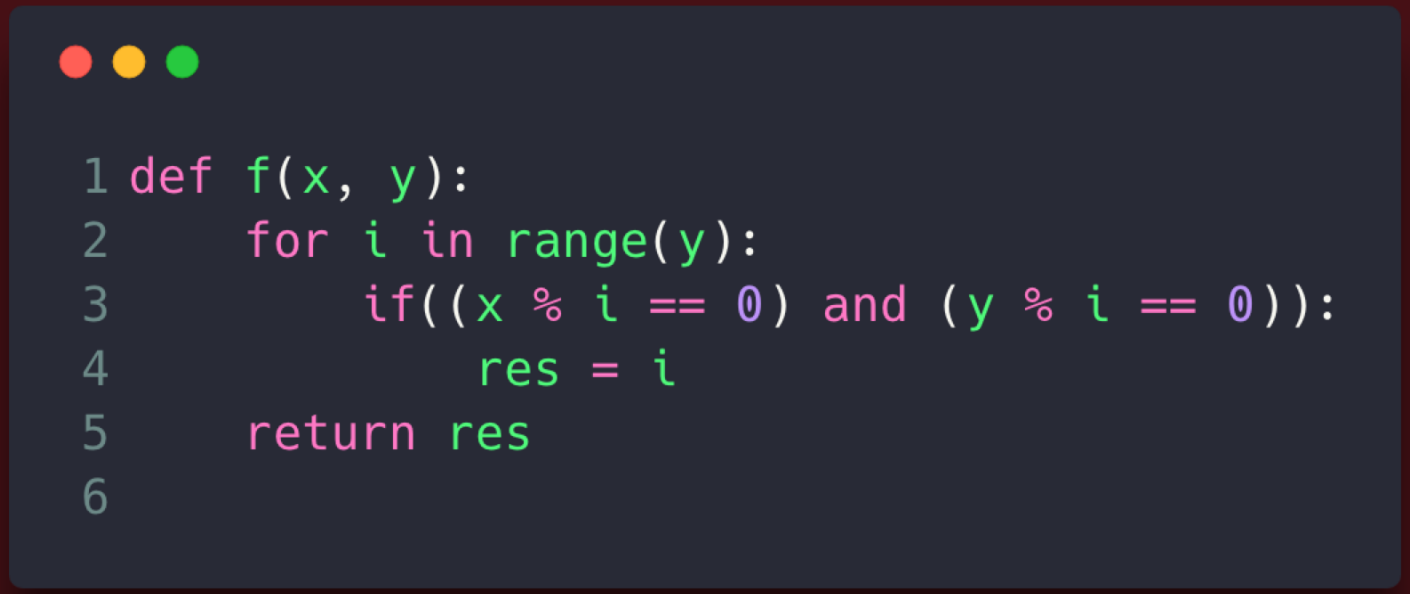
«Le vrai codeur ne teste pas.

À la rigueur, ce sont les utilisateurs qui testent.

Ou les autres, ceux qui doutent.

Mais pas lui.»

@RomainRouvoy


def f(x, y):
for i in range(y):
if((x % i == 0) and (y % i == 0)):
res = i
return res

Tester

C’est accepter

Bugs, bogues, crashs, erreurs...

```
1 → python -i JEIA1.py
2 >>> f(1,2)
3 Traceback (most recent call last):
4   File "<stdin>", line 1, in <module>
5   File "JEIA1.py", line 22, in f
6     if((x % i == 0) and (y % i == 0)):
7 ZeroDivisionError: integer division or modulo by zero
8 >>>
9
```

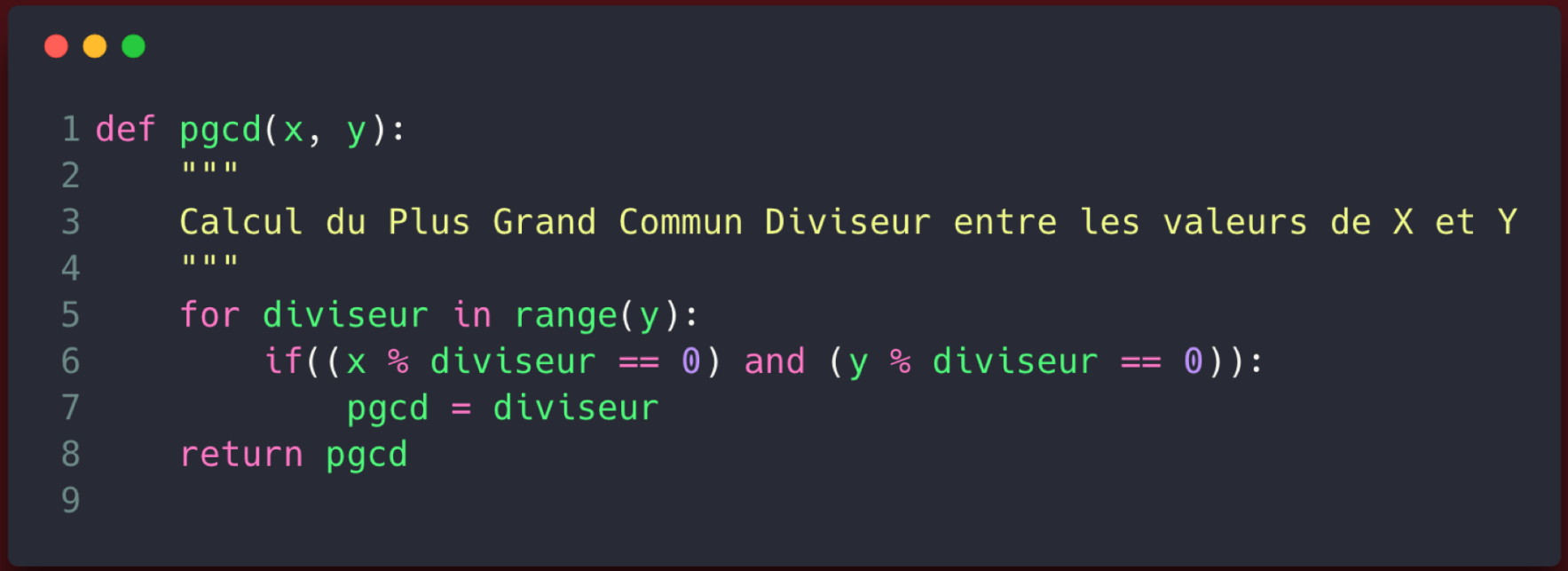
Bugs, bogues, crashs, erreurs...

```
1 def f(x, y):
2     for i in range(y):
3         if((x % i == 0) and (y % i == 0)):
4             res = i
5     return res
6
```

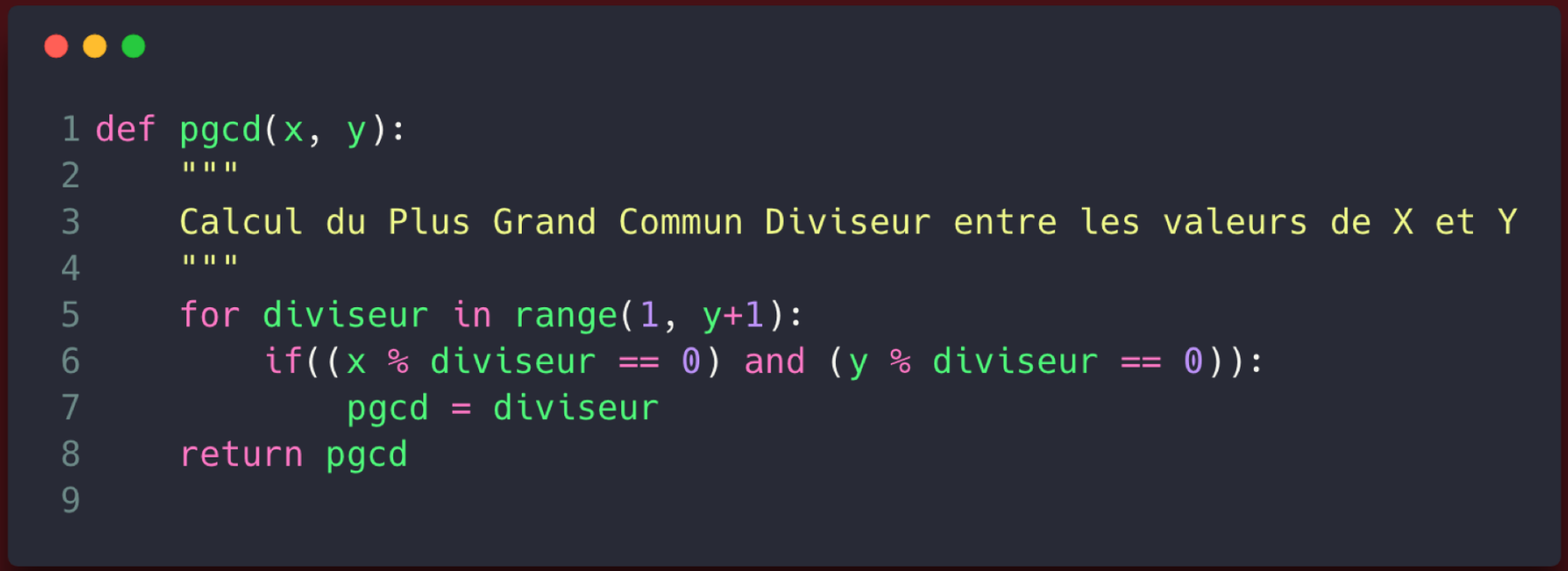
Tester

C'est documenter

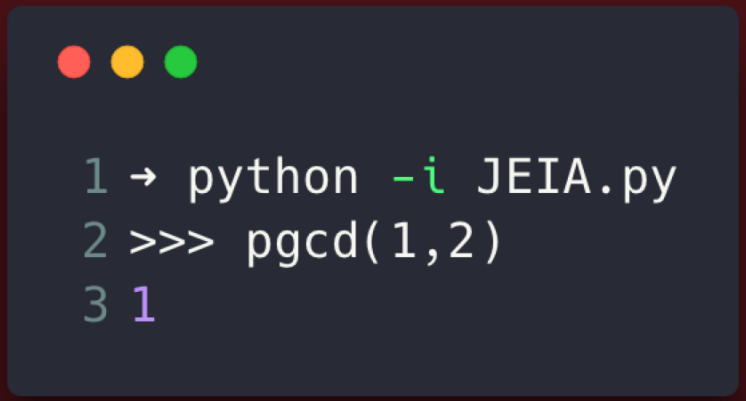
```python
def f(x, y):
    for i in range(y):
        if((x % i == 0) and (y % i == 0)):
            res = i
    return res

```



```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    """
    for diviseur in range(y):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd

```



```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    """
    for diviseur in range(1, y+1):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd

```


1 ➜ python -i JEIA.py
2 >>> pgcd(1,2)
3 1

Tester

C’est spécifier

Tests logiciels

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    """
    for diviseur in range(1, y+1):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd

```

Tests logiciels : DocTest

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    >>> pgcd(14,21)
    7
    """
    for diviseur in range(1, y+1):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd
```

Tests logiciels : DocTest

```
→ python -m doctest -v JEIA.py
Trying:
    pgcd(14,21)
Expecting:
    7
ok
1 items had no tests:
    JEIA
1 items passed all tests:
   1 tests in JEIA.pgcd
1 tests in 2 items.
1 passed and 0 failed.
Test passed.
```

Tests logiciels : DocTest

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    >>> pgcd(14,21)
    7
    >>> pgcd(21,14)
    7
    >>> pgcd(-14,21)
    7
    >>> pgcd(14,-21)
    7
    """
    for diviseur in range(1, y+1):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd
```

Tests logiciels : DocTest

```
→ python -m doctest JEIA.py
**********************************************************************
File "JEIA.py", line 44, in JEIA.pgcd
Failed example:
    pgcd(14,-21)
Exception raised:
    Traceback (most recent call last):
      File "/System/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/doctest.py", line
 1315, in __run
        compileflags, 1) in test.globs
      File "<doctest JEIA.pgcd[3]>", line 1, in <module>
        pgcd(14,-21)
      File "JEIA.py", line 50, in pgcd
        return pgcd
    UnboundLocalError: local variable 'pgcd' referenced before assignment
**********************************************************************
1 items had failures:
   1 of   4 in JEIA.pgcd
***Test Failed*** 1 failures.
```

Tests logiciels : DocTest

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    >>> pgcd(14,21)
    7
    >>> pgcd(21,14)
    7
    >>> pgcd(-14,21)
    7
    >>> pgcd(14,-21)
    7
    """
    for diviseur in range(1, abs(y)+1):
        if((x % diviseur == 0) and (y % diviseur == 0)):
            pgcd = diviseur
    return pgcd
```

Tests logiciels : Do

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Divise
    >>> pgcd(14,21)
    7
    >>> pgcd(21,14)
    7
    >>> pgcd(-14,21)
    7
    >>> pgcd(14,-21)
    7
    """
    for diviseur in range(1, abs(y)+1)
        if((x % diviseur == 0) and (y
            pgcd = diviseur
    return pgcd
```

```
→ python -m doctest JEIA.py -v
Trying:
    pgcd(14,21)
Expecting:
    7
ok
Trying:
    pgcd(21,14)
Expecting:
    7
ok
Trying:
    pgcd(-14,21)
Expecting:
    7
ok
Trying:
    pgcd(14,-21)
Expecting:
    7
ok
1 items had no tests:
    JEIA
1 items passed all tests:
   4 tests in JEIA.pgcd
4 tests in 2 items.
4 passed and 0 failed.
Test passed.
```

Tester

C’est construire

TDD: *Test-Driven Development*

```python
def ppcm(x, y):
    """
    Calcul du Plus Petit Commun Multiple entre X et Y
    >>> ppcm(14,21)
    42
    """
    pass
```

TDD: *Test-Driven Development*

```
➜ python -m doctest JEIA.py
**********************************************************************
File "JEIA.py", line 55, in JEIA.ppcm
Failed example:
    ppcm(14,21)
Expected:
    42
Got nothing
**********************************************************************
1 items had failures:
   1 of   1 in JEIA.ppcm
***Test Failed*** 1 failures.
```

TDD: *Test-Driven D*

```python
def ppcm(x, y):
    """
    Calcul du Plus Petit Comm
    >>> ppcm(14,21)
    42
    """
    return 42
```

```
→  python python -m doctest JEIA.py -v
Trying:
    pgcd(14,21)
Expecting:
    7
ok
Trying:
    pgcd(21,14)
Expecting:
    7
ok
Trying:
    pgcd(-14,21)
Expecting:
    7
ok
Trying:
    pgcd(14,-21)
Expecting:
    7
ok
Trying:
    ppcm(14,21)
Expecting:
    42
ok
1 items had no tests:
    JEIA
2 items passed all tests:
   4 tests in JEIA.pgcd
   1 tests in JEIA.ppcm
5 tests in 3 items.
5 passed and 0 failed.
Test passed.
```

TDD: *Test-Driven Development*

```python
def ppcm(x, y):
    """
    Calcul du Plus Petit Commun Multiple entre X et Y
    >>> ppcm(14,21)
    42
    >>> ppcm(21,-14)
    -42
    """
    return 42
```

TDD: *Test-Driven Development*

```
➜  python python -m doctest JEIA.py
**********************************************************************
File "JEIA.py", line 81, in JEIA.ppcm
Failed example:
    ppcm(21,-14)
Expected:
    -42
Got:
    42
**********************************************************************
1 items had failures:
   1 of   2 in JEIA.ppcm
***Test Failed*** 1 failures.
```

TDD: *Test-Driven Development*

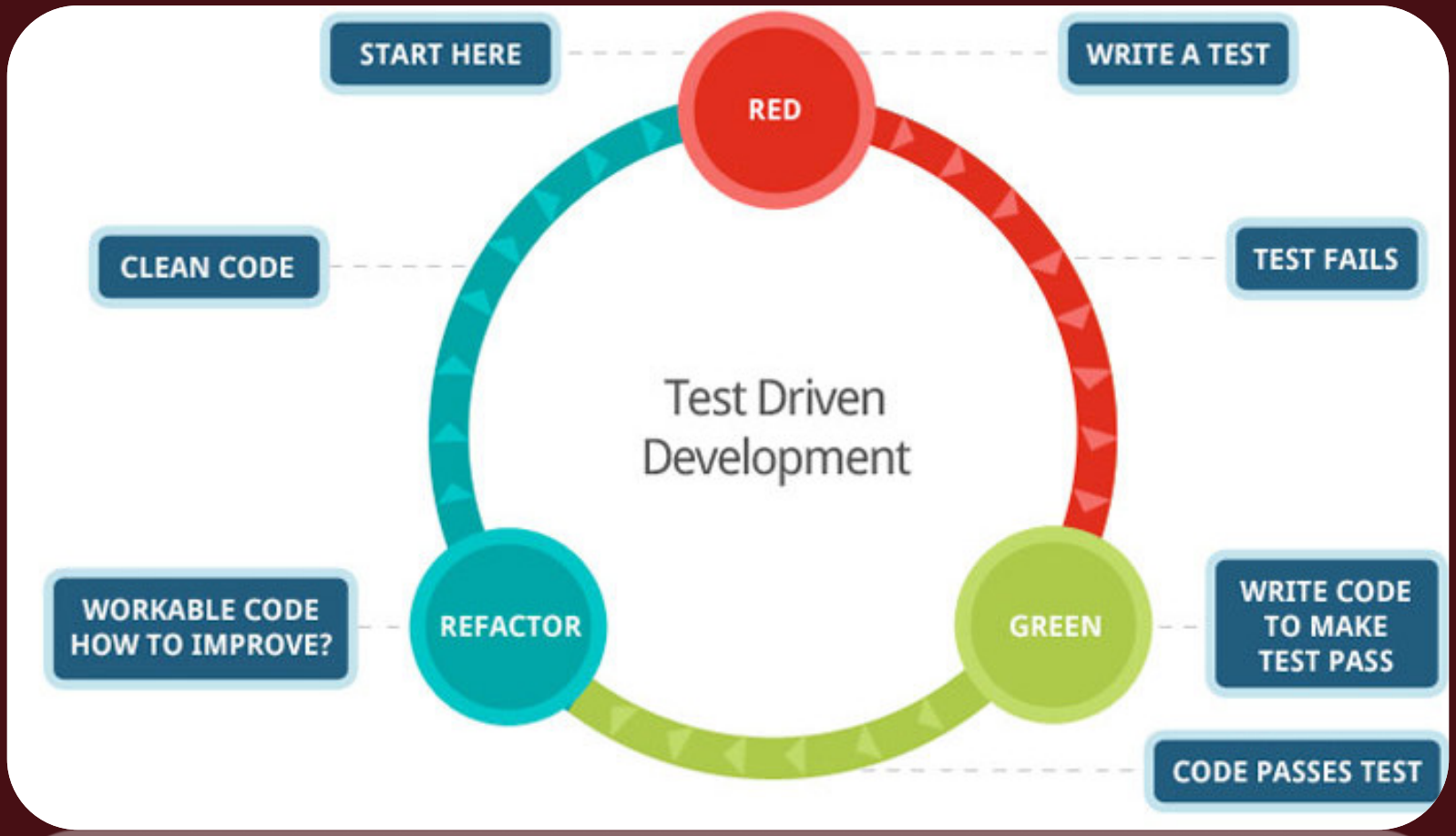
```python
def ppcm(x, y):
    """
    Calcul du Plus Petit Commun Multiple entre X et Y
    >>> ppcm(14,21)
    42
    """
    return x*y // pgcd(x,y)
```

TDD: *Test-Driven D*

```python
def ppcm(x, y):
    """
    Calcul du Plus Petit Comm
    >>> ppcm(14,21)
    42
    """
    return x*y // pgcd(x,y)
```

```
→  python python -m doctest JEIA.py -v
Trying:
    pgcd(14,21)
Expecting:
    7
ok
Trying:
    pgcd(21,14)
Expecting:
    7
ok
Trying:
    pgcd(-14,21)
Expecting:
    7
ok
Trying:
    pgcd(14,-21)
Expecting:
    7
ok
Trying:
    ppcm(14,21)
Expecting:
    42
ok
1 items had no tests:
    JEIA
2 items passed all tests:
   4 tests in JEIA.pgcd
   1 tests in JEIA.ppcm
5 tests in 3 items.
5 passed and 0 failed.
Test passed.
```

TDD: *Test-Driven Development*



TESTER

C’est consolider

Clean Code

*«You know you are working on clean code when each routine you read turns out to be **pretty much what you expected**.»*

*«You can call it beautiful code when the code also makes it look like the **language was made for the problem**»*

Ward Cunningham

Clean Code

- Utiliser des noms pertinents et prononciables
- Utiliser des noms «*cherchables*»
- Utiliser des variables explicatives
- Éviter les associations mentales
- Éviter plus de 2 paramètres par fonction
- Toute fonction doit faire une seule chose
- Toute function doit dire ce qu’elle fait
- Éviter les effets de bord
- Éviter les conditions négatives
- Retirer tout code mort
- …

```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Diviseur entre les valeurs de X et Y
    >>> pgcd(14,21)
    7
    >>> pgcd(21,14)
    7
    >>> pgcd(-14,21)
    7
    >>> pgcd(14,-21)
    7
    """
    for diviseur in les_diviseurs(x,y):
        if(est_diviseur_commun(diviseur,x,y)):
            pgcd = diviseur
    return pgcd

def les_diviseurs(x, y):
    minimum = min(abs(x),abs(y))
    return range(1, minimum+1)

def est_diviseur_commun(diviseur,x,y):
    return (x % diviseur == 0) and (y % diviseur == 0)
```

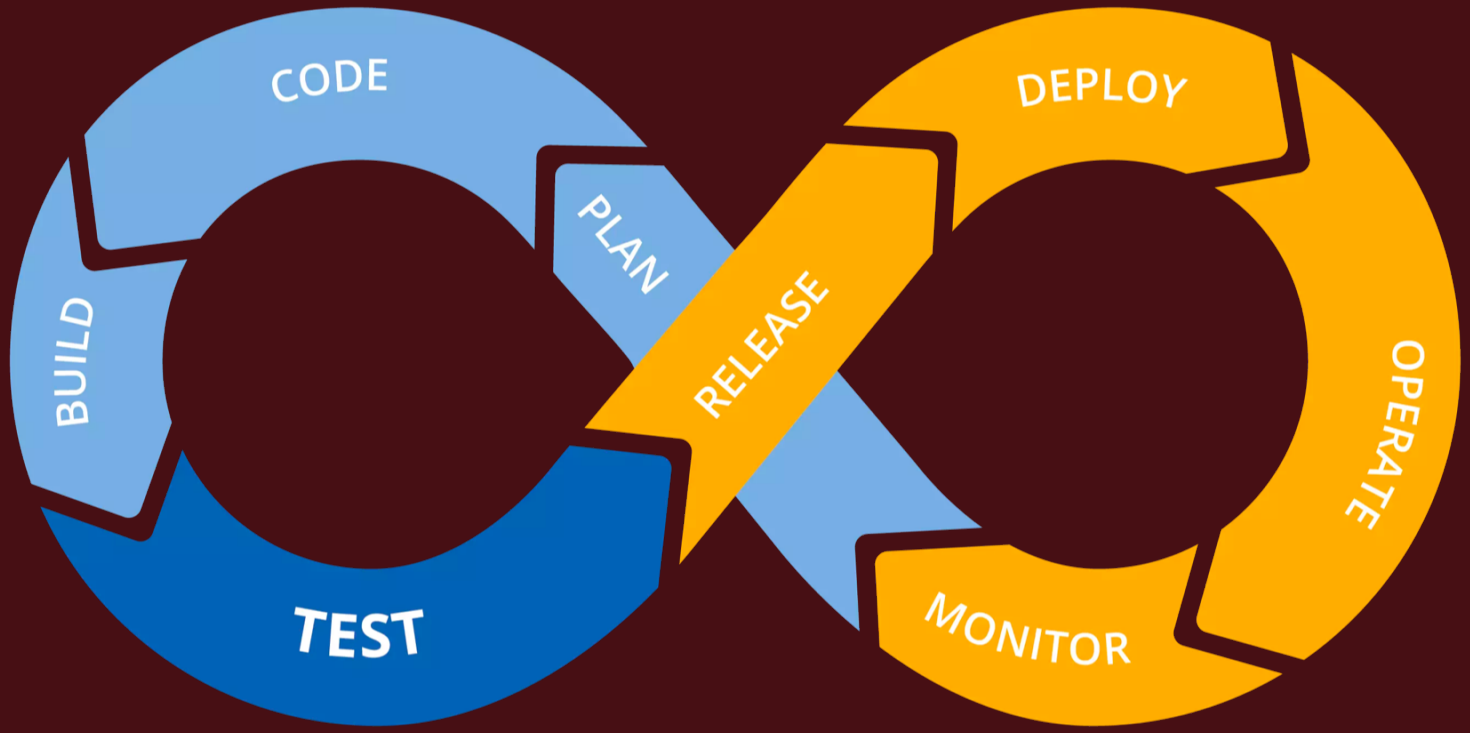
```python
def pgcd(x, y):
    """
    Calcul du Plus Grand Commun Divise
    >>> pgcd(14,21)
    7
    >>> pgcd(21,14)
    7
    >>> pgcd(-14,21)
    7
    >>> pgcd(14,-21)
    7
    """
    for diviseur in les_diviseurs(x,y)
        if(est_diviseur_commun(diviseu
            pgcd = diviseur
    return pgcd

def les_diviseurs(x, y):
    minimum = min(abs(x),abs(y))
    return range(1, minimum+1)

def est_diviseur_commun(diviseur,x,y):
    return (x % diviseur == 0) and (y
```

```
➜  python python -m doctest JEIA.py -v
Trying:
    pgcd(14,21)
Expecting:
    7
ok
Trying:
    pgcd(21,14)
Expecting:
    7
ok
Trying:
    pgcd(-14,21)
Expecting:
    7
ok
Trying:
    pgcd(14,-21)
Expecting:
    7
ok
Trying:
    ppcm(14,21)
Expecting:
    42
ok
1 items had no tests:
    JEIA
2 items passed all tests:
   4 tests in JEIA.pgcd
   1 tests in JEIA.ppcm
5 tests in 3 items.
5 passed and 0 failed.
Test passed.
```

TESTER

C’est avancer

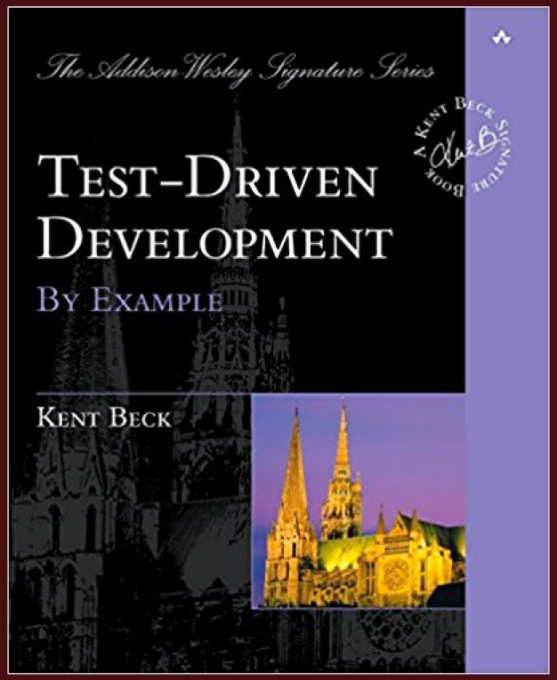
Bouchons de tests
Tests d’integration
Tests d’acceptance
Tests de performance
...
Gestion de versions
Intégration continue
Livraison continue
...


CODE
PLAN
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR

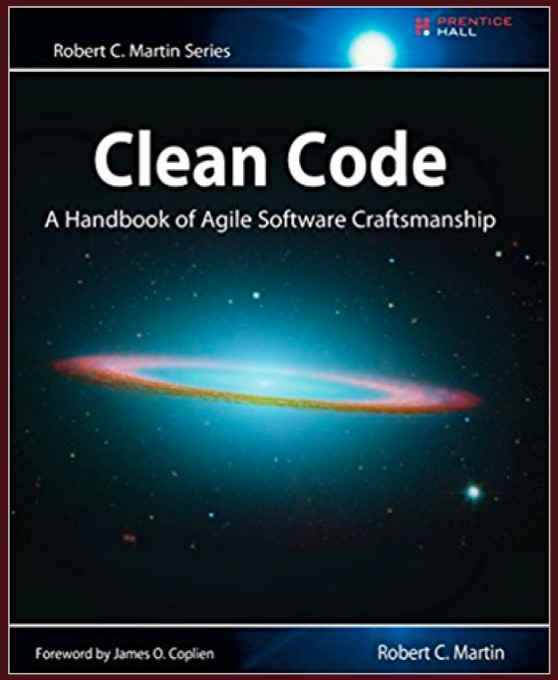
Tester

C’est coder

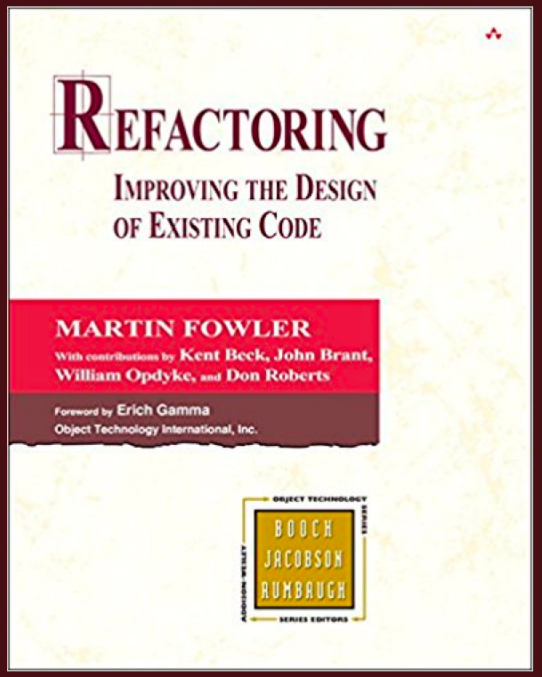
Pour aller plus loin...



Kent Beck



Robert C. Martin



Martin Fowler