# Euclidean algorithm

## Preparation

Watch the demonstration video on test-driven development to learn how to work with doctests. This will make you understand the last two statements of the source code skeleton provided with this exercise.

## Assignment

We provide you with a skeleton for the solution of this exercise. Your task is to implement the functions `gcd` and `lcm`, such that they respectively print the greatest common divisor and the least common multiple of two given integers $a$ and $b$. The greatest common divisor is the largest integer that is both a divisor of $a$ and $b$. The least common multiple is the smallest integer that is both a multiple of $a$ and $b$.

```
def gcd(a, b):
    """
    Computes the greatest common divisor of the two
    integers a and b.

    >>> gcd(252, 105)
    21
    >>> gcd(48, 18)
    6
    """

    # compute greatest common divisor based on the
    # Euclidean algorithm

def lcm(a, b):
    """
    Computes the least common multiple of the two
    integers a and b.

    >>> lcm(252, 105)
    1260
    >>> lcm(48, 18)
    144
    """

    # compute least common multiple

if __name__ == '__main__':
    import doctest
    doctest.testmod()
```

The `fractions` module from the Python Standard Library already contains a function `gcd` that prints the greatest common divisor of two integers. However, for this exercise you cannot use the `fractions` module, as you come up with your own implementation of computing the greatest common divisor by yourself. The implementation of the `gcd` function should be based on the **Euclidean algorithm**. The algorithm starts with a pair of integers and computes a new pair that

contains the smallest of the two integers and the difference of the largest and the smallest integer. This procedure is repeated until both integers have the same value. This value is the greatest common divisor of the original pair of integers.

The basic principle that is applied in the Euclidean algorithm, is that the largest common divisor does not change when the smallest integer is subtracted from the largest integer. As such, the greatest common divisor of 252 and 105 is also the greatest common divisor of 147 (= 252 - 105) and 105. As the largest integer is always lowered during each step of, the algorithm will ultimately stop when both integers are the same (if the procedure would be repeated yet another time, one of the two integers would become equal to zero).

The least common multiple of two integers $a$ and $b$ can be determined as the product of both integers, divided by their greatest common divisor.

## Example

```
>>> gcd(252, 105)
21
>>> gcd(48, 18)
6

>>> lcm(252, 105)
1260
>>> lcm(48, 18)
144
```

## Voorbereiding

Bekijk de video over test-driven development om te leren werken met doctests. Hierdoor zal je de laatste twee statements begrijpen in het skelet van de broncode dat wordt meegegeven met deze opgave.

## Opgave

We hebben hieronder reeds het skelet opgemaakt voor de oplossing van deze opgave. Je taak bestaat erin om de functies `ggd` en `kgv` te implementeren, zodat ze respectievelijk de grootste gemene deler en het kleinste gemene veelvoud van de natuurlijke getallen $a$ en $b$ teruggeven. De grootste gemene deler is het grootste natuurlijke getal dat zowel een deler is van $a$ als van $b$. Het kleinste gemene veelvoud is het kleinste natuurlijke getal dat zowel een veelvoud is van $a$ als van $b$.

```
def ggd(a, b):

    """
    Berekent de grootste gemene deler van de twee
    natuurlijke getallen a en b.

    >>> ggd(252, 105)
    21
    >>> ggd(48, 18)
    6
    """
```

```
    # berekening grootste gemene deler op basis van
    # het algoritme van Euclides

def kgv(a, b):

    """
    Berekent het kleinste gemene veelvoud van de
    twee natuurlijke getallen a en b.

    >>> kgv(252, 105)
    1260
    >>> kgv(48, 18)
    144
    """

    # berekening kleinste gemene veelvoud

if __name__ == '__main__':
    import doctest
    doctest.testmod()
```

De fractions module uit de Python Standard Library bevat een functie gcd die de grootste gemene deler van twee natuurlijke getallen teruggeeft. Voor deze opgave mag je de fractions module echter niet gebruiken, en is het de bedoeling dat je grootste gemene deler zelf berekent. Baseer je implementatie van de functie ggd op het **algoritme van Euclides**. Het algoritme start met een paar natuurlijke getallen, en vormt daarmee een nieuw paar dat bestaat uit het kleinste van de twee getallen, en het verschil tussen het grootste en het kleinste getal. Deze procedure herhaalt zich totdat het paar bestaat uit twee gelijke getallen. Dat getal is de grootste gemene deler van het oorspronkelijke paar natuurlijke getallen.

Het basisprincipe dat hierbij gehanteerd wordt, is dat de grootste gemene deler niet verandert als het grootste getal van het kleinste afgetrokken wordt. Zo is de grootste gemene deler van 252 en 105 ook de grootste gemene deler van 147 (= 252 - 105) en 105. Aangezien grootste getal verlaagd wordt, resulteert het herhaald uitvoeren van deze procedure in steeds kleinere getallen, zodat de herhaling vroeger of later zal stoppen wanneer de twee getallen gelijk zijn (als de procedure daarna nog een keer zou herhaald worden, dan zou één van beide getallen nul worden).

Het kleinste gemene veelvoud van twee natuurlijke getallen $a$ en $b$ kan bepaald worden als het product van beide getallen, gedeeld door hun grootste gemene deler.

## Voorbeeld

```
>>> ggd(252, 105)
21
>>> ggd(48, 18)
6

>>> kgv(252, 105)
1260
>>> kgv(48, 18)
144
```