```
#Abrudan Rebeca Rafaela 931
import random
import time


def gcd_euclidean(x, y):
    """
    check if y divides by x, if yes => x=gcd,
    otherwise x=the rest of the division and new y=old x
    it divides the smaller number, the algorithm stops when it finds the remainder 0.
    """
    if x == 0:
        return y
    else:
        return gcd_euclidean(y % x, x)


def gcd_brute_force(x, y):
    """
    start with the smaller of the two numbers, and continue down to 1,
    check for gcd, one by one. i=gcd
    """
    if x < y:
        i = x
    else:
        i = y

    if x == 0 or x == y:
        return y
```

```
    elif y == 0:
        return x

    while x % i != 0 or y % i != 0:
        i -= 1

    return i



def gcd_subtraction(x, y):
    """
    subtracting numbers from each other until they are equal
    """
    if x == 0 or x == y:
        return y
    elif y == 0:
        return x
    while x != y:
        if x > y:
            x -= y
        else:
            y -= x
    return x



# run-time analysis


# numbersX = []
# numbersY = []
```

```
#
# for _ in range(1000):
#     random_number = random.randint(1, 1000000)
#     numbersX.append(random_number)
#     random_number = random.randint(1, 1000000)
#     numbersY.append(random_number)


#
# start_time = time.time_ns()
# for i in range(0, 900):
#     gcd_euclidean(numbersX[i], numbersY[i])
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
# print("Elapsed time in nanoseconds (euclidean): ", elapsed_time)
#
#
# start_time = time.time_ns()
# for i in range(0, 900):
#     gcd_subtraction(numbersX[i], numbersY[i])
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
# print("Elapsed time in nanoseconds (subtraction): ", elapsed_time)
#
#
# start_time = time.time_ns()
# for i in range(0, 900):
#     gcd_brute_force(numbersX[i], numbersY[i])
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
```

```
# print("Elapsed time in nanoseconds (brute force): ", elapsed_time)


#
# x = 3
# y = 4
# print(gcd_subtraction(x,y))
#
#
#
#
# start_time = time.time_ns()
# for i in range(0, 9):
#     gcd_euclidean(numbersX[i], numbersY[i])
#     print("x:", numbersX[i], "y:", numbersY[i])
#     print("gcd:", gcd_euclidean(numbersX[i], numbersY[i]))
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
# print("Elapsed time in nanoseconds (euclidean): ", elapsed_time)
#
#
# start_time = time.time_ns()
# for i in range(0, 9):
#     gcd_subtraction(numbersX[i], numbersY[i])
#
#     print("x:", numbersX[i], "y:", numbersY[i])
#     print("gcd: ", gcd_subtraction(numbersX[i], numbersY[i]))
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
# print("Elapsed time in nanoseconds (subtraction): ", elapsed_time)
```

```
#
#
# start_time = time.time_ns()
# for i in range(0, 9):
#     gcd_brute_force(numbersX[i], numbersY[i])
#
#     print("x:", numbersX[i], "y:", numbersY[i])
#     print("gcd ", gcd_brute_force(numbersX[i], numbersY[i]))
# end_time = time.time_ns()
# elapsed_time = end_time-start_time
# print("Elapsed time in nanoseconds (brute force): ", elapsed_time)
#

x = 3
y = 0
print("subtraction: ", gcd_subtraction(x, y))
print("euclidean: ", gcd_euclidean(x, y))
print("brute-force: ", gcd_brute_force(x, y))
```