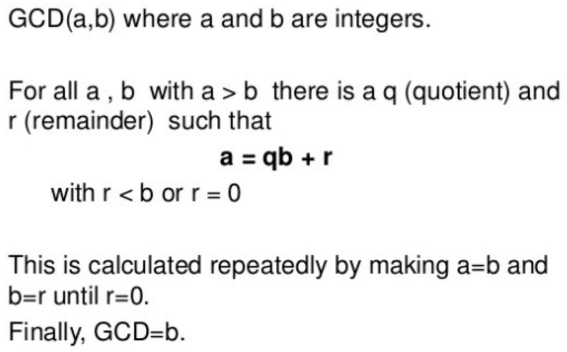
Lab 5: Recursion

Exercise 1 : Greatest Common Divisor using Loop

In this exercise we will implement the Euclidean algorithm for calculating GCD is given below



GCD(a,b) where a and b are integers.

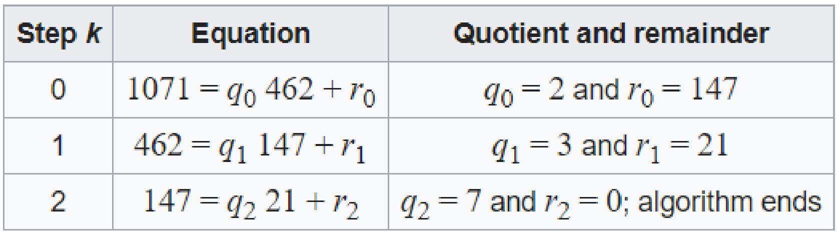
For all a , b with a > b there is a q (quotient) and
r (remainder) such that

a = qb + r

with r < b or r = 0

This is calculated repeatedly by making a=b and
b=r until r=0.
Finally, GCD=b.

Example: Let a = 1071 and b = 462 that is we are searching for GCD(1071, 462) then we will have following steps. Not that b of the current step is used as a in the next step and r of the current step is used b as the next step.



Step *k*	Equation	Quotient and remainder
0	$1071 = q_0\ 462 + r_0$	$q_0 = 2$ and $r_0 = 147$
1	$462 = q_1\ 147 + r_1$	$q_1 = 3$ and $r_1 = 21$
2	$147 = q_2\ 21 + r_2$	$q_2 = 7$ and $r_2 = 0$; algorithm ends

When r becomes 0 then the b gives you the GCD, which is 21 in this example. GCD(1071, 462) = 21

1. Create a class named GCDLoop which has a main method.
2. The class will accept two numbers and return the GCD as shown below

```
> java GCDLoop 90 50
```

> 10

Exercise 2 : Greatest Common Divisor using Recursion

1. Define the recursive GCD function and the base case for the Euclidean algorithm.
2. Implement the function in a class named GCDRec
3. The class will accept two numbers and return the GCD as shown below

```
> java GCDRec 90 50
```

NOTE: Your lab will **not be graded** if

- Your account name does not have the format described in lab1.pdf
- Your repository name is not lab
- Your files have compilation errors
- You haven't complete the steps described in exercises
- Your added/modified files are not submitted to Bitbucket.
- You have to add commit and push files as described in lab1.pdf