## Euclidean Algorithm by Subtraction

In the original Euclide's algorithm the GCD is found by repeated subtraction. For instance, find the GCD of 42 and 30; usually written in short notation GCD(A, B)=GCD(42,30). We begin with the original numbers, say 42 and 30, and subtract the smaller from the larger:

Choose the two smallest numbers in the set (42,30,12), these are 30 and 12. Again, we subtract the smallest from the largest:

30-12=18

And again

18-12=6

And again

12-6=6

And again, until we get 0 as an answer

6-6=0

Once we got to zero (0), the number we subtracted from itself (i.e., 6) is our greatest common factor. Therefore, if 6 is the GCD, then two original numbers can be obtained by  $42 = 7 \times 6$  and  $30 = 5 \times 6$ .

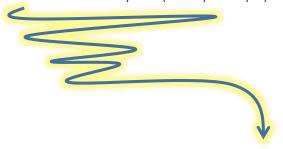
Let's do it again for a harder problem. Find the GCD of 351 and 221. We do exactly the same thing:

26 - 13 = 13

13 - 13 = 0

Our greatest common factor is 13, then  $221 = 13 \times 17$ ,  $351 = 13 \times 27$ .

Now, call the first number A and the second number B, and follow the mechanism above. For the implementation below, it is assumed that initially A>B (to keep it simple).



Version-1	Version-2
% Euclidean (GCD) Algorithm	% Euclidean (GCD) Algorithm
% Computes GCD by repeated subtraction	% Computes GCD by repeated subtraction
% A [integer], B [integer]	% A [integer], B [integer]
% File: EucledeanSubtraction.m	% File: EucledeanSubtraction.m
clc, clear	clc, clear
A=input('Enter an integer number (e.g., 351)	A=input('Enter an integer number (e.g., 351)
\n');	\n');
B=input('Enter another integer number	B=input('Enter another integer number
(e.g.,221 )\n');	(e.g.,221 )\n');
while B~=0	while B~=0
if A>B	if A>B
C=A-B;	A=A-B;
A_D.	else
A=B;	B=B-A;
B=C;	end
else	end
C=B-A;	fprintf('\n The GCD is %d ', A);
B=C;	
,	
end	
end	
fprintf('\n The GCD is %d ', A);	

Version 2 make fewer steps, hence more efficient (but harder to understand, do you?)