Example of using the ExtendedEuclid algorithm

Goal: I want to find the multiplicative inverse of 17, modulo 31, if it exists.

So I use these two numbers as the arguments to the ExtendedEuclid() function, as follows:

Call ExtendedEuclid(31, 17)
\[ a = 31, \ b = 17 \]
Call ExtendedEuclid(17, 31 mod 17 = 14)
\[ a = 17, \ b = 14 \]
Call ExtendedEuclid(14, 17 mod 14 = 3)
\[ a = 14, \ b = 3 \]
Call ExtendedEuclid(3, 14 mod 3 = 2)
\[ a = 3, \ b = 2 \]
Call ExtendedEuclid(2, 3 mod 2 = 1)
\[ a = 2, \ b = 1 \]
Call ExtendedEuclid(1, 2 mod 1 = 0)
\[ a = 1, \ b = 0; \text{ since } b == 0, \text{ we have reached the base case} \]
\[ \text{return } (1, 0, 1) \]
\[ (x' = 1, \ y' = 0, \ d = 1) \text{ is the return value from the sub-call} \]
\[ \text{return } (0, 1-2(0)=1, 1) \]
\[ (x' = 0, \ y' = 1, \ d = 1) \text{ is the return value from the sub-call} \]
\[ \text{return } (1, 0-1(1)=-1, 1) \]
\[ (x' = 1, \ y' = -1, \ d = 1) \text{ is the return value from the sub-call} \]
\[ \text{return } (-1, 1-4(-1)=5, 1) \]
\[ (x' = -1, \ y' = 5, \ d = 1) \text{ is the return value from the sub-call} \]
\[ \text{return } (5, -1-1(5)=-6, 1) \]
\[ (x' = 5, \ y' = -6, \ d = 1) \text{ is the return value from the sub-call} \]
\[ \text{return } (-6, 5-1(-6)=11, 1) \]
\[ (x = -6, y = 11, d = 1) \text{ is the return value from the top level function call.} \]

Relative Primality:
- The second value, \( y = 11 \), corresponds to the second argument of the call \( b = 17 \).
- Since \( a \times x + b \times y = d \), \( 31 \times (-6) + 17 \times 11 \) should equal 1, and it does:
  \[ 31(-6) + 17(11) = -186 + 187 = 1 \]
- Furthermore, the GCD \( d = 1 \).
- Consequently, 17 and 31 are indeed relatively prime.

Multiplicative Inverse:
- The multiplicative inverse of 17 \( (b) \), modulo 31 \( (a) \), is 11 \( (y) \).
- In other words, \( 17 \cdot 11 \equiv 1 \mod 31 \).
Notes:

Order of arguments: If ExtendedEuclid(a,b) returns (x,y,d), then ExtendedEuclid(b,a) returns (y,x,d), so order of the arguments is not an issue, as long as you interpret the results in the right order.

RSA: If you are using the ExtendedEuclid( ) algorithm for RSA, then you will call ExtendedEuclid(\(\phi(N), e\)), and you will get back (x, y, d). The last element of that result triple should be 1, indicating that \(\phi(N)\) and \(e\) are relatively prime, as needed. The value you are really after – the multiplicative inverse of \(e\) modulo \(\phi(N)\) – is \(y\). Call that \(d\) now. Unfortunately, when talking about ExtendedEuclid(), we use the variable \(d\) as the GCD of the arguments, but when talking about RSA, the variable \(d\) is the name of the multiplicative inverse. Anyhow, take the result called \(y\) from ExtendedEuclid( ) and rename it as \(d\), and just run with it as the private key.