Assignment – 4 Date: October 07, 2016

Submission Deadline: October 21, 2016

All assignments will be marked out of 20. Assignments submitted after the deadline will be penalized with 5 marks for each week delay.

1. Implement the Euclidean Algorithm below, to find GCD of two numbers:

```
EUCLIDEAN ALGORITHM(a, b)

r0 ← a
r1 ← b
m ← 1
while r_m ≠ 0
         ⎧ q_m ← ⌊r_{m-1} / r_m⌋
    do   ⎨ r_{m+1} ← r_{m-1} − q_m r_m
         ⎩ m ← m + 1
m ← m − 1
return (q_1, ..., q_m; r_m)
comment: r_m = gcd(a, b)
```

2. Given two integers *a* and *b*, the following algorithm computes GCD (a,b) as well as b^{-1} mod a, when a and b are co-prime to each other. This is called the Extended Euclidean Algorithm.

```
EXTENDED EUCLIDEAN ALGORITHM(a, b)

a0 ← a
b0 ← b
t0 ← 0
t ← 1
s0 ← 1
s ← 0
q ← ⌊a0 / b0⌋
r ← a0 − q b0
while r > 0
         ⎧ temp ← t0 − q t
         ⎪ t0 ← t
         ⎪ t ← temp
         ⎪ temp ← s0 − q s
    do   ⎨ s0 ← s
         ⎪ s ← temp
         ⎪ a0 ← b0
         ⎪ b0 ← r
         ⎪ q ← ⌊a0 / b0⌋
         ⎩ r ← a0 − q b0
r ← b0
return (r, s, t)
comment: r = gcd(a, b) and sa + tb = r
```

This is how the algorithm works:
Given a and b it computes another two number s and t such that s × a + t × b = r = GCD (a,b).
Now, we aim to find b^{-1} mod a, which exists iff a and b are co-prime.
Since a and b are co-prime r = 1.
Therefore, s × a + t × b = 1.
Applying mod a to both sides, (s × a + t × b) mod a = 1 mod a.
Or, t × b mod a = 1 [Since s × a mod a = 0.]
Or, b^{-1} mod a = t. (Think why we also output s.)

Implement the Extended Euclidean Algorithm. Hence prove 28^{-1} mod 75 = 67.

3. Implement the CRT (Chinese Remainder Theorem). Hence solve for x from the following set of congruences:

$$x \equiv 2 \pmod{3}$$
$$x \equiv 3 \pmod{5}$$
$$x \equiv 2 \pmod{7}$$