Zijian Gao
26319560
2/29/2016
CS311 HW3

31.2-2)

a	b	$\lfloor a/b \rfloor$	d	x	y
899	493	1	29	-6	11
493	406	1	29	5	-6
406	87	4	29	-1	5
87	58	1	29	1	-1
58	29	2	29	0	1
29	0	NAN	29	1	0

Return value is (29,-6,11)

31.2-8)

```
lcm(A[$a_1$, $a_2$, …, $a_n$]):
    if (A.size == 1)
        return A[1]
    if (A.size == 0)
        throw error
    acc = A[1]
    for (i from 2 to n):
        acc = lcm_help(acc, A[i])
    return acc
lcm_help(a, b):
    return a x (b / gcd(a,b))
```

Proof of correctness:
lcm_help utilizes reduction to LCM using the GCD. This is correct because gcd(a,b) is the greatest divisor in a and b, and dividing b by the GCD of a,b yields a b' such that and b' are relatively prime. For two relatively prime numbers a and b, the LCM is trivially just a x b. In other words, if a and b have common factors, a x b yields a number with two "copies" of those common factors, and so will be a larger number than the desired LCM. Those factors are exactly GCD(a,b), and dividing one set of them out yields the least number that has a and b as common factors.

This lcm for multiple inputs uses associativity and commutativity of LCM. Since LCM is both associative and commutative, as a base case, it is true that LCM(a,b,c) = LCM(a,LCM(b,c)) = LCM(LCM(a,b),c). This is a recursive definition that is implemented as a simple loop which utilizes this fact to take the lcm of the first and second number, then take that value and iteratively take its lcm with successive values until the end is reached.

Runtime analysis:
For worst case analysis, let y be the number that lcm takes as a parameter.

lcm_help runs in in $\theta(\log(y))$ time due to dependence on the Euclidian GCD algorithm. The loop makes n-1 iterations for a total runtime of $\theta((n-1)\log y)$