Mark Eatough

CSIS 2430 9:00 Class

Programming Project 3

Modular Exponentiation Program

Assignment objective:

Implement Modular Exponentiation. You will need Algorithms 1 & 5 from Chapter 4. Your algorithm will need to solve problems 25-28 on page 255.

What Worked?:

Once I figured out what the algorithms were doing implementing them was pretty simple. I only needed about ten to twelve lines of code for each method.

What did not work?:

Python parses out strings reverse from what I was expecting. I had assumed that the first character in the string would be at position 0, but that was actually the last character in the string.

Comments:

For this assignment figuring out the algorithms took a lot more work than actually implementing them. This is probably more indicative of how programming will be in the real world as a lot of programming jobs involve code maintenance or developing small parts of code for a large program rather than developing an entire program by yourself.

```
'''
****************************************************************
 * Discrete Structures
 * Modular Exponentiation Program
 * Programmer: Mark Eatough
 * Course: CSIS 2430
 * Created September 15, 2013

 *This program finds the modulus of a number raised to a
 *large power given the number, the exponent, and the modulus
 *The algorithm 1 method converts the exponent to a binary number
 *and the algorithm 5 method calls the algorithm one method, and
 *then uses that to find the modulus of our very large number.
*****************************************************************
'''
#algorithm 1 out of book, used to find binary representation of my exponents
def algorithm1(n,b):
   q = n
   binary = ""
   while(q > 0):
      a = q%b
      q = q/b
      binary = str(a) + binary
   return binary

#algorithm 5 out of book, used to find modulus of very large numbers
def algorithm5(n, b, m):
   x = 1
   binary = algorithm1(b, 2)
   i = len(binary)-1
   while (i > 0):
      n*=n
      n = n % m
```

```
        if(binary[i-1] == '1'):
            x*=n
            x = x % m
        i-=1
    return x

#print out the exponent in binary code
print "Find the binary translation of my exponents using algorithm 1:\n"
print "644 in binary is:\t", algorithm1(644, 2)
print "644 in binary is:\t", algorithm1(644, 2)
print "2003 in binary is:\t", algorithm1(2003, 2)
print "1001 in binary is:\t", algorithm1(1001, 2)
#print out my modulus
print "\n\nModulus using algorithm 5:\n"
print "7^644 mod 645 = \t", algorithm5(7,644, 645)
print "11^644 mod 645 = \t", algorithm5(11,644, 645)
print "3^2003 mod 645 = \t", algorithm5(3,2003, 99)
print "123^1001 mod 645 = \t", algorithm5(123,1001, 101)
```