LAB 6

Modular Arithmetic

PROBLEM 6.1. What does each of the following functions do?

(a) `fun[x_]:=If[x==0, "It's zero", "It's not zero"]`

(b) `func[x_]:=x*func[x-1]`

(c) `funct[x_]:=If[x==0, 1, x*funct[x-1]]`

Problem 6.2. Write a recursive function `MyDiv[a_, b_]` that returns the pair $\{\mathrm{Quo}(a,b), \mathrm{Rem}(a,b)\}$. Watch out for weird cases. Then use `MyDiv` to make functions `MyQuo` and `MyRem`.

Problem 6.3. Write a `ModularAddition[a_, b_, m_]` function that computes the sum of a and b, and is only correct mod m. For example, the output of `ModularAddition[6, 7, 10]` should be `3`. You may use Mathematica's + operation and `MyRem`.

Problem 6.4. Look back at the multiplication worksheet where we figured out how to quickly multiply two numbers. Then use `ModularAddition` to write a recursive `ModularMultiply[a_, b_, m_]` function that computes the product of a and b, and is only correct mod m.

Problem 6.5. To make RSA work, we will also need exponents.

(a) Write a recursive function `QuickExponentiate[a_, b_]` that computes a^b (for $b > 0$). Hint: use the same trick as for multiplication.

(b) Write a recursive function `ModularExponentiate[a_, b_, m_]` that computes a^b (for $b > 0$), and is only correct mod m.