Computer Science 308-250B Homework #1
Due Monday January 27, 2003, 17:00

[25 %] Question 1. On the second class, we saw a number of multiplication algorithms including the Russian method known as “multiplication à la russe”. Find below a recursive definition of this algorithm (the comments should help you understand why it works):

Algorithm Mulàlarusse(a,b)
If (b=0)
   Then return 0 // a · b = a · 0 = 0
Else If (b is even)
   Then return Mulàlarusse( a+a, b/2 ) // a · b = (2 · a) · (b/2)
Else return a + Mulàlarusse( a+a, (b-1)/2 ) // a · b = a + (2 · a) · ((b − 1)/2)

The question is: Write an iterative definition of this algorithm (no recursion allowed).

[25 %] Question 2. Now notice that if we imbed (mod n) operators in this definition we end up with an algorithm computing (a · b) mod n instead:

Algorithm MulMOD(a,b,n)
If (b=0)
   Then return 0 // a · b = a · 0 = 0 mod n
Else If (b is even)
   Then return MulMOD( a+a, b/2,n ) // a · b mod n = (2 · a) · (b/2) mod n
Else return ( a + MulMOD( a+a, (b-1)/2,n ) ) mod n
   // a · b mod n = a + (2 · a) · ((b − 1)/2) mod n

The question is: Find the largest value of n of type long for which this method finds the correct answer for all a, b with 0 ≤ a, b ≤ n − 1 and compare it with the direct Java expression (a*b) % n. Write a Java program to find the largest correct values for both methods.

[25 %] Question 3. Now notice that if we replace “+” by “*” in the MulMOD algorithm and “0” by “1”, we end up defining an algorithm for a^b mod n instead of (a · b) mod n:

Algorithm ExpMOD(a,b,n)
If (b=0)
   Then return 1 // a^0 = 1 mod n
Else If (b is even)
   Then return ExpMOD( a*a mod n, b/2,n ) // a^b mod n = (a^2)^b/2 mod n
Else return ( a * ExpMOD( a*a mod n, (b-1)/2,n ) ) mod n
   // a^b mod n = a · (a^2)^((b-1)/2) mod n

The questions are: If you compute a^b mod n as (((a mod n)*a mod n)*a mod n)...*a mod n) how many multiplications mod n will you have to do? (As a function of |a| and |b|.) If you compute a^b mod n by ExpMOD(a,b,n) how many multiplications mod n will you have to do? (As a function of |a| and |b|.)
[25 %] Question 4. Write a Java program to compute \( a^b \) mod \( n \) for \( a, b, n \) of type long (with \( 0 \leq a, b \leq n - 1 \)) and run it with the values
\[
\begin{align*}
a &= 929351740714 \\
b &= 589394265617 \\
n &= 1624430647349
\end{align*}
\]

The question is: What answer do you get? (if your answer is correct you should notice something...)

REMARK: Do not forget what you did in Question 2.
NOTE: For Questions 2 and 4 you may implement either a recursive or iterative method.

[Extra 10%] Bonus question: Approximate integer square root. The following is a recursive definition of an approximate integer square root.
\[
a\text{Sqrt}(4x) = \begin{cases} 
2 \cdot a\text{Sqrt}(x) & \text{if } x \geq 1 \\
1 & \text{if } 1 < x < 0 \\
0 & \text{if } x = 0
\end{cases}
\]

The questions are:
1. Write a recursive algorithm for a\text{Sqrt}(x). Use floors for integer division.
2. Write a non-recursive algorithm for a\text{Sqrt}(x), i.e. an iterative algorithm.
3. How many recursions are needed to compute a\text{Sqrt}(x), as a function of \( |x| \)?
4. Prove the exact (better than approximate!) correctness of a\text{Sqrt}(x), when \( x \) is an even power of 2, i.e. \( x = 2^{2k} \), for some non-negative integer \( k \).
5. How does a\text{Sqrt}(x) compare to \( \sqrt{x} \), when \( x \) is a non-negative integer that is not an even power of 2?