Computer Science 308-250B Homework #1
Due Monday January 26, 2004, 13:30

On the second class we learned 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) :

```
Mulàlarusse(a,b)
If (b=0)
Then   return 0                                          // 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 )
```

[25%]

•1) Write an iterative definition of this algorithm (no recursion allowed).

Now notice that if we imbed (**mod n**) operators in this definition we end up with an algorithm computing **(a*b) mod n** instead:

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

[30%]

•2) Find the largest number **n** of type **long** for which the method **MulMod(a,b,n)** finds the correct product mod **n** for all **a,b** with **0 ≤ a,b ≤ n-1** and compare it to the direct **Java** expression **(a*b)%n**. Write a **Java** program to find the largest correct values for both methods. **NOTE**: if you get a negative result then your answer is wrong.

[15%]

•3) Find a wiser way to compute **(a+b) mod n** allowing you to reach larger values in **MulMod(a,b,n)** of part **•2).**

Now notice that if we replace "**+**" by "*" in the **<u>MulMOD</u>** algorithm and "0" by "1" in the base case, we end up defining an algorithm for $\mathbf{a^b}$ **mod n** instead of **(a*b) mod n**:

```
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=(a^2)^(b/2) mod n
       Else return ( a * ExpMOD( a*a mod n, (b-1)/2,n ) ) mod n// a^b=a*(a^2)^((b-1)/2) mod n
```

[30%]

•4) Write a **Java** program to compute $\mathbf{a^b}$ **mod n** for **a,b,n** of type **long** (with **0 ≤ a,b ≤ n-1**)
-- REMARK: Do not forget what you did in **•2) --** and run it with the values

a := 1274434408442
b := 589394265617
n := 1606818609763

• What is special about the answer $\mathbf{c:=a^b}$ **mod n** you get?
<u>(if your answer is correct you should notice something special…)</u>

Let **d = 433371342353.** Run your program with the values **c,d,n**.

• What is special about the answer $\mathbf{e:=c^d}$ **mod n** you get?
<u>(if your answer is correct you should notice something…)</u>

NOTE: For questions **•2)** and **•4)** you may implement either a recursive or iterative method.