

Homework 2

1. Give the recursive algorithm for finding the binary representation of a number. Find a representation of number such that adding one to it always takes a fixed number of steps (some constant multiple of $\log \log n$, to be precise). This new representation should be such that given the representation, converting it to the binary representation that we need is easy. (Hint: Adding one to a pointer takes $\log \log n$ steps).
2. Consider long division for finding quotient and remainder when a is divided by b .

Give a recursive and an iterative algorithm for finding quotient $q = q_0q_1q_2 \dots$ given two numbers $a = a_0a_1 \dots a_n$ and $b = b_0b_1 \dots b_m$ (a_0, b_0, q_0 are most significant digits) in some base x (q should also be in base x). The only operations allowed are addition, subtraction and comparison. Do not convert the numbers into any other base.

3. Recall the *recursive* $O(\log k)$ algorithm for *power*(x, k) to compute x^k . Convert this into an *iterative* algorithm, with same running time.
4. We can write $n = \sum_{i=0}^k a_i x^i$, i.e. n in base x . Give an algorithm that finds all a_i 's in $O(\log n)$ time.
5. Prove by induction that $\sum_{i=1}^n i = n(n+1)/2$ and $\sum_{i=1}^n i^2 = n(n+1)(2n+1)/6$.