NYU TANDON SCHOOL OF ENGINEERING

CS-UY 2413: Design&Analysis of Algorithms—Spring 2017 Prof. Boris Aronov
Department of Computer Science and Engineering Prof. Lisa Hellerstein

Homework 3

Due Feb. 21 at the start of lecture (Hellerstein's sections).
Due Feb. 22 at the start of lecture (Aronov's section).
This homework should be handed in on paper, not electronically. No late homeworks accepted. Contact your professor for special circumstances.

Policy on collaboration on this homework: The policy for collaboration on this homework is the same as in HW1 and HW2. By handing in this homework, you accept that policy. Remember: A maximum of 3 people per group.

Notes: (i) Every answer has to be justified unless otherwise stated. Show your work!

(ii) You may use any theorem/property/fact proven in class or in the textbook without re-proving it.

(iii) **Compute all running times in this homework in the bit model.**

1. The following is an exercise in modular exponentiation.

   (a) Let $a_i = 5^{2^i} \bmod 18$. Using the following iterative procedure, compute a_i for $i = 1$ to 4:

   $a_0 = 5$
   for $i = 1$ to 4 do
   $a_i = a_{i-1} * a_{i-1} \bmod 18$

   List the values of $a_0, a_1, \ldots, a_4$.

   (b) The number 23, written in binary is 10111, which is $16 + 4 + 2 + 1$.
   Using that fact, express $5^{23} \bmod 18$ as a function of $a_0, a_1, \ldots, a_4$. (Your expression should be a mod 18 expression.) Then calculate $5^{23} \bmod 18$ using the values for a_i that you computed in the previous problem.

2. When doing ordinary division, if we calculate a/b and get the value x, then x is the solution to the equation $bx = a$.

   Suppose that $GCD(b, c) = 1$. Then the division problem $a/b \bmod c$ has a solution x. The solution x satisfies $bx \equiv a \pmod{c}$ (bx does not have to be equal to a, but they must be equivalent, modulo c).

   Consider the division problem $6/5 \bmod 49$. Verify that it has a solution x and calculate x. Show your work.

   Hint: Check your answer. If your answer x is correct, then it should satisfy $5x \equiv 6 \bmod 49$.

3. As discussed in class and in the textbook, if N is a composite number (that is not a Carmichael number), then *at least* half the integers a between 1 and $N-1$ satisfy $a^{N-1} \bmod N \neq 1$.

   For $N = 6$, what fraction of the numbers a between 1 and $N-1$ satisfy $a^{N-1} \bmod N \neq 1$? List all of them, and specify the fraction.

4. If it takes n bits to represent the number N in binary, how many bits does it take to represent the number $\sqrt{N}$? Choose the correct expression from the list below, AND justify your answer:

   (a) $\Theta(n)$

   (b) $\Theta(\sqrt{n})$

   (c) $\Theta(\log(n))$

5. Describe an algorithm that takes as input an n-bit positive integer a, and determines whether or not it is the product of two consecutive integers, x and $(x+1)$. Analyze the running time of your algorithm in the bit model. It should run in time polynomial in n.

6. As described in the textbook, the probability that a uniformly random n-bit number is prime is approximately $1.44/n$. For this problem, assume that it is is exactly $1.44/n$.

   Note: Give exact numerical answers to the questions below (with a few significant digits). No big-Oh's etc.

   (a) Suppose we generate, uniformly, a random 100-bit number. What is its probability of being prime?

   Note: Just to be clear, a random 100-bit number is generated by executing the following loop:

   for i=1 to 100

   generate a random bit (either 0 or 1, each with equal probability) and assign it to b_i

   Output the binary number $b_1 \ldots b_{100}$

   (b) If we repeatedly generate uniformly random 100-bit numbers, until we get a prime, how many numbers do we expect to generate?

7. In an RSA cryptosystem, choose $p = 13$ and $q = 17$.

   (a) Consider the following potential values for e in this cryptosystem: 6, 7, 3, 4, 9. Which one of them is a legal value for e?

   (b) Using that legal value of e, what is the corresponding d?

   (c) Using your values for d and e, encrypt the message 21.

(d) Using your values for d and e, decrypt the message 3.

8. Consider the following, silly randomized method for searching an *unsorted* array $A[1 \ldots n]$ of integers.

```
function SillyRandomSearch(A,x)
Repeat forever:
    Uniformly and randomly generate an integer i such that 1 ≤ i ≤ n
    if A[i] = x: Return i
```

(a) If x is contained in exactly one position of array A, how many times do we expect SillyRandomSearch(A, x) to execute the repeat loop. Give your answer as a function of n. Do *not* use asymptotic notation in your answer (that is, give an exact bound, not an asymptotic bound).

(b) Repeat the previous question, but this time assume that x is contained in exactly two positions in the array.

(c) Now suppose we replace the Repeat loop with the following:

```
Repeat forever:
    Uniformly and randomly generate an integer i such that 1 ≤ i ≤ n
    Uniformly and randomly generate an integer j such that 1 ≤ j ≤ n
    if A[i] = x and A[j] = x: Return (i, j)
```

If x is contained in exactly two positions in array A, how many times do we expect this new repeat loop to be executed? Again, give your answer as a function of n and do *not* use asymptotic notation.

9. In the RSA cryptosystem, Eve, the eavesdropper, can easily find out Bob's public key (N, e) because Bob has made it public. Suppose Alice sends a message x to Bob, after encrypting it using Bob's public key. Recall that x is a number in $\{0, \ldots, N-1\}$. Let y be the encrypted message that she sends. Eve sees y and would like to decrypt it. But Eve doesn't know how to do it, because RSA is designed to make it difficult for her.

Suppose Eve is lucky enough to find out some additional information I. This information may or may not make it easier for Eve to decrypt y. For each of the following pieces I of additional information, indicate whether it seems to make it easier for Eve, or if it doesn't help. If your answer is "easier," then describe how Eve can decrypt y in time polynomial in n, using N, e and I. If your answer is "it doesn't help," then describe how Eve could compute I herself in polynomial time, just using Bob's public key (N, e).

For example, if I is the decryption key d, then it makes things much easier for Eve, because Eve can compute $y^d \bmod N$ to find the value of x. On the other hand, if I

is the single number $N-1$, then it doesn't help, because she could compute $N-1$ herself by subtracting 1 from N.

(a) a number that is equal to $p-1$

(b) a number r such that $0 < r < (p-1)(q-1)$ and $re-1$ is divisible by $(p-1)(q-1)$

(c) a number that is equal to $(p-1)(q-1)$

(d) a number that is equal to $(p-1)^{p-1} \bmod p$

10. Suppose you have a randomized algorithm $\mathcal{A}$ for testing whether a number x is prime, with the following properties:

- If x is prime, $\mathcal{A}$ will always return "prime"
- If x is not prime, $\mathcal{A}$ returns"not prime" with probability 4/5 and "prime" with probability 1/5

(a) If x is not prime, what is the probability that $\mathcal{A}$ will return the wrong answer?

(b) Using $\mathcal{A}$ as a subroutine, describe an algorithm $\mathcal{B}$ with the following properties:

- If x is prime, $\mathcal{B}$ will always return the correct answer.
- If x is not prime, $\mathcal{B}$ will return the correct answer with probability 124/125.