1 Extended Euclidean Algorithm

Given a, b, find x, y such that $ax + by = d$ where d is the GCD of a, b. This will be necessary in implementing RSA encryption.

Algorithm 1 Extended Euclidean Algorithm

```
1: procedure FINDGCD(a,b)
2:     a ← larger number
3:     b ← smaller number
4:     return ← (d, x, y) such that ax + by = d with d = gcd(a, b).
5: Base Case:
6:     if b = 0 then return (a, 1, 0)
7: Recursive Case:
8:     k ← (a − a mod b)/b
9:     (d, x, y) ← FINDGCD(b, a mod b)
       return (d, y, x − ky)
```

- In each step, k is equal to $\lfloor \frac{a}{b} \rfloor$
- When we recursively find (d, x, y), this satisfies the equation

$$b \cdot x + (a \mod b) \cdot y = d$$

  But we have that $a = kb + (a \mod b)$. Substituting, for $a \mod b$ we get:

$$b \cdot x + y \cdot (a - kb) = d$$

  And after regrouping terms, this gives us:

$$a \cdot y + b \cdot (x - ky) = d$$

- We will use the extended euclidean algorithm for finding *inverses* modulo p, i.e. a is the inverse of b modulo p if $ab = 1 \mod p$. This only exists if $d = \gcd(a, b) = 1$.

Exercise 1. Run the extended Euclidean algorithm on $a = 51$ and $b = 20$. Use the table to help you organize your work. *Hint:* Fill up the table for a, b, k first, then work your way up from the bottom for x and y.

a	b	k	x	y
51	20	2		

2 RSA

RSA is an encryption algorithm loosely based on the assumption that factoring numbers is difficult. While this fact is unproven (we don't even know whether integer factorization is in NP, not to mention P = NP), it seems pretty secure.

The encryption scheme involves having a public key k_e and a private key k_d. When Alice wants to send a message to Bob, she encodes the message with k_e and then Bob decodes the message with k_d. We need to show that (1) It is hard to compute the private key from the public key and (2) It is hard to determine the message m from the encryption of m with the public key without having the private key.

In order to efficiently implement RSA, we need the help of a few algorithms:

- **Repeated Squaring**. This allows us to calculate exponents efficiently.
- **Extended Euclidean Algorithm**. We need this to calculate the private key exponent d from the public key exponent e.
- **Rabin-Miller Primality Test**. Allows us to probabilistically find large primes efficiently.

Creating the keys:

1. Bob finds primes p, q and then calculates $n = pq$.
2. Bob also chooses a number $e < n$ such that e is coprime with $\phi(n) = (p-1)(q-1)$, called Euler's totient function. We often will choose e to be $2^{2^4} + 1$ which is prime.
3. Bob calculates d such that $de = 1 \mod \phi(n)$. This is done using the extended euclidean algorithm where we try to find x, y such that $ex + \phi(n)y = \gcd(e, \phi(n)) = 1$, which would imply that $ex = 1 \mod \phi(n)$.
4. The public key is $k_e = (n, e)$ and the private key is $k_d = d$.

For plaintext m and ciphertext c:

Encryption. $c = m^e \mod n$.

Decryption. $c^d \mod n = m^{ed \mod \phi(n)} \mod n = m$

Exercise 2. Alice generating a new RSA key, and generates the very large, very secure primes $p = 11$ and $q = 29$. She chooses $e = 3$. What is n? What value of d should be used in the private key? What is the encryption of the message $m = 100$?

3 Random Walks

A random walk is an iterative process on a set of vertices V. In each step, you move from the current vertex v_0 to each $v \in V$ with some probability. The simplest version of a random walk is a one-dimensional random walk in which the vertices are the integers, you start at 0, and at each step you either move up one (with probability 1/2) or down one (with probability 1/2).

2-SAT: In lecture, we gave the following randomized algorithm for solving 2-SAT. Start with some truth assignment, say by setting all the variables to false. Find some clause that is not yet satisfied. Randomly choose one the variables in that clause, say by flipping a coin, and change its value. Continue this process, until either all clauses are satisfied or you get tired of flipping coins.

We used a random walk with a completely reflecting boundary at 0 to model our randomized solution to 2-SAT. Fix some solution S and keep track of the number of variables k consistent with the solution S. In each step, we either increase or decrease k by one. Using this model, we showed that the expected running time of our algorithm is $O(n^2)$.

Exercise 3. This weekend, you decide to go to a casino and gamble. You start with k dollars, and you decide that if you ever have $n \geq k$ dollars, you will take your winnings and go home. Assuming that at each step you either win \$1 or lose \$1 (with equal probability), what is the probability that you instead lose all your money?

Exercise 4. Consider the same situation as before. How many steps are expected to occur before you either run out of money or earn n and decide to leave?

Exercise 5. (Challenging) Now suppose that there are $n+1$ people in a circle numbered $0, 1, \ldots, n$. Person 0 starts with a bag of candy. At each step, the person with the bag of candy passes it either left or right with equal probability. The *last* person to receive the bag wins (and gets to keep all the candy). So if you were playing, then you would want to receive the bag only after everyone else has received the bag at least once. What is the probability that person i wins?

4 Linear programming

Objective function (to be minimized or maximized) and constraints, all linear in some variables $x_1, \ldots, x_n$.

Geometric visualization:

- A linear equation of the form $c_1x_1 + \ldots c_nx_n = c$ determines a hyperplane in n-dimensional space.
- A constraint $c_1x_1 + \ldots c_nx_n \leq c$ (or $\geq c$) therefore defines a half-space, which consists of all points to one side of the hyperplane.
- The set of points satisfying all the constraints is given by the intersection of all these half-spaces, which is a convex polyhedron. (You can think about why it has to be convex.)
- The objective function, $a_1x_1 + \ldots + a_nx_n$, can be thought of as a moveable hyperplane: We want to find the highest (maximization) or lowest (minimization) value of α such that the hyperplane $a_1x_1 + \ldots + a_nx_n = \alpha$ still intersects the polyhedron. Therefore the optimum occurs at a corner of the polyhedron, which is why the **simplex algorithm** works.

Exercise 6. Please express the following problem as a linear programming problem: A farmer has 10 acres to plant in wheat and rye. He has to plant at least 7 acres. However, he has only \$1200 to spend and each acre of wheat costs \$200 to plant and each acre of rye costs \$100 to plant. Moreover, the farmer has to get the planting done in 12 hours and it takes an hour to plant an acre of wheat and 2 hours to plant an acre of rye. If the profit is \$500 per acre of wheat and \$300 per acre of rye how many acres of each should be planted to maximize profits?

Exercise 7. Please express the following problem as a linear programming problem: A gold processor has two sources of gold ore, source A and source B. In order to kep his plant running, at least three tons of ore must be processed each day. Ore from source A costs $20 per ton to process, and ore from source B costs $10 per ton to process. Costs must be kept to less than $80 per day. Moreover, Federal Regulations require that the amount of ore from source B cannot exceed twice the amount of ore from source A. If ore from source A yields 2 oz. of gold per ton, and ore from source B yields 3 oz. of gold per ton, how many tons of ore from both sources must be processed each day to maximize the amount of gold extracted subject to the above constraints?