1 Problem 1

Run the Extended Euclidean algorithm for \(a = 111\) and \(b = 75\) to find their \(gcd\), \(d\), and find integers \(x\) and \(y\) such that \(ax + by = d\).

**Solution:** Running the extended Euclidean algorithm, we get:

\[
\begin{array}{c|c|c|c|c}
 a & b & k & x & y \\
 111 & 75 & 1 & -2 & 3 \\
 75 & 36 & 2 & 1 & -2 \\
 36 & 3 & 12 & 0 & 1 \\
 3 & & & 1 & 0 \\
\end{array}
\]

Therefore, \(GCD(a, b) = 3\), \(x = -2\), \(y = 3\). We can verify that \(111 \cdot (-2) + 75 \cdot 3 = 3\).

2 Problem 2

(a) Prove that RSA is multiplicative: if \(C_1\) is the encryption of \(M_1\) and \(C_2\) is the encryption of \(M_2\), then \(C_1C_2\) is the encryption of \(M_1M_2\).

(b) Let \((n, e)\) be a public key of RSA. Show that if we have an efficient algorithm \(A\) which decrypts 1 percent of the messages, we can build an efficient randomized algorithm that decrypts every message with probability > 0.9.

**Solution:**

(a) Follows from definitions: if \(C_1\) is the encryption of \(M_1\), then \(C_1 = M_1^e \mod n\); then \(C_1C_2 = M_1^e \cdot M_2^e \mod n\).

(b) Suppose that \(A\) is a procedure that can decrypt 1/100 of the messages that Alice encrypts. We show a procedure that decrypts every message with high probability: Let \(c(M_1)\) be an encrypted message and we want to compute \(M_1\). We choose randomly a message \(M_2\) and compute \(gcd(M_2, n)\) (recall that \(n\) is public). If it is not 1, then we have succeeded to factor \(n\) and we can decrypt every message. Otherwise look at the message \(M = M_1M_2\). Since \(M_2\) is chosen uniformly from all the messages that have \(gcd(M_2, n) = 1\), and since \(gcd(M_2, n) = 1\), \(M\) is uniformly distributed over the space of all messages.
Therefore with probability 1/100 A can decrypt \( c(M) \). We now use the fact that \( c(M) = c(M_1)c(M_2) \). We compute \( c(M_2) \) and multiply it by \( c(M_1) \) we now have \( c(M) \). We now give \( c(M) \) to A. Suppose A computes \( M \) for us, so we divide it by \( M_2 \) and retrieve \( M_1 \). We have shown that with probability 1/100 we have succeeded to decrypt every message. Then, we just need to find a \( k \) such that \( (1 - 0.01)^k < 0.1 \). The smallest such \( k \) is 230.

3 Problem 3

(a) We are filling a knapsack with items. The knapsack has a maximum weight capacity of \( W \). Each item \( i \) is associated with a value \( v_i \) and weight \( w_i \). We can split any item \( i \) into a fractional piece before putting the fractional piece into the bag (e.g. we can take \( \frac{1}{2} \) of item 1). There are \( n \) total items. Describe a linear program to find the maximum value knapsack we can fill.

(b) Now suppose the knapsack is shared between \( k \) different people who have different opinions about the values of items. For person \( j \) (\( j = 1, 2, ..., k \)), the value of item \( i \) is \( v^j_i \geq 0 \). For each way of putting items into the knapsack, different people will have different evaluations for the knapsack (\( p \) fraction of item \( i \) will contribute \( p \cdot v^j_i \) to the evaluation of person \( j \)). We want to be fair to all the people, so we will define the actual value of the knapsack to be the minimum evaluation out of the \( k \) people. Write a linear program that finds the maximum possible value of a knapsack.

**Solution:**

(a) Let \( p_i \) be the fraction of item \( i \) we insert into the knapsack, \( w_i \) and \( v_i \) be the total weight and valuation of the item respectively. Then:

\[
\max \sum_{i=1}^{n} p_i v_i \\
\text{s.t.} \sum_{i=1}^{n} p_i w_i \leq W \\
0 \leq p_i \leq 1
\]

(b) Similar to above, except now we define the utility of person \( j \) to be \( u_j = \sum_{i=1}^{n} p_i v^j_i \). Then:

\[
\max u \\
\text{s.t.} \ u \leq u_j \forall j \\
\ u_j = \sum_{i=1}^{n} p_i v^j_i \\
\sum_{i=1}^{n} p_i w_i \leq W \\
0 \leq p_i \leq 1
\]