RSA
Dr. Ostheimer, CS 14, Fall 2023

Step I: Receiver chooses keys.

1. Choose two large prime numbers p and q, and let $n = pq$.
2. Choose B so that $n > 2^B$.
3. Compute $m = (p - 1)(q - 1)$.
4. Choose e and d, multiplicative inverses mod m.

Step 2: Receiver publishes encryption keys B, n and e for all to see.

Step 3: Sender encrypts messages and sends to receiver.

1. Sender converts text to a list of decimals $[m_1, m_2, \ldots, m_k]$ using pre-processing and B (described in separate handout).
2. Sender converts to list $[c_1, c_2, \ldots, c_k]$ as follows: $c_i = m_i^e \bmod n$.
3. Sender sends encrypted decimals $[c_1, c_2, \ldots, c_k]$ to receiver.

Step 4: Receiver decrypts message.

1. Receiver converts encrypted decimals $[c_1, c_2, \ldots, c_k]$ to $[m_1, m_2, \ldots, m_k]$ as follows: $m_i = c_i^d \bmod n$.
2. Receiver converts $[m_1, m_2, \ldots, m_k]$ back to text using post-processing and B (described in separate handout).

Eavesdropper The eavesdropper needs to know m in order to use the fast algorithm to compute d from e. In order to compute m, he needs p and q. Thus, the eavesdropper needs to factor n. Since we believe that the Factoring Problem is not in $\mathcal{P}$, we think that we are safe telling the eavesdropper e and n. This is a **public key** system.

Extra Credit. Prove that RSA works!