# Problem 1

# Problem 2

1. For key generation in RSA, you start by picking two large prime numbers $p$ and $q$ (these should be large enough that the product of $p \cdot q$ is on the magnitude of $2^1024$, this makes it infeasible to factor $p \cdot q$). The product of $p$ and $q$ is one part of the public key, which we will call $n$. You then calculate $\phi(n) = (p-1)(q-1)$. You then pick an $e$ which is relatively prime to $\phi(n)$ ($\phi(n)$ and $e$ do not share any common factors besides 1). This $e$ is also a public key. You then find a $d$ such that $ed = 1 \pmod{\phi(n)}$. This $d$ can be found using the extended Euclidean algorithm. $e$ and $n$ are both public keys, while $d$ is a private key kept only by the person who created the system.
   Encryption is done by raising the message to the power of $e$ modulo $n$: $C = m^e \pmod{n}$.
   Decryption is done by raising the encrypted message to the power of $d$ modulo $n$: $m^{ed} = m^{t \cdot \phi(n)+1} = m \pmod{n}$

2. Yes RSA encryption is secure against a known plaintext attack, as long as $n$ is sufficiently large. To discover the key $d$, the attacker would have to compute $\phi(n)$ which is as difficult as factoring $n$. Factoring $n$ is infeasible for an $n$ on the order of $2^{1024}$.

# Problem 3

# Problem 4

# Problem 5

# Problem 6