Task 2:

3. The private key was calculated in two steps. First a pair of factors, p and q, were found for the public key n by taking the square root of n and checking every positive integer smaller than sqrt(n) to see whether it divided n with a remainder of 0. Once found, that integer became p, and n / p became q. Second, given e and having just calculated p and q, the extended Euclidean algorithm was used to find the modular inverse of e mod $(p - 1)(q - 1)$, which equals the private key.

Task 3:

4. An RSA public key, N, is constructed by multiplying two integers, p and q, of arbitrary length together, to produce a number which is used in conjunction with an exponent, e, to encode a message, m, via the formula m^e mod N. Factoring an RSA public key is impossible to do efficiently; the largest known factorization is for one of 768 bits. As such, a 1024-bit public key should theoretically be unfactorable. However, while calculating the factors of N runs in exponential time, calculating the greatest common divisor (GCD) of two keys, N_1 and N_2 can be done in logarithmic time. Because the GCD can be rapidly calculated, the factors of any two key that share a factor larger than 1 can both be found very easily, by computing the GCD (the first factor) and then dividing the public key by that GCD to get the second factor.

5. The private key is found in two stages. The first stage identifies Waldo, a key which shares a factor > 1 with our key. This is done by computing the GCD for our key and all other keys, and terminating once a GCD > 1 is found. The second stage divides the key N_1 by the GCD to get q, the second factor (while the GCD = p, the first factor). The formula $d = e^{-1} \bmod (p-1)(q-1)$ is then used to calculate the private key d.

Task 4:

2. This RSA broadcast attack is a simple version of Håstad's broadcast attack, which allows an attacker to compute the message text m from a set of encoded ciphertexts c_i which all use the same exponent e. Specifically, so long as e or more ciphertexts (and their associated public keys n_i) are known, the original message m is no longer secure. Assuming the GCD of all public keys is equal to 1 (otherwise we can use the GCD method in Task 3 to find the message m), we can use the Chinese remainder theorem to compute another ciphertext, c_{i+1} such that $c_{i+1} = m^e \bmod (N_1 N_2 \ldots N_i)$. This being the case, m can be found very simply by taking the eth root of the result of the ciphertext c_{i+1}. In this case, $e = 3$, so there are three public keys, three ciphertexts, and we take the cube root of the result of the Chinese remainder theorem to get m.

3. To recover the original message m, I simply applied the version of Håstad's broadcast attack described above, where $e = 3$. Given the three public keys and three ciphertexts, I found a fourth ciphertext using the Chinese remainder theorem, then found the cube root of that, which was the

message *m.* This was done using the helper functions mul_inv(), which found the multiplicative modular inverse, chinese_remainder(), which found the fourth ciphertext, and find_invpow(), which calculated the cube root.