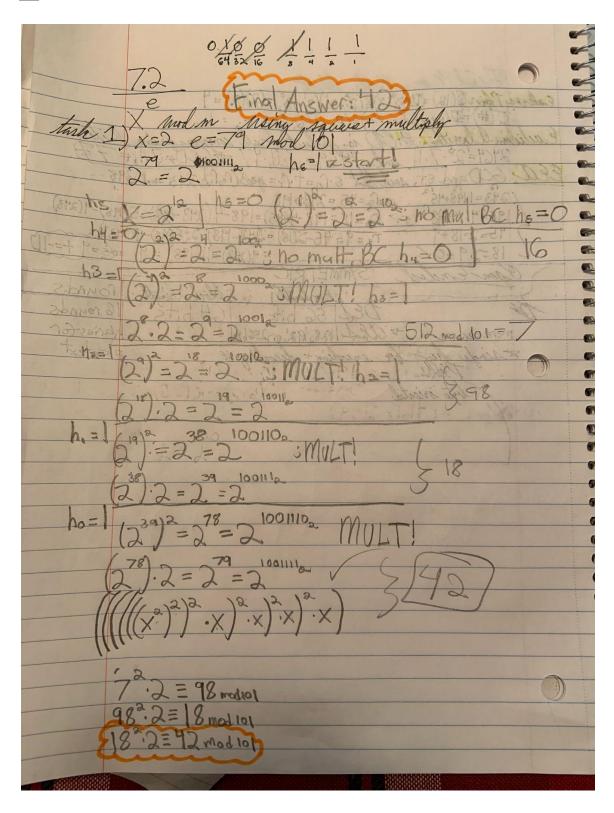
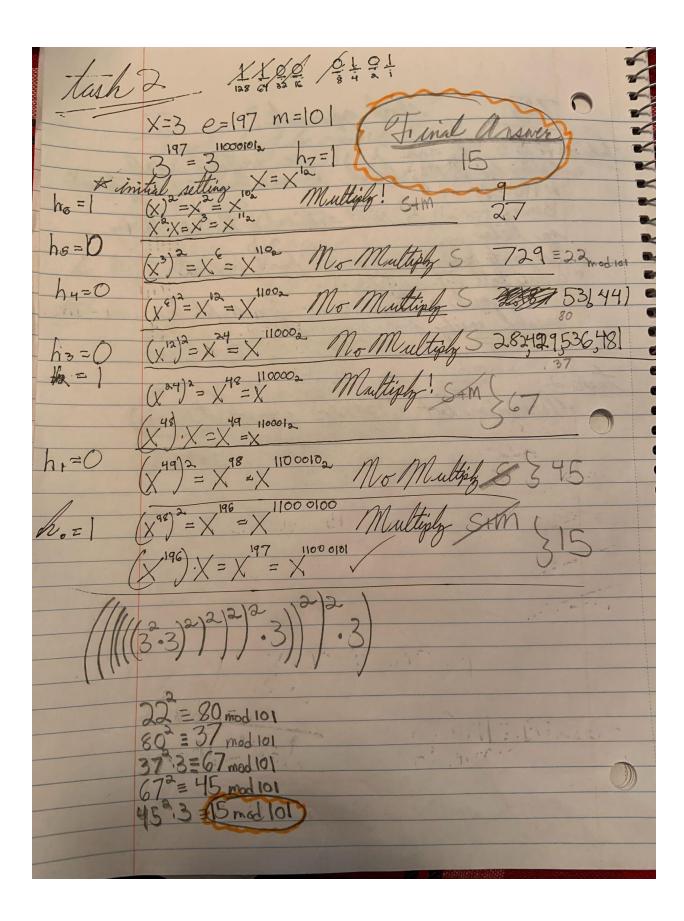
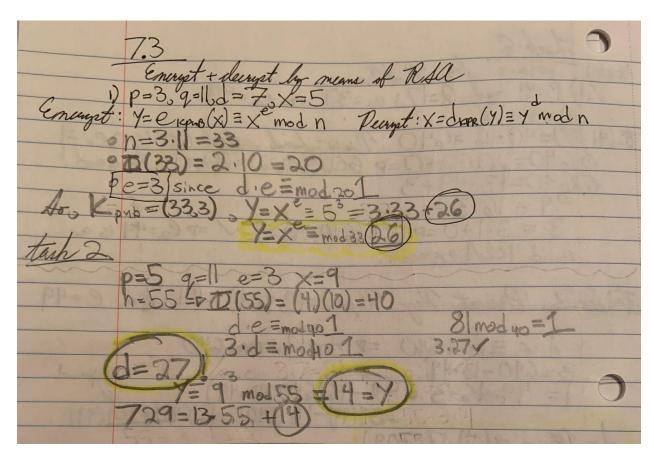
Carl Cortez CIS 628 Chapter 7 Lab 8

> Carl Cortez CIS 628 Chapter 7 Lab 8

= task1 primes for RSA = 7.1) P=41 and Q=17; e=32 or e=49
3 = 3.1+0=> GCD(640,49)=1 V=P6=49 is a
tash 2 Private New Mpr = (P. 90d) = (4617 d) e=49
d'e=mod 640 = V d.49=mod 640 3=640-13.49 1=49-16.3=49-16(640-13.49) 209(49)-16(640)=-16(640)+209(49) Aos News = (41)-17,209)







<u>7.5</u>

In practice the short exponents e = 3, 17 and 216 +1 are widely used. 1. Why can't we use these three short exponents as values for the exponent d in applications where we want to accelerate decryption?

Since these are, "widely used," I would assume that people are aware of these short exponents.

This reminds me of the time I hacked free wi-fi during college. I took some password guesses on a neighboring wi-fi network and had success with *password* as the password. A simple guess on ideal passwords/solutions could be used in this dynamic. It's not wise to use common passwords or in this case short exponents because they can be easily guessed and brute forced.

2. Suggest a minimum bit length for the exponent d and explain your answer.

From page 184 of the textbook, they mention 128 bits being a larger number to avoid brute-force. I imagine this is a good start for bit lengths of exponent d. If we were to exceed 128 bits, I imagine that would improve the security even more.

7.11

In this exercise, you are asked to attack an RSA encrypted message. Imagine being the attacker: You obtain the ciphertext y = 1141 by eavesdropping on a certain connection. The public key is kpub = (n,e)=(2623,2111).

1. Consider the encryption formula. All variables except the plaintext x are known. Why can't you simply solve the equation for x?

Initially, I thought an answer could be retrieved using properties of logs. However, these outcomes for x^e can be congruent to a variety of outcomes.

2. In order to determine the private key d, you have to calculate $d \equiv e^{-1} \mod \Phi(n)$. There is an efficient expression for calculating $\Phi(n)$. Can we use this formula here?

We're lucky enough to know what \mathbf{n} and \mathbf{e} are. However, from the equation of $\Phi(n)$, we need to know the two prime numbers used in the product (p and q). Since we don't have those, we

cannot use the formula here.

3. Calculate the plaintext x by computing the private key d through factoring $n = p \cdot q$. Does this approach remain suitable for numbers with a length of 1024 bit or more?

