1. [Task 2]A brief description about your steps to get the private key

In order to obtain the private key, the following steps were required.

First, I found my public key by running the provided execution command (python get_pri_key.py ykim691).

The next step was to find the two prime numbers. Knowing that the equation for the modulus of N is the product of two prime numbers, p and q, then if I find one prime the other should be followed by a simple division calculation.

The next step was to search for the fastest **factoring** algorithm. While there are multiple algorithms, the first algorithm that I tried was “Trial division”. However, its return time took 200 seconds which was higher than my goal of 30 seconds. I tried another algorithm called “**Pollard’s rho**”, and applied it in my source code. It was successful as the return time was less than a second. So far, the public key N, one of its prime p through the factoring algorithm, as well as q via simple division have been found.

Lastly, I started working on finding the private key using the two primes p and q. In the introduction of this project, the formula generating private key was given.

$$d \equiv e^{-1} \mod \phi(N)$$

The final step was applying **modular multiplicative inverse** to $\phi(N) = (p-1) * (q-1)$. I found the “**Extended Euclidean**” algorithm is commonly used for this purpose. Once I applied it into the code, I found the private key.

Specifically given the public key, N (0xcd62c4cf8e961cd) and the exponent, e (0x10001), I performed the factoring and modular multiplicative inverse on N to obtain the private key d(0x418932A17ED615).

2. [Task 3-1] Your understanding about the weak key problem caused by Ps and Qs

The vulnerability of RSA is the fact that the public key, N, is a product of the two primes p and q. This means N only can have P and Q as its primes excluding 1 and itself. When the other public key shares its prime, it will be either P or Q and that gives easy access to the other non shared prime by division of public key N by the known shared prime number.

Q. How to know if these two public keys share a prime?
Because the two public keys are moduli, checking if the Greatest Common Divisor of the two is bigger than 1 can determine whether the two public keys share a prime or not.

> Let's say public key N1 has its primes [1, p1, q1, N1], and another public key N2 has its primes [1, p2, q2, N2].
> ⇒If GCD(N1,N2) = 1, it means two keys do not share any prime, because 1 is common prime for every number.
> ⇒If GCD(N1, N2) > 1, it means two keys share a prime because the public key has only two primes.

Once the two primes are determined, the attacker easily obtains the private key by modular multiplicative inverse.

3. [Task 3-2] A simple description about your steps to get the private key
This task divides into two sections. One is finding Waldo, and the other is looking for the private key.

In the first section, the important point of solving this problem is understanding the concept of the weak key problem when two keys share a prime. In order to find out a public key which has the same prime with mine, I used the function Greatest Common Divisor(GCD). We come to the conclusion that if GCD is bigger than 1, it shares a prime with my public key, because every number has 1 as its prime. There was only one key which has GCD greater than 1, and it is my Waldo.

Second section is generating the private key. Using GCD on my public key and Waldo's, the prime P was found. The other prime Q was automatically computed by the division of N by P. Having found P and Q, I found the private key using modular multiplicative inverse.