1. As mentioned in the question, when we send the ciphertext c, it returns the corresponding plaintext m. However we cannot able to get corresponding plaintext of the given ciphertext. Therefore, we should modify given c (say c_) such that after we get corresponding plaintext (say m_) we should be able to return back to m which is original plaintext that we are expected to find. For this purpose, we can choose a random number r (i choose 13 but it doesn't matter) and raised with power of e because actually we do it to the original message in encryption. Then, we create modified ciphertext c_ with multiplying with the r^e in modulo N. The operations that we were done can be understood better if we investigate what server will do after we sent the modified ciphertext:
    - For decryption, server should raise my message with power d.
    - (Since c ≡ m^e, what we are sending to the server actually; (r^e * m^e) % N)
    - Server should return (r^e * m^e)^d (modN).
    - Since e * d = 1 (mod N), (r^e * m^e)^d (modN) = r*e (modN). This means that server will return us r*e (modN).

That means that r*e (modN) = m_. We can return back to original message easily from this point. We just need to calculate inverse modulo of random number r. Therefore we can obtain message with the following code: "message = (m_ * inv_r) % N". After sending byte array of corresponding message to the server, I got congrats message. Message was "Bravo! You find it. Your secret code is 75963".

2. In RSA OAEP, we get different ciphertext even if we use same plaintext, because random numbers are used. In this question, this random number R is given as 8, which is very small, so we can make exhaustive search on R. Moreover, the PIN is a 4 digit number which means we can make exhaustive search for PIN as well. Therefore, we can try all possible Rs for all possible PINs. For each possibilities we should perform encryption operation and compare with

the c given in the question. If they are equal in an iteration, this means that corresponding values of that iteration (i.e. PIN and r) give us the PIN and R. I obtained PIN: 9495 and r: 255. When PIN was sent to the server, it returned congrats.

3. We can find the session key with exhaustive search. I found k as 64278. We know that t = h^k * m (mod p), this means that m = t * (h^k)^-1 (mod p).
4. I cannot recover.
5. Since the r values are same, session key is also same. So that we can find private key with the following calculation: private_key = ((s2*h1 - s1*h2) * modinv(r * (s1 - s2), q)) % q

6. Encryptor said that “I ran out of random numbers for the signature of the second message”, this means that, session keys can be related each other i.e. k2 = i*k1. The only way that comes to my mind in order to find ‘i’ was iterating over some values. In each iteration private key is calculated by the following code: private_key = (s1*h2 - s2*h1*i) * modinv((s2*r1*i)%q - (s1*r2)%q, q) % q. After calculating private key in an iteration, we check the following condition is hold; g^private_key (mod p) is equal to Beta, if that is true this means that we found the private key, otherwise we continue.