1. I've first found an integer that is relatively prime to the N given by the server using the egcd function from myntl.py. Say this integer is x. Then found the $x^e \ mod(N) = y$ and created new $\overline{c} = c * y \ \ mod(N)$ which is different from c and query RSA Oracle with $\overline{c}$. From query we get $\overline{m} = \overline{c}^d \ mod(N)$. Since $\overline{c} = c * y \ \ mod(N)$ then $\overline{m} = (c * y)^d \ mod(N)$ and $c^d = message$ then $\overline{m} = (c)^d * y^d \ mod(N)$ and $y^d = x^{e*d} \ mod(N) = x$. If we get the inverse of x mod(N) using modinv from myntl.py and multiply it with $\overline{m}$ mod(N) we get $c^d = message$. $message = x^{-1} * \overline{m} \ mod(N)$. Then convert the message into bytes and send it to the server as res. The message is **b'Bravo: you find it. Your secret code is 62569'.**

2. Random number R is an 8 bit unsigned integer and the PIN is in between 1000 to 9999. Since both R and PIN have small range we can try all the R's and for all R's try all the PINs. We need to find R, to get the R, first we equalize R to 2^7 and increment it in a while loop. So for each possible 8 bit unsigned integer we can try all the possible PINs and encrypt these possible messages using corresponding R and public values and check if it's equal to c. If for that R we couldn't find the PIN then we move onto the next possible R which we get by incrementing by one. Then for this R we try all the possible PINs in between 1000 to 9999 and encrypt the possible PINs with R and public values and check if it's equal to the ciphertext c. If the found ciphertext is equal to the given ciphertext then we have found the R value and the PIN. In the end **I have found R as 194 and PIN as 7821.**

3. The flaw in the python code is that the range for k is small, it's between 1 to $2^{16} - 1$. So I have tried the possible k values and $g^k \ mod(p)$ and checked if it's equal to the r. We also know that $t = h^k * m \ mod(p)$. Then when we find out the correct k we can calculate the message as $m = (h^k)^{-1} * t \ \ mod(p)$. **I have found the k as 31659 and when I convert the message integer to bytes I have found the message as b'Why is Monday so far from Friday, and Friday so close to Monday?'**

4. The given two signatures' r values are the same which means we have used the same session key during signature processing. Then we can find the private key of the signature owner as $a = (s_i h_j - s_j h_i)(r(s_j - s_i))^{-1} \bmod q$. I've calculated the hash values of the messages and I've calculated the private key a as mentioned above then **I've found a= 168874198460519327134644531443752111733505626315532547031556139226 71**

5. Bonus: From the hint of the question I've thought that since hoca ran out of the random numbers during signing the second message, he may use a session key which is related to the first signature's session key. So for that reason I have created a while loop that goes through all the possible multiples of the first session key. In the while loop I have calculated the private key a for each of the possible multiples as in the formulation from the class: $a = (s_i h_j - s_j h_i x)(s_j r_i x - s_i r_j)^{-1} \bmod q$. Then I have checked the beta value as if the given beta value is equal to $g^a mod\ (q)$, if it's equal then we have found the private key. **The private key that I have found is**

38468022319344408234287699540778097655787016963246135508538566407 2 and the second session key is 127 * first session key.