CS411 Homework #3
Hakan Buğra Erentuğ 23637

1a) What we should do first, writing the eqution $-x^8 = x^4 + x^3 + x + 1$ and derived from them find all power of x until x^{14}. Then we should the polynomial multiplication which provided by https://www.symbolab.com/solver/polynomial-multiplication-calculator easily. Then, we should eliminate the coefficients with 2 and cover all terms with the power of more than 7 by the equations which is the given in the question. The result is "$x^6 + x^5 + 1$". Unfortunate, I cannot append any python code to show the process due to most of the solution is coming from paperwork.

1b) For that question I found a online solution and implement that to our question. Please check, one.py for the solution. What we do is basically taking inverse of P(x) and multipying it with itself in modulo p(x). Still, a lot of hand and paper work is required but I have successfully managed to express them into python code. Please also check the online resource in the http://pythonfiddle.com/binary-finite-field-multiplication/ . Still, the same multiplication process have been following in the question.

2a) Please check two.ipynb for detailed expiation but briefly, until we get a pair in rainbow table, we need to make reduction and hash the value again. After finding the pair, we need to check if the first element of the pair is in the chain. The followings are the outputs and passwords.

```
digest[0]= RSURBK
digest[1] case2...
digest[1] KJMB?D
digest[2] case2...
digest[2] LHBTKY
digest[3] case2...
digest[3] DQNQMO
digest[4]= YQLLZB
digest[5] case2...
digest[5] UPYC!L
digest[6] case2...
digest[6] EA?!WT
digest[7] case2...
digest[7] !BURIB
digest[8]= EOAFLL
digest[9]= TVGH.O
```

3a) It is not secure, because it can be decrypted by greatest common divisor formula. Before to starting the explain I would like to remind you p and q are prime. Which means the $c_p = (kp)^e$ can be factored by only k and p. I would also remind you to p*q is n which we already have. The basic of deception are the followings…

The n's factorizitaion is p and q number
The c_p's factoriezaion is k*p*k*p*k*p*k…. (e times).
SO if we take egcd of n and c_p we would obtain the p which is the only common element. Also, we can obtain q by dividing n with p.
The rest of the decryption is easy and not asked to explain however, the b section of the question illustrates how easy to decrypt after the factorization of n.

3b) Please check the DeterministicRSA.py to check understand and verify the code. Just like the method above, we can easily factorized p and q. The p and q primes and output of the code is

CS411 Homework #3
Hakan Buğra Erentuğ 23637

The first prime:
45931572870827881561956359348907507610206389661943955042830107463735662349972722791061733124489189272240130584647783528115980820171539864787746408093357313112739577731204137447836378064314806894823016698356673012051408104260148385313472774949091208519062499104153730513034259359276259882037652364138616878279

****

The second prime:
162714425555590299907582727396980097499459248735262849323615961586880482025872588759372594209176495245117290424311606087206252036379802275425817797958224085089194159244862342658014419330000145704525568894322127427547322440610723007470284010301550512933411159828723823852019169776225493115796888791016517378863

The rest is calculating phi which is (p-1)*(q-1) and taking inverse modulo of e. Because in the RSA to obtain decryption key we have to obey and apply “e x d = 1 mod(phi(n))” formula.

The result of inverse modulo is d and cipher^{d} = message mod(n). When we cast the bytes to string we obtain: Message: b'**Insanity is doing the same thing, over and over again, but expecting different results.**'

4a) This question is highly relevant with {0,1} game which stated in lecture. Luckily, the RSA is a deterministic algorithm. Which means same input gives the output always. We have a function that decypte the cipher to plaintext, however, it is forbidden to use the original cipher. Therefore, calculating another ciphertext which gives the same result and decypting them is enough. To decypt we have to find {c * c * c * c..} {d times} in modulo n. But what if we take c+n as a new c’. Therefore the $(c+n)^{d}$ will give $c^{d} + c^{d-1}*n + c^{d-2}*n^{2}….+n^{d}$ in modulo n will eliminate all terms with n coefficient and result in $c^{d.}$. The decryprion of c + n will give the message m.

4b) I did not write rather than adding c + n in python console and copying and pasting to RSAOracle.exe. The result is casting from bytes to string and result is, You discovered my verry secret message:) Bravo