1a) Since 61 is prime, number of generators are equal to φ(61-1).

=16

As you can see in my hw2\_1.py folder, I calculated the 16 of the generators and they are 2,6,7,10,17,18,26,30,31,35,43,44,51,54,55,59.

1b) As also you can see in my hw2\_1.py folder, my program calculated the 5th order of subgroup elements. They are 1,9,20,34,58. And the generators are 9, 20,34,58.

2)Since p and q are prime φ(n)= (p-1)(q-1). you can see my calculations in hw2\_2.py folder.

d=17780486146994356390536698036064157558306067628911641314354623865206367540771264269376489317932941129613874729594828696361809009003718595069818719707531861843667554232045010862980569632311263598996529228578729062647856589914115371678446914411535684158634995069502339626673711631508561804630291588176843797867

and

m= 30256242323116471143377579036851734956599566784957808777900084159344311403348077243305468893900944356797865493742870723640786398026030923177760147308711037452403389422258824881485692843121042180280211955250327896263032672055294111900141070981523217773015658586930923177732713966172972763510074059270940820416

3) If we want to find x we need to inverse of a. When we multiply both side of the equation by inverse of a our equation becomes x mod n.

As you can see my hw2\_3 first and second a don’t have any inverse so first and second x are can not find. For the third values we can calculate inverse of a and when we multiply both side by the inverse of a we can get the x value.

X= 327252728639173874206458501252

4) For

p1(x) = x5 + x2 + 1, Let our initial state be 00001;

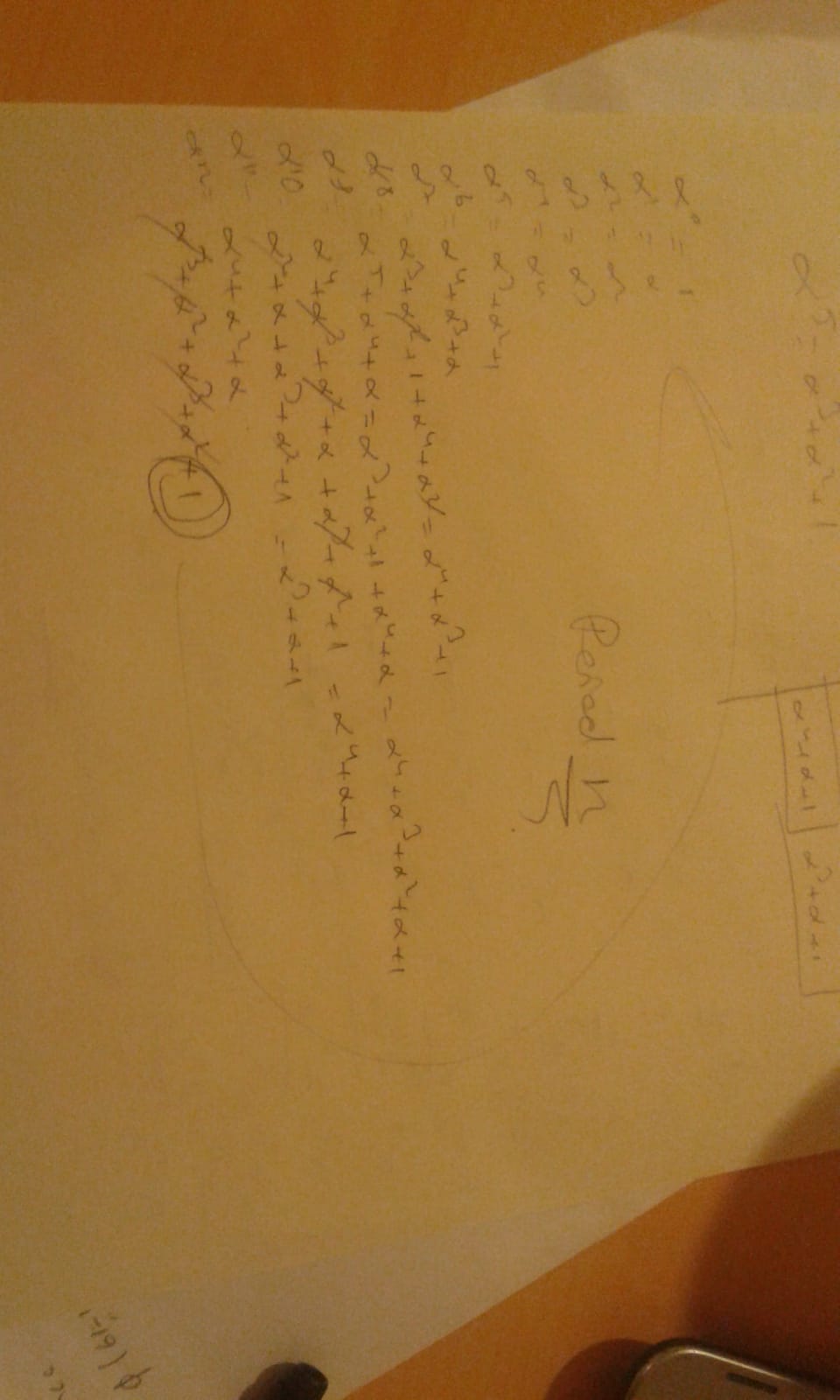
|  |  |
| --- | --- |
| i=0 | 00001 |
| i=1 | 10000 |
| I=2 | 01000 |
| I=3 | 10100 |
| I=4 | 01010 |
| I=5 | 10101 |
| I=6 | 11010 |
| I=7 | 11101 |
| I=8 | 01110 |
| I=9 | 10111 |
| I=10 | 11011 |
| i-11 | 01101 |
| I=12 | 00110 |
| I=13 | 00011 |
| O=14 | 10001 |
| I=15 | 11000 |
| I=16 | 11100 |
| I=17 | 11110 |
| I=18 | 11111 |
| I=19 | 01111 |
| I=20 | 00111 |
| I=21 | 10011 |
| I=22 | 11001 |
| I=23 | 01100 |
| I=24 | 10110 |
| I=25 | 01011 |
| I=26 | 00101 |
| I=27 | 10010 |
| I=28 | 01001 |
| I=29 | 00100 |
| I=30 | 00010 |
| I=31 | 00001 |

As seen in the above p1(x) s period is 31 which is maximum number of period(). Since p1(x) also a primitive polynomials, maximum period is expected.

For p2(x) = x5 + x3 + x2 + 1, Let our initial state be 00001.Since this polynomial is not primitive, expected period is less than the maximum period. p2(x)’s period is 12 when we select the initial state as 00001. However as you can see in my hw2\_4.py folder, when you change the initial state, the period of polynomial changes. Period becomes 1,2,3,4,6and 12 which are the factors of 12.

|  |  |
| --- | --- |
| I=0 | 00001 |
| I=1 | 10000 |
| I=2 | 01000 |
| I=3 | 10100 |
| I=4 | 11010 |
| I=5 | 11101 |
| I=6 | 11110 |
| I=7 | 01111 |
| I=8 | 10111 |
| I=9 | 01011 |
| I=10 | 00101 |
| I=11 | 00010 |
| I=12 | 00001 |

When we try to solve period of the polynomial by writing the x^3+x^2+1 instead of x^5, we find the period 12 again.



5)In my hw2\_5.py code, I found the period of each sequence and each of them are 100.Secondly, I found the number of 0s and 1s for each x values. For x1 and x2 number of 0s are 59 and number of 1s are 41. So, number of 0s and 1s are unbalanced for x1 and x2. However, number of 0s are 48 and 1s are 52 in x3. As a result of this, we can say x3 is balanced. Thus we can say that x1 and x2 are predictable, x3 is not.

6) We know ho the message starts. When we convert “Dear Student”to binary sequences we get

[1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0]

If we xor this binary sequence with first 84 bit of the given cipher text we get the first 84 bit of key which is [1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1]

We know the c(x) of this key by the BM algorithm and I found that by using the function in lfsr.py. We get the c(x) as [1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1] which is 1+x^3+x^17. So we understand that our key is produced by this polynomial. We need to know the initial state and we can get the initial state with the reverse order of first 17 bit of the key which is [1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1].

So our message is:

Dear student,

You have worked and that paid off :)

You have just earned 20 bonus points. Congrats!

Best,

Erkay Savas

['D', 'e', 'a', 'r', ' ', 'S', 't' , 'u', 'd', 'e', 'n', 't', ', ', ' ', '\n', 'Y', 'o', 'u', ' ', 'h', 'a', 'v', 'e', ' ', 'w', 'o', 'r', 'k', 'e', 'd', ' ', 'h', 'a', 'r', 'd', ' ', 'a', 'n', 'd', ' ', 't', 'h', 'a', 't', ' ', 'p', 'a', 'i', 'd', ' ', 'o', 'f', 'f', ':', ')', ' ', '\n', 'Y', 'o', 'u', ' ', 'h', 'a', 'v', 'e', ' ', 'j', 'u', 's', 't', ' ', 'e', 'a', 'r', 'n', 'e', 'd', ' ', '2', '0', ' ', 'b', 'o', 'n', 'u', 's', ' ', 'p', 'o', 'i', 'n', 't', 's', '.', ' ', 'C', 'o', 'n', 'g', 'r', 'a', 't', 's', '!', ' ', '\n', 'B', 'e', 's', 't', ',', ' ', '\n', 'E', 'r', 'k', 'a', 'y', ' ', 'S', 'a', 'v', 'a', 's']