1 Problem 1

1. Yes. The polynomial is irreducible over $GF(2)$, and it generates a maximal length LFSR sequence of length 255.
2. The maximum cycle length generated by $x^8 + x^4 + x^3 + x^2 + 1$ is $2^8 - 1 = 255$. We can use the following program to verify the answer:

```
key = [0, 0, 0, 0, 0, 0, 0, 1]
for i in range(1, 1000):
    if key == [0, 0, 0, 0, 0, 0, 0, 1]:
        print(f'{i}: {key}')
    tmp = key[0]
    for j in range(7):
        key[j] = key[j + 1]
    key[7] = 0
    if tmp:
        key[3] = 1 ^ key[3]
        key[4] = 1 ^ key[4]
        key[5] = 1 ^ key[5]
        key[7] = 1 ^ key[7]
```

Output:

```
1: [0, 0, 0, 0, 0, 0, 0, 1]
256: [0, 0, 0, 0, 0, 0, 0, 1]
511: [0, 0, 0, 0, 0, 0, 0, 1]
766: [0, 0, 0, 0, 0, 0, 0, 1]
```

3. Not all irreducible polynomials are primitive polynomials. For example, $x^4 + x^3 + x^2 + x + 1$ is irreducible over $GF(2)$, but it is not primitive because it does not generate a maximal length LFSR sequence of length 15.

```
key = [0, 0, 0, 1]
for i in range(1,20):
    if key == [0, 0, 0, 1]:
        print(f'{i}: {key}')
    tmp = key[0]
    for j in range(3):
        key[j] = key[j + 1]
    key[3] = 0
    if tmp:
        key[0] = 1 ^ key[0]
        key[1] = 1 ^ key[1]
        key[2] = 1 ^ key[2]
        key[3] = 1 ^ key[3]
```

Output:

```
1: [0, 0, 0, 1]
6: [0, 0, 0, 1]
11: [0, 0, 0, 1]
16: [0, 0, 0, 1]
```

2 Problem 2

1. output:

```
Cipher text: 010000000101011001001010010100010101001101110101000101111100010
101011100011010000011000110111011100110011101010101011010011100000000101110
101100110101000001110111110110011011110101111100010001100100001010001010000
1
101001101010011000110010100101110100010011100101101100010000111001100111101
1
011000001111011001101011000010010001010010101111000010010001110111100100000
0
011010001011100000000010111001001010010110100111001000111011011000001100111
1
100110001111011110011001110011001111011110000000100001100111111111001100000
11
011111010001010001001010011010000011010110001101111101111001100011000010111
0
110100100000000101001001101010001111011010010010110101101000101001101011110
10
100101000100001001111010000111110101011011010100010010001111001000000101111
11
011110011001010101101100101001111111100001111110000110100100101110010000101
1
011000111010011110101011001110001001001011000001111101001001011111000111111
0
101010100101101110110001110011010111110100010100101011101111110111111001100
0
010101101100001101101001001100000000110001100010011101001001011110011000100
00
100001111101100100010100101001111110100000110110100000111010000000111110100
0
001011010011010000000100011110111100011000110100110110110100010111111000000
00
101111111001100011101110110100111110000111011101000000100100001111011000100
11
110101110100100001110101100110100100000111101001001111110001100001011110110
1
001101001111111111010001111010100000010110100101011001010110010010110111011
1
000011111110111100111100000101100100000111101100001100010001011011001010011
1
000100000001011110101100000000011010110010111000110000000110000100001111100
101
001000001001011010010101000111110100100010101101001001010111100110000111011
11
100010001010011100001000110001111110000110000101111001011110110100110100101
0
111010010011011110000001111101101000010011100011001010000010100110110001000
0
111001101010101101111001110111011100010111001101000111011101110111010101111
1
110011100110010100101100001000010011001000001110111111001001110011011111110
1
010001001101111001011010110100110001000100000100111011110100100010011110000
0
101010101010010100100000101001011010000100110010100000000111010100010101000
11
101010011101010100111100010111100100001000110011101000001100111010110011001
0
000001100111100010010011110111001100000001101101101011010010011110100001001
0
011000111100000010010110000110110101000111010100011101110010000111010100000
11
0110101010111011000101111

Decrypted text: ATNYCUWEARESTRIVINGTOBEAGREATUNIVERSITYTHATTRANSCENDSDISCIPL
INARYDIVIDESTOSOLVETHEINCREASINGLYCOMPLEXPROBLEMSTHATTHEWORLDFACESWEWILLCONT
INUETOBEGUIDEDBYTHEIDEATHATWECANACHIEVESOMETHINGMUCHGREATERTOGETHERTHANWECAN
INDIVIDUALLYAFTERALLTHATWASTHEIDEATHATLEDTOTHECREATIONOFOURUNIVERSITYINTHEFI
RSTPLACE
```

2. Yes. Given a 8-stage LFSR, we know:

$$\begin{cases} a_n = (a_{n+1}C_7 + a_{n+2}C_6 + a_{n+3}C_5 + \ldots + a_{n+6}C_2 + a_{n+7}C_1 + a_{n+8}C_0) \mod 2 \\ a_{n+1} = (a_{n+2}C_7 + a_{n+3}C_6 + a_{n+4}C_5 + \ldots + a_{n+7}C_2 + a_{n+8}C_1 + a_{n+9}C_0) \mod 2 \\ a_{n+2} = (a_{n+3}C_7 + a_{n+4}C_6 + a_{n+5}C_5 + \ldots + a_{n+8}C_2 + a_{n+9}C_1 + a_{n+10}C_0) \mod 2 \\ a_{n+3} = (a_{n+4}C_7 + a_{n+5}C_6 + a_{n+6}C_5 + \ldots + a_{n+9}C_2 + a_{n+10}C_1 + a_{n+11}C_0) \mod 2 \\ a_{n+4} = (a_{n+5}C_7 + a_{n+6}C_6 + a_{n+7}C_5 + \ldots + a_{n+10}C_2 + a_{n+11}C_1 + a_{n+12}C_0) \mod 2 \\ a_{n+5} = (a_{n+6}C_7 + a_{n+7}C_6 + a_{n+8}C_5 + \ldots + a_{n+11}C_2 + a_{n+12}C_1 + a_{n+13}C_0) \mod 2 \\ a_{n+6} = (a_{n+7}C_7 + a_{n+8}C_6 + a_{n+9}C_5 + \ldots + a_{n+12}C_2 + a_{n+13}C_1 + a_{n+14}C_0) \mod 2 \\ a_{n+7} = (a_{n+8}C_7 + a_{n+9}C_6 + a_{n+10}C_5 + \ldots + a_{n+13}C_2 + a_{n+14}C_1 + a_{n+15}C_0) \mod 2 \end{cases}$$

Knowing $a_0,\ a_1,\ \ldots,\ a_{15}$, can compute $C_0,\ C_1,\ \ldots,\ C_7$, thus can solve a 8-stage LFSR.

3. $C_0 = 1,\ C_1 = 0,\ C_2 = 0,\ C_3 = 0,\ C_4 = 1,\ C_5 = 1,\ C_6 = 1,\ C_7 = 0$

```python
f16keyoutput = [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0]
solution = [0, 0, 0, 0, 0, 0, 0, 0]

for i in range(255):
    flg = 1
    for j in range(8):
        c = 0
        for k in range(8):
            c += solution[k] * f16keyoutput[j+1+k]
        if(f16keyoutput[j] != c%2):
            flg = 0
            break
    if flg:
        print(solution)
        break

    cnt = 0
    while(solution[cnt]):
        solution[cnt] = 0
        cnt += 1
    solution[cnt] = 1
```

Output:

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

3 Problem 3

1. output:

```
Naive algorithm:
(1, 2, 3, 4): 38688 (2, 3, 1, 4): 54314 (3, 4, 1, 2): 42889
(1, 2, 4, 3): 39161 (2, 3, 4, 1): 54895 (3, 4, 2, 1): 38954
(1, 3, 2, 4): 39184 (2, 4, 1, 3): 43461 (4, 1, 2, 3): 31338
(1, 3, 4, 2): 54739 (2, 4, 3, 1): 42571 (4, 1, 3, 2): 35229
(1, 4, 2, 3): 43078 (3, 1, 2, 4): 43135 (4, 2, 1, 3): 35416
(1, 4, 3, 2): 35168 (3, 1, 4, 2): 42929 (4, 2, 3, 1): 31271
(2, 1, 3, 4): 39189 (3, 2, 1, 4): 35047 (4, 3, 1, 2): 39013
(2, 1, 4, 3): 58516 (3, 2, 4, 1): 42893 (4, 3, 2, 1): 38922
Average: 41666.666666666664, Standard Deviation: 7168.251295740747

Fisher-Yates shuffle:
(1, 2, 3, 4): 41951 (2, 3, 1, 4): 41694 (3, 4, 1, 2): 41600
(1, 2, 4, 3): 41832 (2, 3, 4, 1): 41521 (3, 4, 2, 1): 41528
(1, 3, 2, 4): 41968 (2, 4, 1, 3): 41997 (4, 1, 2, 3): 41651
(1, 3, 4, 2): 42073 (2, 4, 3, 1): 41199 (4, 1, 3, 2): 41263
(1, 4, 2, 3): 41594 (3, 1, 2, 4): 41737 (4, 2, 1, 3): 41348
(1, 4, 3, 2): 41466 (3, 1, 4, 2): 41953 (4, 2, 3, 1): 41777
(2, 1, 3, 4): 41668 (3, 2, 1, 4): 41553 (4, 3, 1, 2): 41643
(2, 1, 4, 3): 41730 (3, 2, 4, 1): 41299 (4, 3, 2, 1): 41955
Average: 41666.666666666664, Standard Deviation: 240.0315372797685
```

2. Fisher-Yates shuffle is better because it demonstrates a more uniform distribution, as indicated by a lower standard deviation compared to the Naïve algorithm.
3. The Naïve algorithm is known to be biased because the probability of each element in each position is not equal, primarily due to the fact that it always selects from the entire range of indices for swapping. In contrast, the Fisher-Yates shuffle ensures each position can only swap with positions before it (or itself), leading to a more uniform and unbiased distribution.

4 Appendix

Package used in the program: numpy

```
pip install numpy
```