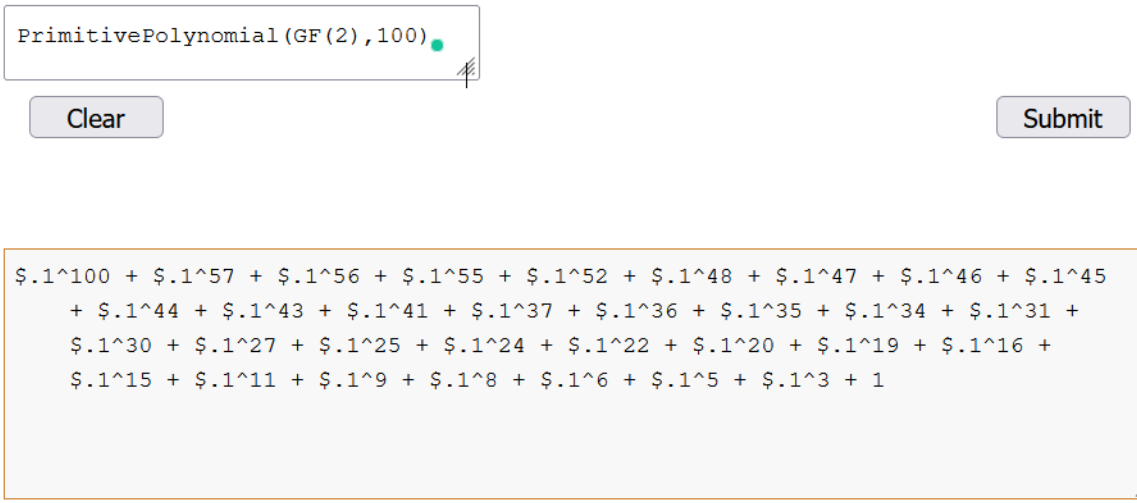
# First mandatory assignment

## Problem 1.

The minimal possible cycle length for an LFSR on 100 bits is (2^100) – 1. This is because the 0 state maps to itself. Now to construct a LFSR I have used magma to construct a primitive polynomial for me as you can see from the screenshot.




## Problem 2.

To execute the implementation, unzip the “Trivium-stream-cipher.zip” and run the “~/src/main/Main” file using java JDK 17

You should see the following in the terminal:

*This should be the 1000 bit stream cipher using the key and IV provided in the mandatory note:*

*[1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, ..., 1, 1, 0, 0, 1, 1]*

I’m not including the full 1000 bits in this document due to it flooding the whole page with 1’s and 0’s.

## Problem 3.

To execute the implementation, unzip the “feistel-network.zip” and run the “~/src/main/Main” file using java JDK 17

You should see the following in the terminal:

*Encrypting mandatory conditions*

*Key: [0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0]*

*Plaintext: [0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0]*

*Encrypted message: [0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1]*

*Decrypting mandatory conditions*

*Key: [0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0]*

*Cipher: [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]*

*Decrypted message: [1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0]*

## **Problem 4.**

To access both subtasks, unzip “feistel-exhaustive-search” and I have run all of my python files using python 2.5.3

### **Subtask 1:**

When executing “keySearcher.py” the output to the terminal should be:

*[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] … (searching keys)*

*… The key is: [1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1]*

### **Subtask 2:**

When executing “4.2.py” the output to the terminal should be:

*… (Keys)* **...**

*Found the key*

*0100011110010111*

*0011110000101111*