REPORT [Cyclic Code Encoder**]**

Pitaka Laxmi Venkata Naga Satya (2203124)

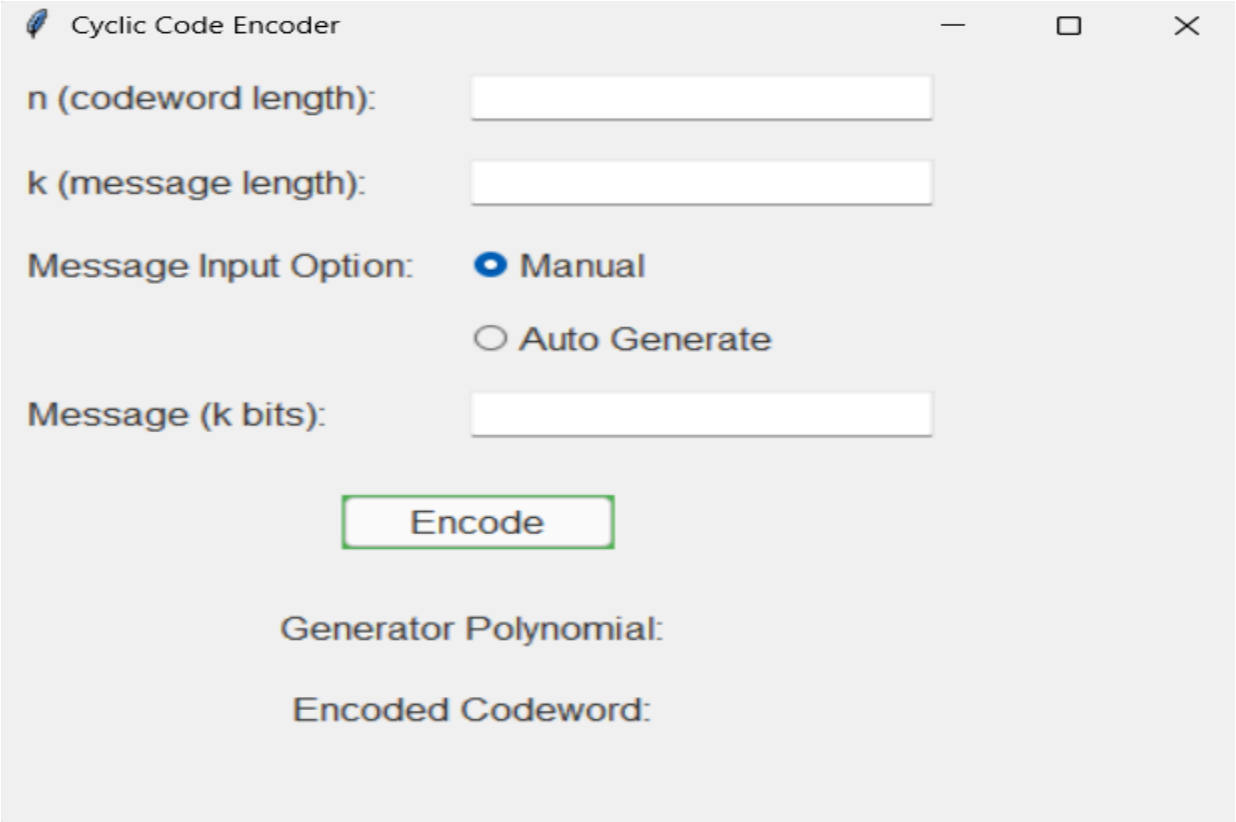
Banoth Anusha (2203104)

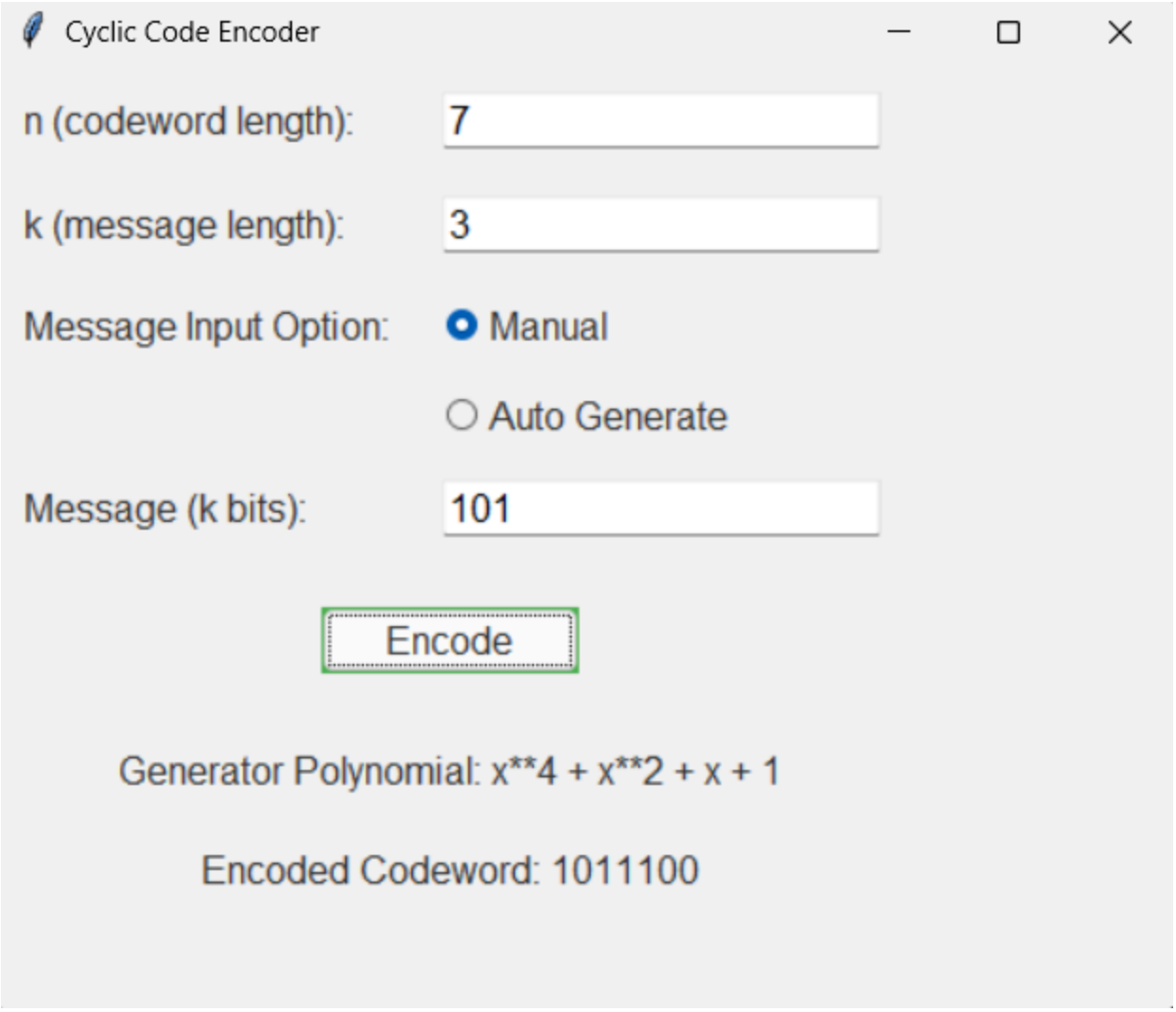
1. Introduction This project implements a Cyclic Code Encoder using Python. The application is designed to take user input parameters for a cyclic code, such as codeword length (n) and message length (k), generate a valid generator polynomial, and encode a given message accordingly. The program also provides a Graphical User Interface (GUI) for ease of use.

2. Design Overview The cyclic encoding process involves the following steps:

- Finding the generator polynomial (g(x)) that satisfies cyclic encoding properties.
- Encoding a given k-bit message using the generator polynomial.
- Providing an option for users to enter a message manually or generate it randomly.
- Displaying the resulting generator polynomial and the encoded codeword.

The implementation is structured into modular functions, ensuring reusability and clarity.





3. Tools and Technologies Used

- **Python**: The primary programming language used for implementing the encoder.
- **NumPy**: Utilized for numerical operations and handling polynomial calculations.
- **SymPy**: Used for symbolic mathematics, particularly for polynomial operations in finite fields.
- **Tkinter**: The built-in Python library used for creating the GUI, enabling user interaction.
- **Random Module**: Used to generate random binary messages for automatic message encoding.

4. Implementation Details

- **Finding the Generator Polynomial**
    - The function `find_generator_polynomial(n, k)` searches for a polynomial that divides x^n - 1 in the finite field GF(2).

    - This ensures proper cyclic encoding and allows the generation of valid codewords.
- **Encoding Process**
    - The message polynomial is multiplied by x^(n-k) to shift the message bits.
    - A division is performed with the generator polynomial to compute parity bits.
    - The remainder is added to the shifted message to generate the final encoded codeword.
- **Graphical User Interface**
    - Tkinter provides a simple interface with input fields for n, k, and message entry.
    - Users can select between manual input and automatic message generation.
    - The encoded codeword and generator polynomial are displayed as output.

5. Challenges and Solutions

- **Polynomial Selection**: The main challenge was ensuring that the chosen generator polynomial is valid. The implemented method systematically checks polynomials in GF(2) to find a suitable candidate.
- **GUI Integration**: Tkinter was effectively used to simplify user interaction and validate inputs to prevent errors.

6. Conclusion The cyclic code encoder successfully implements encoding of binary messages using cyclic redundancy principles. The application provides a user-friendly interface and performs encoding using algebraic techniques with Python libraries.
