Example 7 (Solutions) — Shift Ciphers as a Cryptosystem

Goal. Describe the family of shift ciphers as a formal cryptosystem and verify that encryption and decryption are inverses.

Full Walkthrough and Explanation

We want to represent the shift cipher in the five-part framework

$$(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D}).$$

Step 1 — Mapping letters to numbers. Each letter is represented as an integer between 0 and 25:

$$A = 0,\ B = 1,\ \ldots,\ Z = 25.$$

This lets us use modular arithmetic instead of alphabet juggling.

Step 2 — Defining encryption and decryption.

$$E_k(p) = (p + k) \bmod 26, \qquad D_k(c) = (c - k) \bmod 26.$$

Here k is the key (the amount of shift).

Step 3 — Building the formal 5-tuple.

$$\begin{aligned}
&\mathcal{P} = \mathcal{C} = \text{all strings of elements in } \mathbb{Z}_{26},\\
&\mathcal{K} = \mathbb{Z}_{26},\\
&\mathcal{E} = \{\, E_k(p) = (p + k) \bmod 26 \mid k \in \mathbb{Z}_{26} \,\},\\
&\mathcal{D} = \{\, D_k(c) = (c - k) \bmod 26 \mid k \in \mathbb{Z}_{26} \,\}.
\end{aligned}$$

Step 4 — Verifying the "undo" property.

$$D_k(E_k(p)) = ((p + k) - k) \bmod 26 = p.$$

So decryption perfectly reverses encryption.

Key Insight. Shift ciphers show how a single idea—addition mod 26—can define a whole family of related ciphers, one for each k in $\mathbb{Z}_{26}$.

Common pitfalls

- Students sometimes treat "keyspace" $\mathcal{K}$ as just one value instead of the full set of possible k.
- Forgetting the modulus (especially when $p + k > 25$) leads to wrong letters.
- Because there are only 26 possible k, a brute-force attack breaks the cipher immediately—this motivates more sophisticated systems.

Practice Problem Solutions

Problem A (Easier). Given $k = 5$:

$$E_k(p) = (p + 5) \bmod 26, \qquad D_k(c) = (c - 5) \bmod 26.$$

"Mod 26" guarantees we stay inside the alphabet—after Z (25), we wrap around to A (0).

—

Problem B (Similar). Let $p = 19$ (the letter T) and $k = 7$.

$$E_k(p) = (19 + 7) \bmod 26 = 0 \Rightarrow \text{A}.$$

Encrypting T gives A. Decrypting:

$$D_k(0) = (0 - 7) \bmod 26 = 19 \Rightarrow \text{T}.$$

We return to the original plaintext, confirming correctness.

—

Problem C (Harder). If we expand the system to include digits 0–9, we now have 36 symbols. So the modulus becomes 36 and each component adjusts:

$$\begin{aligned}
&\mathcal{P} = \mathcal{C} = \text{all strings of elements in } \mathbb{Z}_{36}, \\
&\mathcal{K} = \mathbb{Z}_{36}, \\
&\mathcal{E} = \{E_k(p) = (p + k) \bmod 36\}, \\
&\mathcal{D} = \{D_k(c) = (c - k) \bmod 36\}.
\end{aligned}$$

The idea is identical—just a larger alphabet!

—

Reflection Answer. Writing cryptography formally gives us a reusable structure: we can swap in new alphabets, key spaces, or modular groups and instantly define new families of ciphers. It's mathematics as blueprint—one small idea, infinitely extendable.