Problem 1. *(Decipher Blum-Goldwasser)* Bob receives the following cryptogram from Alice:

\[ c = (10101011100001101000101110010111100110111000, x_{t+1} = 1306) \]

The message \( m \) has been encrypted using the Blum-Goldwasser cryptosystem with public key \( n = 1333 = 31 \cdot 43 \). The letters of the Latin alphabet \( A, \ldots, Z \) are represented by the following 5 bit scheme: \( A = 00000, \ B = 00001, \ldots, Z = 11001 \). Decipher the cryptogram \( c \).

*Remark:* The security requirement to use at most \( h = \lfloor \log_2 \lfloor \log_2 (n) \rfloor \rfloor \) bits of the Blum-Blum-Shub generator is violated in this example. Instead, 5 bits of the output are used.

Problem 2. *(Blum-Blum-Shub generator)* The security of the Blum-Blum-Shub generator is based on the difficulty to compute square roots modulo \( n = pq \) for two distinct primes \( p \) and \( q \) with \( p, q \equiv 3 \mod 4 \).

Design a generator for pseudo-random bits which is based on the hardness of the RSA-problem.

Problem 3. *(Basic requirements for cryptographic hash functions)* Using a block cipher \( E_K(x) \) with block length \( k \) and key \( K \), a hash function \( h(m) \) is provided in the following way.

Append \( m \) with zero bits until it is a multiple of \( k \), divide \( m \) into \( n \) blocks of \( k \) bits each.

\[
\begin{align*}
c \leftarrow E_{m_0}(m_0) \\
\text{for } i \text{ in } 1 \ldots (n - 1) \text{ do} \\
\quad c \leftarrow c \oplus E_{m_0}(m_i) \\
\text{end for} \\
h(m) \leftarrow c
\end{align*}
\]

The operator \( \oplus \) denotes bitwise adding modulo 2, or in other words XOR.

a) Does this function fulfill the basic requirements for a cryptographic hash function?

b) Does this function fulfill the basic requirements for a cryptographic hash function, if the operator XOR (\( \oplus \)) is replaced by AND (\( \odot \)), i.e., bitwise multiplication modulo 2?

c) Why is the replacement of XOR by AND a bad idea?