Problem 1

(a) For example, the Fermat test means computing $a^{p-1} \bmod p$ for some $2 \leq a \leq p-1$, and check if it equals 1. If p is a prime, it must always be so. For example, computing $2^{1040} \bmod 1041 = 4$, $2^{1042} \bmod 1043 \neq 1$, $2^{1047} \bmod 1047 \neq 1$, $2^{1048} \bmod 1049 = 1$, and conclude that 1049 is a prime.

(b)

$$\phi(n) = (p-1)(q-1) = 126 \cdot 1048 = 132048.$$

Picking smallest $e > 20$ such that $\gcd(132048, e) = 1$. Testing gives $\gcd(132048, 21) \neq 1$ and $\gcd(132048, 23) = 1$, so $e = 23$. The secret d is the inverse of e modulo $\phi(n)$, i.e., $d \cdot e = 1 \bmod \phi(n)$. We compute it through Euclidean algorithm to be $d = 51671$.

(c) Computing $C = 2345^{23}$. Square and multiply means first squaring by computing

$$M = 2345, M^2 = 2345^2 = 36882, M^4 = (M^2)^2 = 36882^2 = 75094,$$

$$M^8 = (M^4)^2 = 75094^2 = 45692, M^{16} = (M^8)^2 = 21231,$$

all calculations in $\mathbb{Z}_n$. Then observe that

$$C = 2345^{23} = 2345^{16+4+2+1} = 2345^{16} \cdot 2345^4 \cdot 2345^2 \cdot 2345 = 21231 \cdot 75094 \cdot 36882 \cdot 2345 = 58952.$$

(d) Decryption is given by $M' = C^d \bmod n$ and a correct ciphertext is created as $C = M^e \bmod n$, where M is the message. We also have $e \cdot d = 1 \bmod \phi(n)$, so we can write $e \cdot d = 1 + k \cdot \phi(n)$ for some integer k. Then

$$M' = C^d = (M^e)^d = M^{ed} = M^{1+k \cdot \phi(n)} = M \cdot M^{k\phi(n)} = M \bmod n,$$

because of Euler's theorem. This is true when $\gcd(M, n) = 1$. For the case $\gcd(M, n) \neq 1$, the statement is still true but has to be proved by using Chinese remainder theorem or similar (not required).

Problem 2

(a) WRONG ($P_1(x) = (x^2 + x + 1)(x^3 + x^2 + 1)$)
(b) WRONG (if it is not irreducible it cannot be primitive)
(c) CORRECT ($T_1 = 15$ and $T_2 = 2^i T_1$ computing $i = 1$ from the formula $2^{i-1} < j = 2 \leq 2^i$)
(d) WRONG ($(\alpha^{10})^3 = 1$ so ord$(\alpha^{10}) = 3$)
(e) CORRECT ($x = 1$)

Problem 3

(a) Given formulas:

$$D = H_0 - H(M),$$

$$N_0 = H(K)/D.$$

Computing

$$H_0 = \log L = \log 2^{128} = 128.$$

$$H(M) = 2^{127} \cdot (-2^{-127} \log(2^{-127})) + 2^{127} \cdot (0 \log(0)) = 127.$$

$$H(K) = \log(|\mathcal{K}|) = \log(2^{128}) = 128.$$

Finally,

$$N_0 = H(K)/D = \frac{128}{(128 - 127)} \approx 128 \text{ symbols.}$$

(b) The conditions are ciphertext-only attacks (Eve has access only to the ciphertext) and unconditional security (Eve has infinite computing power). This means that after observing slightly more than $N_0 = 128$ blocks, she can determine the secret key, for example by using an exhaustive key search. In practice, we do not have infinite computing power and we talk about a cipher being broken only if there is an attack method that is more efficient than exhaustive key search.

(c) For example homophonic coding. In this particular example it would be done by adding a uniform random bit in the first position of the plaintext block. Then all 128 bit vectors are equally likely and $D = 0$. In decryption, we just set the first position of the plaintext block to be 0 again.

Problem 4

(a)
- $\Gamma_0 = \{P_1P_2; P_1P_3; P_2P_3; P_4P_5\}$ (and any subset in $\text{cl}(\Gamma_0)$ can access the secret).
- Ideal SSS means that it is perfect and the size of shares are minimum (same as the size of the secret). Since the construction can be viewed as two independent threshold schemes, one (2,3)-threshold scheme for P_1, P_2, P_3 and one (2,2)-threshold scheme for P_4, P_5, and it is known that Shamir threshold scheme is ideal, it follows.
- Finally, a simple reconstruction. Since $k = 2$, we have the secret polynomial as $a(x) = a_0 + a_1x$ and from the shares we have $y_2 = a(2) = a_0 + 2a_1 = 3$ and $y_3 = a(3) = a_0 + 3a_1 = 10$, leading to $K = a_0 = 8$ after solving the system of 2 equations in two unknowns in $\mathbb{F}_{19}$.

(b) see the lecture notes. When q is composite P_S can be larger than $1/q$, since a linear equation can then have more than one solution $\bmod q$.

Problem 5

This is essentially the problem you solved in project 3. Recall that the correlation attack uses a divide-and-conquer approach. We first establish a correlaton between output and LFSR sequences, by observing (or recalling from the project) that $P(z_i = u_j) = 3/4$, $j = 1, 2, 3$, in the cipher. So the approach is to write out the possible sequences from a single LFSR, compare with $\mathbf{z}$ and find the sequence closest in Hamming sense.

Start with LFSR3 (shortest): The three nonzero starting states and their corresponding LFSR sequences are

$(0,1)$	0110 1101 1011 0110 1101 1011 0110 1101 1011
$(1,0)$	1011 0110 1101 1011 0110 1101 1011 0110 1101
$(1,1)$	1101 1011 0110 1101 1011 0110 1101 1011 0110
$\mathbf{z}$	0011 1110 1011 0110 1111 1001 0110 0101 1011

The distances (number of positions in which they differ from $\mathbf{z}$) are 7, 17 and 25. Clearly the sequence from $(0,1)$ is closest to $\mathbf{z}$.

Do the same with LFSR2, checking 7 sequences, to get starting state $(0,0,1)$.

Finally, knowing two of three sequences you can extract the starting state of LFSR1.

LFSR3 $(0, 1)$	0110 1101 1011 0110 1101
LFSR2 $(0, 0, 1)$	0011 1010 0111 0100 1110
z	0011 1110 1011 0110 1111
LFSR1 ?	x0x1 x110 00xx ...

After some calculation, it gives starting state $(0, 0, 0, 1)$.

You can of cause also do as for LFSR2 and LFSR3, but then you need to check 15 different sequences.