Problems

1.1. The ciphertext below was encrypted using a substitution cipher. Decrypt the ciphertext without knowledge of the key.

```
  lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx qmbm wi
bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jx
ymbinlmtmipw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr
yjeryrkbi jx bpr qmbm mvvjudwko bj yt wkbrusurbmbwjk
lmird jk xjubt trmui jx ibndt

  wb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi
iwokwxwvmkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m
vjyshrbr rashmkmbwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd
wkiswurd bj invp mk rabrkb bpmb pr vjnhd urmvp bpr ibmbr
jx rkhwopbrkrd ywkd vmsmlhr jx urvjokwgwko ijnkdhrii
ijnkd mkd ipmsrhrii ipmsr w dj kjb drry ytirhx bpr xwkmh
mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj djnlb
bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd
wkbrusurbmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr
riirkvr jx jqwkmcmk qmumbr cwhh urymwk wkbmvb
```

1. Compute the relative frequency of all letters `A...Z` in the ciphertext. You may want to use a tool such as the open-source program CrypTool [50] for this task. However, a paper and pencil approach is also still doable.
2. Decrypt the ciphertext with the help of the relative letter frequency of the English language (see Table 1.1 in Sect. 1.2.2). Note that the text is relatively short and that the letter frequencies in it might not perfectly align with that of general English language from the table.
3. Who wrote the text?

1.2. We received the following ciphertext which was encoded with a shift cipher:
`xultpaajcxitltlxaarpjhtiwtgxktghidhipxciwtvgtpilpit ghlxiwiwtxgqadds`.

1. Perform an attack against the cipher based on a letter frequency count: How many letters do you have to identify through a frequency count to recover the key? What is the cleartext?
2. Who wrote this message?

1.3. We consider the long-term security of the Advanced Encryption Standard (AES) with a key length of 128-bit with respect to exhaustive key-search attacks. AES is perhaps the most widely used symmetric cipher at this time.

1. Assume that an attacker has a special purpose application specific integrated circuit (ASIC) which checks $5 \cdot 10^8$ keys per second, and she has a budget of $1 million. One ASIC costs $50, and we assume 100% overhead for integrating

the ASIC (manufacturing the printed circuit boards, power supply, cooling, etc.). How many ASICs can we run in parallel with the given budget? How long does an average key search take? Relate this time to the age of the Universe, which is about 10^{10} years.
2. We try now to take advances in computer technology into account. Predicting the future tends to be tricky but the estimate usually applied is Moore's Law, which states that the computer power doubles every 18 months while the costs of integrated circuits stay constant. How many years do we have to wait until a key-search machine can be built for breaking AES with 128 bit with an average search time of 24 hours? Again, assume a budget of $1 million (do not take inflation into account).

1.4. We now consider the relation between passwords and key size. For this purpose we consider a cryptosystem where the user enters a key in the form of a password.

1. Assume a password consisting of 8 letters, where each letter is encoded by the ASCII scheme (7 bits per character, i.e., 128 possible characters). What is the size of the key space which can be constructed by such passwords?
2. What is the corresponding key length in bits?
3. Assume that most users use only the 26 lowercase letters from the alphabet instead of the full 7 bits of the ASCII-encoding. What is the corresponding key length in bits in this case?
4. At least how many characters are required for a password in order to generate a key length of 128 bits in case of letters consisting of

   a. 7-bit characters?
   b. 26 lowercase letters from the alphabet?

1.5. As we learned in this chapter, modular arithmetic is the basis of many cryptosystems. As a consequence, we will address this topic with several problems in this and upcoming chapters.

Let's start with an easy one: Compute the result without a calculator.

1. $15 \cdot 29 \bmod 13$
2. $2 \cdot 29 \bmod 13$
3. $2 \cdot 3 \bmod 13$
4. $-11 \cdot 3 \bmod 13$

The results should be given in the range from $0, 1, \ldots,$ modulus-1. Briefly describe the relation between the different parts of the problem.

1.6. Compute without a calculator:

1. $1/5 \bmod 13$
2. $1/5 \bmod 7$
3. $3 \cdot 2/5 \bmod 7$

1.7. We consider the ring $\mathbb{Z}_4$. Construct a table which describes the addition of all elements in the ring with each other:

+	0	1	2	3
0	0	1	2	3
1	1	2 ⋯		
2	⋯			
3				

1. Construct the multiplication table for $\mathbb{Z}_4$.
2. Construct the addition and multiplication tables for $\mathbb{Z}_5$.
3. Construct the addition and multiplication tables for $\mathbb{Z}_6$.
4. There are elements in $\mathbb{Z}_4$ and $\mathbb{Z}_6$ without a multiplicative inverse. Which elements are these? Why does a multiplicative inverse exist for all nonzero elements in $\mathbb{Z}_5$?

1.8. What is the multiplicative inverse of 5 in $\mathbb{Z}_{11}$, $\mathbb{Z}_{12}$, and $\mathbb{Z}_{13}$? You can do a trial-and-error search using a calculator or a PC.

With this simple problem we want now to stress the fact that the inverse of an integer in a given ring depends completely on the ring considered. That is, if the modulus changes, the inverse changes. Hence, it doesn't make sense to talk about an inverse of an element unless it is clear what the modulus is. This fact is crucial for the RSA cryptosystem, which is introduced in Chap. 7. The extended Euclidean algorithm, which can be used for computing inverses efficiently, is introduced in Sect. 6.3.

1.9. Compute x as far as possible without a calculator. Where appropriate, make use of a smart decomposition of the exponent as shown in the example in Sect. 1.4.1:

1. $x = 3^2 \bmod 13$
2. $x = 7^2 \bmod 13$
3. $x = 3^{10} \bmod 13$
4. $x = 7^{100} \bmod 13$
5. $7^x = 11 \bmod 13$

The last problem is called a *discrete logarithm* and points to a hard problem which we discuss in Chap. 8. The security of many public-key schemes is based on the hardness of solving the discrete logarithm for large numbers, e.g., with more than 1000 bits.

1.10. Find all integers n between $0 \leq n < m$ that are relatively prime to m for $m = 4, 5, 9, 26$. We denote the *number* of integers n which fulfill the condition by $\phi(m)$, e.g. $\phi(3) = 2$. This function is called "Euler's phi function". What is $\phi(m)$ for $m = 4, 5, 9, 26$?

1.11. This problem deals with the affine cipher with the key parameters $a = 7$, $b = 22$.

1. Decrypt the text below:
   `falszztysyjzyjkywjrztyjztyynaryjkyswarztyegyyj`
2. Who wrote the line?

1.12. Now, we want to extend the affine cipher from Sect. 1.4.4 such that we can encrypt and decrypt messages written with the full German alphabet. The German alphabet consists of the English one together with the three umlauts, Ä, Ö, Ü, and the (even stranger) "double s" character ß. We use the following mapping from letters to integers:

A $\leftrightarrow$ 0	B $\leftrightarrow$ 1	C $\leftrightarrow$ 2	D $\leftrightarrow$ 3	E $\leftrightarrow$ 4	F $\leftrightarrow$ 5
G $\leftrightarrow$ 6	H $\leftrightarrow$ 7	I $\leftrightarrow$ 8	J $\leftrightarrow$ 9	K $\leftrightarrow$ 10	L $\leftrightarrow$ 11
M $\leftrightarrow$ 12	N $\leftrightarrow$ 13	O $\leftrightarrow$ 14	P $\leftrightarrow$ 15	Q $\leftrightarrow$ 16	R $\leftrightarrow$ 17
S $\leftrightarrow$ 18	T $\leftrightarrow$ 19	U $\leftrightarrow$ 20	V $\leftrightarrow$ 21	W $\leftrightarrow$ 22	X $\leftrightarrow$ 23
Y $\leftrightarrow$ 24	Z $\leftrightarrow$ 25	Ä $\leftrightarrow$ 26	Ö $\leftrightarrow$ 27	Ü $\leftrightarrow$ 28	ß $\leftrightarrow$ 29

1. What are the encryption and decryption equations for the cipher?
2. How large is the key space of the affine cipher for this alphabet?
3. The following ciphertext was encrypted using the key $(a = 17, b = 1)$. What is the corresponding plaintext?

   ```
   ä u ß w ß
   ```

4. From which village does the plaintext come?

1.13. In an attack scenario, we assume that the attacker Oscar manages somehow to provide Alice with a few pieces of plaintext that she encrypts. Show how Oscar can break the affine cipher by using two pairs of plaintext–ciphertext, (x_1, y_1) and (x_2, y_2). What is the condition for choosing x_1 and x_2?

Remark: In practice, such an assumption turns out to be valid for certain settings, e.g., encryption by Web servers, etc. This attack scenario is, thus, very important and is denoted as a *chosen plaintext attack*.

1.14. An obvious approach to increase the security of a symmetric algorithm is to apply the same cipher twice, i.e.:

$$y = e_{k2}(e_{k1}(x))$$

As is often the case in cryptography, things are very tricky and results are often different from the expected and/ or desired ones. In this problem we show that a double encryption with the affine cipher is only as secure as single encryption! Assume two affine ciphers $e_{k1} = a_1 x + b_1$ and $e_{k2} = a_2 x + b_2$.

1. Show that there is a single affine cipher $e_{k3} = a_3 x + b_3$ which performs exactly the same encryption (and decryption) as the combination $e_{k2}(e_{k1}(x))$.
2. Find the values for a_3, b_3 when $a_1 = 3, b_1 = 5$ and $a_2 = 11, b_2 = 7$.
3. For verification: (1) encrypt the letter `K` first with e_{k1} and the result with e_{k2}, and (2) encrypt the letter `K` with e_{k3}.
4. Briefly describe what happens if an exhaustive key-search attack is applied to a double-encrypted affine ciphertext. Is the effective key space increased?

Remark: The issue of multiple encryption is of great practical importance in the case of the Data Encryption Standard (DES), for which multiple encryption (in particular, triple encryption) does increase security considerably.