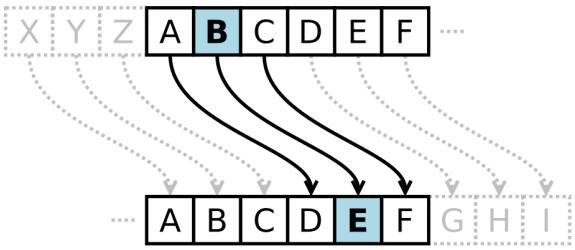
Caesar – Verschlüsselung

Steffen Weißer

2018

1 Verschlüsselung

Möchte man geheime Nachrichten verschicken, so muss man sich etwas überlegen, damit sie nicht jeder lesen kann. Beispielsweise kann man sich eine Geheimschrift ausdenken, in der man jeden Buchstaben durch ein Symbol ersetzt. Bereits Caesar im alten Rom hatte sich darüber Gedanken gemacht. Wie könnte er seinen Freunden eine Nachricht schicken ohne dass der Bote sie lesen kann? Seine Idee war, einfach jeden Buchstaben in der Nachricht durch den Buchstaben, der drei Stellen weiter hinten im Alphabet folgt, zu ersetzen.



So wird aus dem `B` ein `E` und aus dem `e` ein `h`. Auf diese Weise erhält man zum Beispiel aus

`Hallo` den Geheimtext `Kdoor`.

Kommt man hinten am Alphabet an, so macht man einfach vorne wieder weiter. Aus `Y` wird somit `B`. Diese Art einen Geheimtext zu erstellen nennt man *Caesar-Verschlüsselung*.

Bei dieser Verschlüsselung muss man nicht unbedingt den dritten nachfolgenden Buchstaben verwenden. Mann könnte auch den fünften oder den sechsten benutzen. Diese Zahl nennt man Schlüssel. Wählt man als Schlüssel 1, so wird aus dem `A` ein `B`. Wählt man als Schlüssel 26 so wird aus dem `A` wieder ein `A`, da das Alphabet 26 Buchstaben hat. Aus diesem Grund kann man einen Text auch ganz einfach wieder entschlüsseln, sofern man den Schlüssel kennt. Ist der Geheimtext mit dem Schlüssel 3 erstellt worden, so erhält man die entschlüsselte Nachricht zurück, wenn man den Geheimtext nochmals verschlüsselt und dafür den Schlüssel 23 wählt. Denn 3+23=26.

Wie lautet dein Name als verschlüsselter Text mit Schlüssel 3?

2 Python

Jetzt müssen wir uns überlegen, wie man eine solche Caesar-Verschlüsselung in Python programmieren kann. Hierzu benötigen wir einige verschiedene Schritte. Zunächst stellt sich die Frage, wie man von einem Buchstaben im Alphabet den Buchstaben herausfindet, der drei Stellen weiter hinten im Alphabet steht.

Im Computer ist jedem Buchstaben eine Zahl zugeordnet. Genauer gesagt, ist sogar jedem Zeichen eine Zahl zugeordnet. Diese Zahl erhält man mit der Funktion `ord()`. Möchtest du zum Beispiel die Zahl zu dem Buchstaben `A` wissen, so kannst du folgendes in Python eingeben

```
>>> ord('A')
65
```

und erfährst, dass der Buchstabe `A` der Zahl 65 zugeordnet ist. Möchtest du Umgekehrt wissen, welcher Buchstabe einer Zahl zugeordnet ist, so verwendest du die Funktion `chr()`. Also zum Beispiel:

```
>>> chr(65)
'A'
```

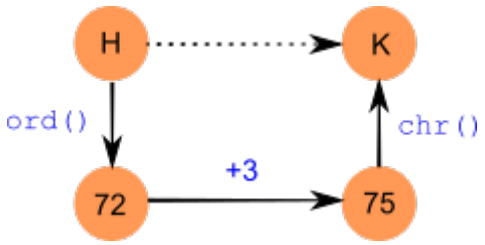
Die Zuordnung der Buchstaben zu den Zahlen ist in der ASCII-Tabelle festgelegt. Du kannst dir diese Tabelle ansehen, indem du mit einer `for`-Schleife die Zahlen `i` und die dazu gehörenden Zeichen mit `chr(i)` ausgeben lässt. Das funktioniert zum Beispiel mit:

```
>>> for i in range(32, 64):
        print(i,chr(i),"   ",i+32,chr(i+32),"   ",i+64,chr(i+64))

32          64 @        96 `
33 !        65 A        97 a
34 "        66 B        98 b
35 #        67 C        99 c
36 $        68 D        100 d
37 %        69 E        101 e
38 &        70 F        102 f
39 '        71 G        103 g
40 (        72 H        104 h
41 )        73 I        105 i
42 *        74 J        106 j
43 +        75 K        107 k
44 ,        76 L        108 l
45 -        77 M        109 m
46 .        78 N        110 n
47 /        79 O        111 o
48 0        80 P        112 p
49 1        81 Q        113 q
50 2        82 R        114 r
51 3        83 S        115 s
52 4        84 T        116 t
```

```
53 5       85 U       117 u
54 6       86 V       118 v
55 7       87 W       119 w
56 8       88 X       120 x
57 9       89 Y       121 y
58 :       90 Z       122 z
59 ;       91 [       123 {
60 <       92 \       124 |
61 =       93 ]       125 }
62 >       94 ^       126 ~
63 ?       95 _       127 -
```

Wollen wir nun das Wort `'Hallo'` verschlüsseln, so sehen wir in der Tabelle, dass aus dem `'H'` ein `'K'` wird. Außerdem wird aus dem `'a'` ein `'d'` und so weiter. In Python können wir das nun mit Hilfe der Funktionen `ord()` und `chr()` erreichen. Hierzu gehen wir wie in der Grafik angedeutet vor:



In Python lautet dies dann:

```
>>> ord('H')
72
>>> ord('H')+3
75
>>> chr(ord('H')+3)
'K'
```

Was geschieht nun allerdings, wenn wir das Selbe mit dem Buchstaben `'Z'` machen?

```
>>> chr(ord('Z')+3)
']'
```

Wir sind also über das Alphabet hinaus gegangen. In diesem Fall müssen wir wieder vorn bei `'A'` beginnen. Das erreichen wir mit einer `if`-Abfrage und der folgenden Rechnung:

```
>>> buchstabe = 'Z'
>>> if ord(buchstabe)+3 > ord('Z'):
        chr( ord(buchstabe)+3 - ord('Z') + ord('A') - 1 )
else:
        chr( ord(buchstabe)+3 )

'C'
>>>
```

Diese Umrechnung ist für Großbuchstaben. Natürlich benötigen wir später eine entsprechende Abfrage für Kleinbuchstaben.

Wir haben nun gelernt, wie man die Buchstaben für die Caesar-Verschlüsselung verschiebt. Als nächstes sollten wir uns überlegen, wie wir vorgehen, wenn wir nicht nur einen Buchstaben sondern ein ganzes Wort oder sogar einen Text haben. Wir wollen nun nacheinander alle Buchstaben in `text='Hallo'` einzeln bearbeiten. Um dies einmal auszuprobieren, geben wir jeden Buchstaben nacheinander aus. Hierzu können wir eine `for`-Schleife verwenden um jeden Buchstaben im Text zu bearbeiten.

```
>>> text = 'Hallo'
>>> for buchstabe in text:
        print(buchstabe)

H
a
l
l
o
>>>
```

Wir können auch die Buchstaben in ihre Zahlen umrechnen:

```
>>> for buchstabe in text:
        print(ord(buchstabe))

72
97
108
108
111
```

Kannst du dieses kleine Programm so abändern, dass es das Wort „Hallo“ verschlüsselt und der selbe Geheimtext wie oben entsteht?

Beim verschlüsseln speichert man den Geheimtext am besten in einem neuen String. Dazu fängt man mit einem leeren String `neuer_text=''` an und fügt nach und nach die Buchstaben hinten an. Das Anfügen von Buchstaben geschieht mit dem Operator `+=`. Beispielsweise wird im Folgenden Buchstabe für Buchstabe das Wort „Hallo“ in den neuen String geschrieben.

```
>>> neuer_text = ''
>>> neuer_text += 'H'
>>> print(neuer_text)
H
>>> neuer_text += 'a'
>>> print(neuer_text)
Ha
>>> neuer_text += 'l'
>>> print(neuer_text)
Hal
```

```
>>> neuer_text += 'l'
>>> print(neuer_text)
Hall
>>> neuer_text += 'o'
>>> print(neuer_text)
Hallo
```

3 Ein ganzes Programm

Nun hast du alles zusammen und kannst ein Programm schreiben, das einen Text verschlüsselt oder auch entschlüsselt. Am Besten schreibst du das Programm in eine Datei und gehst wie folgt vor. Zunächst legst du die drei Variablen

```
code = 3
text = 'Hallo'
neuer_text = ''
```

an. In `text` steht der Text zum Verschlüsseln und in `neuer_text` soll der bearbeitete und somit verschlüsselte Text abgespeichert werden. In den bisherigen Beispielen haben wir den Text immer verschlüsselt, in dem wir die Buchstaben um 3 verschoben haben. Um das etwas allgemeiner zu halten und später auch ändern zu können verwenden wir nun die Variable `code`. Dies wird also der Schlüssel für unsere Verschlüsselung sein. Wenn wir früher `ord(H)+3` geschrieben haben, verwenden wir nun `ord(H)+code`. Auf diese Weise können wir einfach `code=23` setzen und das Programm funktioniert mit dem neuen Schlüssel 23.

Als nächstes schreibst du eine `for`-Schleife mit der du nacheinander alle Buchstaben bearbeiten kannst. In dieser Schleife verschlüsselst du nun Zeichen für Zeichen. Hierbei musst du jedoch drei Fälle unterscheiden: Ist das Zeichen ein Großbuchstabe oder ist es ein Kleinbuchstabe oder irgend etwas anderes. Hierzu verwendest du am Besten eine `if-elif-else` Anweisung. Diese könnte wie folgt aussehen, wobei die Stellen mit `...` noch zu füllen sind:

```
if  buchstabe >= 'A' and buchstabe <= 'Z':
    ...
elif ...:
    ...
else:
    ...
```

Handelt es sich um einen Groß- oder Kleinbuchstaben gehst du wie oben vor und hängst den verschlüsselten Buchstaben an `neuer_text` an. Allerdings darfst du nicht vergessen den Fall zu behandeln, dass du über das Alphabet hinaus läufst. Ist das Zeichen weder ein Groß- noch ein Kleinbuchstabe hängst du es einfach unverändert an den neuen Text an.

Ist die `for`-Schleife beendet und du hast alle Zeichen in `text` bearbeitet, so steht der verschlüsselte Text in `neuer_text` und du kannst ihn ausgeben.

Wenn dein Programm funktioniert, kannst du es auch in eine Funktion schreiben. Auf diese Weise ist es leichter verschiedene Texte zu verschlüsseln oder sie zu entschlüsseln. Beispielsweise könnte das wie folgt aussehen:

```
def caesar(text, code=3):
    neuer_text = ''
    ...
    print(neuer_text)

caesar('Hallo', 3)
caesar('Kdoor', 26-3)
```

4 Geheime Botschaft

Den folgenden Text habe ich mit `code=5` verschlüsselt. Kanst du ihn wieder entschlüsseln?

```
geheimtext = """
             Twiszsl gwfzhmy szw ijw Izrrj,
             ifx Ljsnj gjmjwwxhmy ifx Hmftx.
             Fqgjwy Jnsxyjns
             """
```