DS 14 – Teamprojekt: Verschlüsselung (Teil 2) und Berufe in der Softwareentwicklung

Informatik 10 – MINT-Zweig | Projektarbeit

Lernziele

Ich kann

- erklären, wie Entschlüsselung und Verschlüsselung bei der Caesar-Chiffre zusammenhängen, und eine Entschlüsselungsfunktion implementieren,
- mit dem Ausprobieren aller Schlüssel (Brute-Force) einen Geheimtext knacken und die Sicherheit der Caesar-Chiffre beurteilen,
- ein Programmierprojekt testen, dokumentieren und kurz vorstellen,
- Berufe, Berufsfelder und Berufsbiografien mit Bezug zur Softwareentwicklung mit seriösen Quellen recherchieren und die Ergebnisse geordnet festhalten.

🗣️ Sprachhilfe

- „Beim Entschlüsseln wird jeder Buchstabe um den Schlüssel **zurück** verschoben. Das ist dasselbe wie Verschlüsseln mit `26 - s`, weil …“
- „Der Hin-und-zurück-Test zeigt, dass …, denn nach dem Entschlüsseln kommt wieder … heraus.“
- „Die Caesar-Chiffre ist unsicher, weil es nur … sinnvolle Schlüssel gibt.“
- „Laut … (Quelle, Datum) gehört zu diesem Beruf …“

1 Rückblick: der Stand aus Teil 1

Eine geheime Nachricht nützt nur etwas, wenn man sie auch wieder **entschlüsseln** kann. Kopiert eure fertigen Funktionen aus DS 13 in die nächste Zelle. Sie sollten ungefähr so aussehen wie hier. Führt die Zelle aus: Es muss `KDOOR ZHOW` herauskommen.

```python
# Eure Funktionen aus DS 13 (oder diese Fassung)
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

def verschluessle_buchstabe(buchstabe, schluessel):
    position = alphabet.index(buchstabe)
    neue_position = (position + schluessel) % 26
    return alphabet[neue_position]

def verschluessle_text(text, schluessel):
    ergebnis = ""
    for zeichen in text.upper():
        if zeichen in alphabet:
            neu = verschluessle_buchstabe(zeichen, schluessel)
            ergebnis = ergebnis + neu
        else:
            ergebnis = ergebnis + zeichen
    return ergebnis

print(verschluessle_text("HALLO WELT", 3))
```

```
KDOOR ZHOW
```

> 📌 **Merksatz:** Verschlüsseln mit der Caesar-Chiffre: Position des Buchstabens bestimmen, um den Schlüssel verschieben, mit `% 26` umlaufen lassen und den Buchstaben an der neuen Position ablesen. Zeichen, die keine Buchstaben sind, bleiben unverändert.

2 Entschlüsseln: zurück verschieben

Beim Entschlüsseln rechnet man `- schluessel` statt `+ schluessel`. Was passiert am **Anfang** des Alphabets? A hat die Position 0, und 0 − 3 = −3.

🔮 **Vorhersage:** Was geben die beiden Zeilen aus?

```
print((0 - 3) % 26)
print((1 - 3) % 26)
```

```
23
24
```

In Python liefert `% 26` immer ein Ergebnis von 0 bis 25, auch bei negativen Zahlen. Aus A (0) wird X (23), aus B (1) wird Y (24).

Zweiter Weg: Stell dir das Alphabet als Uhr mit 26 Stunden vor. 3 Schritte zurück landen an derselben Stelle wie 23 Schritte vorwärts. Zurückschieben um `s` ist dasselbe wie Vorwärtsschieben um `26 - s`.

> 📌 **Merksatz:** Entschlüsseln mit dem Schlüssel `s` entspricht dem Verschlüsseln mit `-s` oder `26 - s`.

📝 Teilaufgabe D – Entschlüsseln

Schreibt eine Funktion `entschluessle_text(geheimtext, schluessel)`. Ihr könnt `verschluessle_text` wiederverwenden: Die Funktion braucht nur eine Zeile. Ersetzt die beiden `___`. Testet mit einem **Hin-und-zurück-Test**: verschlüsseln, dann wieder entschlüsseln. Es muss der ursprüngliche Text herauskommen (in Großbuchstaben).

Klartext	Schlüssel	verschlüsselt	wieder entschlüsselt	✓ oder ✗
`HALLO WELT`	3			
`Treffen um acht`	5			

Lückencode

```
def entschluessle_text(geheimtext, schluessel):
    return verschluessle_text(___, ___)

geheim = verschluessle_text("HALLO WELT", 3)
print(geheim)
print(entschluessle_text(geheim, 3), "erwartet: HALLO WELT")
print(entschluessle_text("YWJKKJS ZR FHMY", 5), "erwartet: TREFFEN UM ACHT")
```

3 Wie sicher ist die Caesar-Chiffre?

🚀 Erweiterung 1 – Brute-Force-Entschlüsselung (mit Gerüst)

Eine Funktion `brute_force(geheimtext)` probiert automatisch alle 26 Schlüssel durch und gibt alle Ergebnisse aus. So sucht man den richtigen Klartext von Hand heraus. Testet mit `KDOOR ZHOW`.

```python
def brute_force(geheimtext):
    # für jeden schluessel von 0 bis 25:
    #     schluessel und entschluesselten Text ausgeben
    pass

brute_force("KDOOR ZHOW")
```

🚀 Erweiterung 2 – Menü (selbstständig)

Erweitert euer Programm um ein Text-Menü: 1 = verschlüsseln, 2 = entschlüsseln, 3 = beenden. Das Menü erscheint so lange, bis „beenden“ gewählt wurde.

> 💡 **Tipp:** Vorher steht nicht fest, wie oft das Menü erscheint. Welche Schleife passt also?

⚖️ Beurteilung: Wie sicher ist die Caesar-Chiffre?

1. Wie viele verschiedene Schlüssel gibt es, die den Text tatsächlich verändern?
2. Wie lange braucht ein Programm, das **alle Schlüssel nacheinander ausprobiert** (**Brute-Force**, „rohe Gewalt“), um einen Geheimtext zu knacken?
3. Was müsste ein sicheres Verfahren anders machen?

Unsere Beurteilung:

…

4 Das Projekt abschließen

Zum Abschluss gehört mehr als ein laufendes Programm. Schreibt an den Anfang der Datei einen **Kommentarkopf**, zum Beispiel:

```
# Caesar-Chiffre: Texte ver- und entschlüsseln
# Team: Alex, Sam, Lea
# Funktionen: verschluessle_buchstabe, verschluessle_text,
#             entschluessle_text
# Bekannte Grenzen: Umlaute und ß werden nicht verschlüsselt.
```

📝 Projektabschluss

1. Ergänzt euer Programm um einen Kommentarkopf.
2. Führt mit dem fertigen Programm noch einmal **alle** Tests aus den Teilaufgaben A bis D durch.
3. Bereitet eine Vorstellung von etwa einer Minute vor: Was kann euer Programm? Aus welchen Funktionen besteht es? Welches Problem war am schwierigsten, und wie habt ihr es gelöst? Was würdet ihr noch verbessern?
4. Schaut im Team zurück:

Rückblick im Team	Stichpunkte
Das lief bei der Zusammenarbeit gut:	
Das würden wir beim nächsten Projekt anders machen:	
Das habe ich in diesem Projekt gelernt:	

---

5 🧭 Berufliche Orientierung: Berufe rund um die Softwareentwicklung

Euer Programm ist klein. Nach demselben Prinzip (planen, zerlegen, programmieren, testen, zusammenführen) entsteht auch große Software: in Autos und Smartphones, in Krankenhäusern, Banken und Behörden, in Spielen und Lern-Apps. Daran arbeiten Menschen mit ganz unterschiedlichen Ausbildungswegen.

Recherchiert in Partner- oder Kleingruppenarbeit zu **einem** Beruf bzw. Berufsfeld, zum Beispiel:

- Fachinformatikerin oder Fachinformatiker für Anwendungsentwicklung
- Softwareentwicklerin oder Softwareentwickler
- IT-Sicherheitsexpertin oder IT-Sicherheitsexperte (hier spielt Verschlüsselung eine große Rolle)
- Data Scientist oder Data Engineer
- UX/UI-Designerin oder UX/UI-Designer (Gestaltung von Bedienoberflächen)

> 📌 **Definition:** Eine **Berufsbiografie** beschreibt den beruflichen Werdegang einer realen Person: Schulabschluss, Ausbildung oder Studium, Berufseinstieg und die weiteren Stationen bis zur heutigen Tätigkeit.

Seriöse Quellen finden

- **BERUFENET** der Bundesagentur für Arbeit (`berufenet.arbeitsagentur.de`): Steckbriefe mit Aufgaben, Zugangswegen und Ausbildung.
- **planet-beruf.de** und **abi.de**: Berufs- und Studienwahl für Schülerinnen und Schüler, auch mit Erfahrungsberichten.
- **Hochschulkompass** (`hochschulkompass.de`): Studiengänge an deutschen Hochschulen.
- Die **Berufsberatung** der Agentur für Arbeit und das **Berufsinformationszentrum (BiZ)**.
- Für Berufsbiografien: Interviews und Porträts, z. B. auf Karriereseiten von Unternehmen. Bedenke: Unternehmen werben dort auch für sich.

> 💡 **Tipp:** Prüfe jede Information mit drei Fragen: **Wer** hat sie veröffentlicht? **Wann**? Wird etwas **verkauft oder beworben**? Bestätige wichtige Angaben mit einer zweiten Quelle. Vorsicht bei KI-Chatbots: Sie können überzeugend klingende, aber falsche Angaben erfinden. Schreibe nur auf, was du in einer Quelle gefunden hast, und notiere sie.

Leitfragen

1. Welche Aufgaben und Tätigkeiten gehören zu diesem Berufsbild?
2. Welche Ausbildung bzw. welches Studium wird benötigt?
3. Welche Programmiersprachen oder Werkzeuge werden häufig eingesetzt?
4. Wie hat sich das Berufsbild in den letzten Jahren verändert, z. B. durch KI-Werkzeuge?
5. Was interessiert euch persönlich an diesem Berufsfeld, oder was eher nicht?
6. Welche Person aus diesem Berufsfeld könnt ihr finden? Recherchiert ihren Werdegang.

📝 Berufsfeld-Steckbrief

(Doppelklick zum Bearbeiten dieser Zelle)

	Stichpunkte	Quelle
Beruf/Berufsfeld		
Aufgaben und Tätigkeiten		
Ausbildung oder Studium		
Werkzeuge und Programmiersprachen		
Veränderung des Berufsbilds		
Meine Einschätzung		–

📝 Berufsbiografie

Sucht den Werdegang einer **realen** Person, z. B. in einem Interview oder Porträt. Erfindet nichts: Was ihr nicht herausfinden könnt, kennzeichnet ihr mit „unbekannt“.

Station	Zeitraum	Was genau?
Schulabschluss		
Ausbildung oder Studium		
Berufseinstieg		
heutige Tätigkeit		
Quelle (mit Datum)		

🎤 Kurzbeitrag (eine Minute)

Was macht man in diesem Beruf? Wie kommt man dorthin? Was hat euch bei der Recherche überrascht?

Stichpunkte:

…

📌 Das Wichtigste in Kürze

- Entschlüsseln heißt: jeden Buchstaben um den Schlüssel **zurück** verschieben. Das entspricht dem Verschlüsseln mit `-s` oder `26 - s`.
- `% 26` liefert in Python auch bei negativen Zahlen ein Ergebnis von 0 bis 25.
- Mit **Brute-Force** probiert man alle Schlüssel aus. Weil es nur 25 sinnvolle Schlüssel gibt, ist die Caesar-Chiffre nicht sicher.
- Zum Projektabschluss gehören: alle Tests wiederholen, das Programm dokumentieren (Kommentarkopf), es vorstellen und im Team zurückblicken.
- Berufe und Berufsbiografien recherchiert man mit seriösen Quellen wie BERUFENET, gibt jede Angabe mit Quelle an und erfindet nichts dazu.

Zum Weiterüben: `DS14_Weitere_Uebungen.ipynb`.

Lösung

✅ Lösungen

Lösung

```python
# Lösung Teilaufgabe D
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

def verschluessle_buchstabe(buchstabe, schluessel):
    position = alphabet.index(buchstabe)
    neue_position = (position + schluessel) % 26
    return alphabet[neue_position]

def verschluessle_text(text, schluessel):
    ergebnis = ""
    for zeichen in text.upper():
        if zeichen in alphabet:
            neu = verschluessle_buchstabe(zeichen, schluessel)
            ergebnis = ergebnis + neu
        else:
            ergebnis = ergebnis + zeichen
    return ergebnis

def entschluessle_text(geheimtext, schluessel):
    return verschluessle_text(geheimtext, 26 - schluessel)

geheim = verschluessle_text("HALLO WELT", 3)
print(geheim)
print(entschluessle_text(geheim, 3))
print(entschluessle_text("YWJKKJS ZR FHMY", 5))
# Mit -schluessel geht es genauso, weil % 26 auch negative Zahlen richtig behandelt.
```

```
KDOOR ZHOW
HALLO WELT
TREFFEN UM ACHT
```

Lösung

```python
# Lösung Erweiterung 1 – Brute-Force
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

def verschluessle_buchstabe(buchstabe, schluessel):
    position = alphabet.index(buchstabe)
    neue_position = (position + schluessel) % 26
    return alphabet[neue_position]

def verschluessle_text(text, schluessel):
    ergebnis = ""
    for zeichen in text.upper():
        if zeichen in alphabet:
            neu = verschluessle_buchstabe(zeichen, schluessel)
            ergebnis = ergebnis + neu
        else:
            ergebnis = ergebnis + zeichen
    return ergebnis

def entschluessle_text(geheimtext, schluessel):
    return verschluessle_text(geheimtext, 26 - schluessel)

def brute_force(geheimtext):
    for schluessel in range(26):
        print(schluessel, ":", entschluessle_text(geheimtext, schluessel))

brute_force("KDOOR ZHOW")
# Nur bei Schlüssel 3 entsteht ein sinnvoller Text.
```

```
0 : KDOOR ZHOW
1 : JCNNQ YGNV
2 : IBMMP XFMU
3 : HALLO WELT
4 : GZKKN VDKS
5 : FYJJM UCJR
6 : EXIIL TBIQ
7 : DWHHK SAHP
8 : CVGGJ RZGO
9 : BUFFI QYFN
10 : ATEEH PXEM
11 : ZSDDG OWDL
12 : YRCCF NVCK
13 : XQBBE MUBJ
14 : WPAAD LTAI
15 : VOZZC KSZH
16 : UNYYB JRYG
17 : TMXXA IQXF
18 : SLWWZ HPWE
19 : RKVVY GOVD
20 : QJUUX FNUC
21 : PITTW EMTB
22 : OHSSV DLSA
23 : NGRRU CKRZ
24 : MFQQT BJQY
25 : LEPPS AIPX
```

Lösung

```python
# Lösung Erweiterung 2 – Menü (while, weil die Anzahl der Durchläufe offen ist)
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

def verschluessle_buchstabe(buchstabe, schluessel):
    position = alphabet.index(buchstabe)
    neue_position = (position + schluessel) % 26
    return alphabet[neue_position]

def verschluessle_text(text, schluessel):
    ergebnis = ""
    for zeichen in text.upper():
        if zeichen in alphabet:
            neu = verschluessle_buchstabe(zeichen, schluessel)
            ergebnis = ergebnis + neu
        else:
            ergebnis = ergebnis + zeichen
    return ergebnis

def entschluessle_text(geheimtext, schluessel):
    return verschluessle_text(geheimtext, 26 - schluessel)

auswahl = ""
while auswahl != "3":
    print("1 = verschlüsseln, 2 = entschlüsseln, 3 = beenden")
    auswahl = input("Deine Wahl: ")
    if auswahl == "1" or auswahl == "2":
        text = input("Text: ")
        schluessel = int(input("Schlüssel: "))
        if auswahl == "1":
            print("Ergebnis:", verschluessle_text(text, schluessel))
        else:
            print("Ergebnis:", entschluessle_text(text, schluessel))
    elif auswahl != "3":
        print("Ungültige Wahl.")
print("Programm beendet.")
```

```
1 = verschlüsseln, 2 = entschlüsseln, 3 = beenden
Deine Wahl: 1
Text: Hallo Welt
Schlüssel: 3
Ergebnis: KDOOR ZHOW
1 = verschlüsseln, 2 = entschlüsseln, 3 = beenden
Deine Wahl: 2
Text: KDOOR ZHOW
Schlüssel: 3
Ergebnis: HALLO WELT
1 = verschlüsseln, 2 = entschlüsseln, 3 = beenden
Deine Wahl: 3
Programm beendet.
```

Lösung

Lösung Beurteilung: Die Caesar-Chiffre ist **nicht sicher**. Es gibt nur 25 Schlüssel, die den Text verändern (26 wirkt wie 0). Ein Programm probiert sie in einem Bruchteil einer Sekunde alle durch, und den richtigen Klartext erkennt man sofort, weil nur er sinnvoll ist. Außerdem bleibt die Häufigkeit der Buchstaben erhalten: Der häufigste Buchstabe steht bei einem längeren deutschen Text wahrscheinlich für E. Ein sicheres Verfahren braucht so viele Schlüssel, dass Ausprobieren praktisch unmöglich ist, und darf die Buchstabenhäufigkeiten nicht durchreichen.

Berufsorientierung: Eine gute Vorstellung nennt, was das Programm kann, wie es aufgebaut ist (Alphabet, drei Funktionen, Hauptprogramm) und ein konkretes Problem mit Lösung. Beim Steckbrief kommt es darauf an, dass die Stichpunkte die Leitfragen konkret beantworten, jede Sachangabe eine Quelle hat und die

Einschätzung begründet ist. Beispiel: Fachinformatiker/-in für Anwendungsentwicklung, duale Ausbildung, in der Regel 3 Jahre (Quelle: BERUFENET).