LU03f - Eingaben in Python

Lernziele:

- Programme schreiben die einen Text vom User entgegenehmen können
- Sie wissen was ein „String“ in der Programmierung bedeutet.
- Sie wissen wie man Strings zusammenfügt (d. h. „verkettet“).

Eingabe von Zeichenketten

Mit einer Eingabe ist der vom Benutzer geschriebene Text gemeint, der vom Programm gelesen wird. Die Eingabe wird immer als Zeichenkette gelesen. Zum Lesen von Eingaben verwenden wir die Funktion/Methode, die in Python integriert ist. Die Funktion kann mit dem Befehl `input_string = input('…')` verwendet werden.

Im Folgenden finden Sie ein Beispiel für ein Programm, das eine Benutzereingabe verlangt, die vom Benutzer eingegebene Zeichenkette liest und sie anschließend ausgibt.

```
def user_input():
    input_string = input('Write a
message:\n')
    print(input_string)

if __name__ == '__main__':
    user_input()
```

Genauer gesagt, wird die Eingabe-Funktion gestartet und das Programm so lange angehalten, bis der Benutzer etwas schreibt. Wenn der Benutzer seine Eingabe durch das drücken der Eingabetaste abschliesst, wird die eingegebene Zeichenkette der String-Variablen `input_string` zugewiesen. Das Programm kann dann später auf die Variable `input_string` verweisen - im obigen Beispiel wird die Variable `input_string` im Druckbefehl `print(input_string)` referenziert.

Wenn das Programm ausgeführt wird, kann die Ausgabe wie im folgenden Beispiel aussehen. In diesem Beispiel hat der Benutzer den Text „Hello world“ geschrieben - Benutzereingaben sind in den Beispielen markiert.

```
Write a message:
User: <Hello World>
```

```
Hello World
```

Grundlagen der Zeichenketten

Wie Sie vielleicht schon bemerkt haben, sprechen wir in der Programmierung von „Strings“ und nicht von „Text“. Der Begriff „String“ ist eine Abkürzung für „ Zeichenkette“, die beschreibt, wie der Computer Text auf einer grundlegenderen Ebene sieht: als eine Folge von einzelnen Zeichen.

Bislang haben wir Strings auf zwei Arten verwendet. Beim Üben des Druckbefehls haben wir die zu druckende Zeichenkette in Anführungszeichen an den Druckbefehl übergeben, und beim Üben des Lesens von Eingaben haben wir die gelesene Zeichenkette in einer Variablen gespeichert.

In der Praxis sind Variablen benannte Behälter, die Informationen eines bestimmten Typs enthalten und einen Namen haben. Typischerweise, und in Python fast immer, wird einer Variablen bei ihrer Deklaration auch ein Wert zugewiesen. Sie können einen Wert zuweisen, indem Sie nach der Deklaration ein Gleichheitszeichen gefolgt von dem Wert setzen.

Eine String-Variable namens message, der der Wert „Hello world!“ zugewiesen wird, wird wie folgt deklariert:

```
message = 'Hello world!'
```

Wenn eine Variable erstellt wird, wird im Programm ein bestimmter Container zur Verfügung gestellt, auf dessen Inhalt später Bezug genommen werden kann. Variablen werden über ihren Namen referenziert. Das Erstellen und Ausdrucken einer String-Variable erfolgt beispielsweise wie unten dargestellt:

```
message = 'Hello world!'
print(message)

Hello world!
```

Eine Zeichenkette, die in Anführungszeichen einer Programmiersprache eingeschlossen ist, wird als „String-Literal“ bezeichnet, d. h. eine Zeichenkette mit einem bestimmten Wert. Ein häufiger Programmierfehler ist der Versuch, Variablennamen in Anführungszeichen zu setzen. Würde man die String-Variable `message` in Anführungszeichen setzen, würde das Programm den Text „message“ anstelle des Textes „Hello world!“ ausgeben, der in der Variablen `message` enthalten ist.

```
message = 'Hello world!'
```

```
print('message')

message
```

Verkettung - Strings zusammenfügen

Die zu druckende Zeichenkette kann mit dem Operator + aus mehreren Zeichenketten gebildet werden. Das folgende Programm gibt zum Beispiel „Hello world!“ in einer Zeile aus.

```
def main():
    text = 'Hello ' + 'world')
    print(text)

if __name__ == '__main__':
    main()
```

Die gleiche Methode kann verwendet werden, um ein String-Literal und den Wert einer String-Variablen zu verbinden. Wir verwenden aber konsequent den f-String um Variablen und String-Literale zu verbinden.

```
def main():
    message = 'Hello world!'
    print(f'{message} ... and the universe!')

if __name__ == '__main__':
    main()

Hello world! ... and the universe!
```

Wir können dasselbe mit einer beliebigen Anzahl von Zeichenketten tun.

```
def main():
    start = 'My name is'
    end = ', James Bond'
    print(f'{start} Bond {end}')

if __name__ == '__main__':
    main()

My name is Bond, James Bond
```

Programmausführung wartet auf Eingabe

Wenn die Programmausführung zu einer Anweisung kommt, die versucht, eine Eingabe vom Benutzer zu lesen (der Befehl `input()`), wird die Ausführung angehalten und gewartet. Die Ausführung wird erst fortgesetzt, wenn der Benutzer eine Eingabe gemacht und die Eingabetaste gedrückt hat.

Im folgenden Beispiel fordert das Programm den Benutzer zur Eingabe von drei Zeichenfolgen auf. Zunächst gibt das Programm „Write the first string:“ aus und wartet dann auf eine Benutzereingabe. Wenn der Benutzer einen Text schreibt, gibt das Programm „Write the second string:“ aus und wartet dann wieder auf eine Benutzereingabe. Dies wird ein drittes Mal wiederholt, bis das Programm alle drei Zeichenfolgen ausgibt.

```
def main():
    first = input('Write the first string:')
    second = input('Write the second
string:')
    third = input('Write the third string:')

    print('You wrote: ')
    print(first)
    print(second)
    print(third)

if __name__ == '__main__':
    main()

Write the first string:
User: <String number one>
Write the second string:
User: <String number two>
Write the third string:
User: <String number three>

You wrote:
String number one
String number two
String number three
```

Im vorherigen Beispiel haben wir die Benutzereingaben in drei verschiedenen String-Variablen gespeichert. Dies ist möglich, solange die Variablen alle unterschiedliche Namen haben.

Wir können kompliziertere Texte bilden, deren Inhalt sich je nach den Eingaben des Benutzers ändert, indem wir mehr Zeichenfolgen verwenden. Im folgenden Beispiel erfährt der Benutzer etwas mehr über die Texte, die er geschrieben hat - beachten Sie, dass die Reihenfolge, in der die Zeichenketten gedruckt werden, geändert werden kann. Im folgenden Beispiel wird die dritte Zeichenkette zuerst gedruckt.

```
def main():
    first = input('Write the first string:')
    second = input('Write the second
string:')
    third = input('Write the third string:')

    print(f'Last string you wrote was
{third}, which')
    print(f'was preceded by {second}.')
    print(f'The first string was {first}.')

    print('All together: ' + first + second +
third)

if __name__ == '__main__':
    main()
```

```
Write the first string:
User: <one>
Write the second string:
User: <two>
Write the third string:
User: <three>

Last string you wrote was three, which
was preceded by two.
The first string was one.
All together: onetwothree
```
