Kapitel 1
Programmieren für Einsteiger

Hier sind Sie richtig, wenn Sie noch nie programmiert haben. Erst einmal haben Sie die richtige Sprache erwischt, aus meiner Sicht natürlich das richtige Buch und vor allem nun auch noch den richtigen Abschnitt.

Sollten Sie allerdings schon Programmiererfahrung haben, werden Sie sich hier vielleicht langweilen. Und bevor Sie noch einschlafen und mir den Abschnitt verschnarchen, blättern Sie lieber zum nächsten Abschnitt weiter.

1.1 Was tut eigentlich ein Programmierer?

Programmieren ist eine schöne Beschäftigung. Sie können herumkommandieren, ohne dass es Ihnen irgendjemand übel nimmt. Anstatt selbst zu arbeiten, überlassen Sie dies dem Computer. Sie tun Dinge, die andere Leute nicht können. Und Sie erschaffen kreativ eine virtuelle Welt.

Ein Programm besteht aus Anweisungen, die Sie dem Computer vorlegen. Und er wird sie geduldig genau so ausführen, wie Sie sie formuliert haben, also auch mit den Fehlern, die Sie einbauen. Also werden Sie irgendwann nach diesen Fehlern suchen müssen. Der frustrierende Teil der Arbeit ist, dass Sie immer auf Ihre eigenen Fehler stoßen werden.

Da ein Computer von Haus aus nicht fantasiebegabt ist, sind die Anweisungen sehr einfach und klar. Man merkt an dieser Stelle, dass der Computer ursprünglich geschaffen wurde, um mathematische Probleme zu lösen. Weiß der Computer, wie er ein Problem lösen kann, dann kann er das immer wieder tun. Und er tut es eben sehr schnell. Sie werden noch sehen, dass man mit einem Computer neben der Mathematik noch anderen Unsinn machen kann.

Rechne du doch!

Hinweis

Damit Sie ein Bild über das Programmieren an sich gewinnen, werde ich Sie nun im Schnelldurchgang mit einigen Python-Listings belästigen. Geraten Sie nicht in Panik. Alles wird an anderer Stelle ausführlich beschrieben. Hier geht es nur um den Überblick. Entspannen Sie sich und lehnen Sie sich zurück!

Nehmen wir an, Sie wollen für die nächste Demo Tomaten kaufen. Sie wissen, dass fünf Tomaten 1,35 Euro kosten. Sie wollen aber 17 Tomaten, weil Ihr Patronengurt genau so viele aufnimmt. Sie erkennen sofort: Das ist der klassische Dreisatz. Sie zücken vielleicht schon Ihren Taschenrechner. Aber halt! Anstatt den Dreisatz jedes Mal selbst durchzurechnen, schreiben Sie ein Programm. Das bedeutet, Sie erklären dem Computer einmal, wie er mit dem Dreisatz umzugehen hat, und danach kann er das alleine. Leider müssen Sie dazu einmal darüber nachdenken, wie so ein Dreisatz funktioniert. Er besteht aus folgenden Schritten. Es sind drei. Logisch. Sonst hieße das Verfahren auch nicht Dreisatz.

1. Teile den Preis des Tomatenpakets durch die Anzahl der Tomaten.
2. Multipliziere diesen Preis mit der Zahl der gewünschten Tomaten.
3. Gib das Ergebnis auf dem Bildschirm aus.

Dies ist ein Programm. Allerdings ist die Programmiersprache für den Computer schwer nachvollziehbar. Der hätte es nämlich gern ein wenig formaler. Nehmen wir also irgendeine Programmiersprache. Wie wäre es mit Python?

```
PreisTomatenPaket = 1.35
Anzahl = 5
GewuenschteAnzahl = 17
PreisEinerTomate = PreisTomatenPaket / Anzahl
Ergebnis = PreisEinerTomate * GewuenschteAnzahl
print(Ergebnis)
```

Listing 1.1 Python im Tomatendreisatz

Wenn der Computer das Programm durchläuft, ermittelt er als Ergebnis 4,59. Wollen Sie eine andere Anzahl an Tomaten, ändern Sie im Programm den entsprechenden Wert und starten das Programm noch einmal. Ändert sich der Preis, ändern Sie ihn.

Natürlich gibt es auch eine Anweisung, mit der der Computer den Benutzer auffordert, Zahlen einzugeben. So müssen Sie nicht für jede Wertänderung das Programm anpassen. Aber lassen Sie mir dieses süße Geheimnis für später.

Eine Beschreibung zur Lösung eines Problems nennt man Algorithmus. Ein gut gegliederter Algorithmus kann leicht in ein Programm überführt werden.

Von Fall zu Fall

Das Programm könnte aber noch mehr. Wenn Sie ab 10 Tomaten Rabatt von 10 Prozent bekommen, würde dies der Computer auch berechnen können. Dazu schieben Sie an geeigneter Stelle den folgenden Algorithmus ein.

- Wenn die gewünschte Menge der Tomaten 10 oder höher ist ...
  - ... ziehe vom Einzelpreis 10 Prozent ab.

10 Prozent abzuziehen ist dasselbe wie 90 Prozent vom Preis. 90 Prozent von irgendetwas errechnet man einfach, indem man irgendetwas mit 0,9 multipliziert. Das vereinfacht den Programmiercode etwas. Die Abfrage wird dann an passender Stelle in unser Listing 1.1 eingebaut.

```
PreisTomatenPaket = 1.35
Anzahl = 5
GewuenschteAnzahl = 17
PreisEinerTomate = PreisTomatenPaket / Anzahl
if GewuenschteAnzahl >= 10:
    PreisEinerTomate = PreisEinerTomate * 0.90
Ergebnis = PreisEinerTomate * GewuenschteAnzahl
print(Ergebnis)
```

Listing 1.2 Tomatenberechnung mit Rabattabfrage

Sie könnten jetzt noch weitere Sonderfälle in das Programm einbauen. So könnte es sein, dass ein Beutel mit 7 Tomaten mit 1,40 billiger ist als die einzelnen Tomaten. Also soll der Computer doch bitte errechnen, wie viele Beutel ich kaufen muss, wie viele einzelne Tomaten und was das alles zusammen kostet.

Spicker ×

Das Prinzip ist, dass Sie dem Computer erklären, wie es gemacht wird, und der Computer erledigt die Rechnerei für Sie. Das gilt für Tomateneinkäufe wie für Börsengeschäfte oder Sparbriefe. Lassen Sie den Computer arbeiten!

Wiederholungstäter

Ihre Freundin erzählt Ihnen, dass der Nachbar seit einem Jahr mit dem Rauchen aufgehört hat und dadurch 1.000 Euro gespart hat, die er nun auf der Bank in einem Sparbrief angelegt hat. Sie rauchen zwar nicht. Diese Einnahmequelle fällt flach. Aber Sie beschließen, das Tomatenwerfen einzustellen, und wollen nun auch 1.000 Euro anlegen.

Bei Sparbriefen werden die Zinsen immer wieder am Ende des Jahres hinzugefügt und das führt uns zu einem weiteren Element der Programmierung, nämlich der Wiederholung. Der Programmierer spricht auch von Schleifen. Sie bekommen für einen Sparbrief jedes Jahr 6 Prozent Zinsen, die auf die Einlagen aufgeschlagen werden, also in den Folgejahren mitverzinst werden. Der Vertrag läuft über 7 Jahre. Wenn Sie das dem Computer vorlegen, können Sie ihm einfach sagen: Wiederhole diese Berechnung bitte 7 Mal.

Damit das Beispiel nicht zu trivial wird, nehmen wir noch an, dass Sie ab dem 3. Jahr sogar 9 Prozent Zinsen bekommen. Ich kann bei den Zinsen großzügig sein. Ich muss sie Ihnen ja nicht auszahlen.

- Die Geldanlage sind 1.000 Euro.
- Wiederhole 7 Mal:
  - Wenn das Jahr kleiner als 3 ist:
    - Schlage der Einlage 6 Prozent Zinsen zu.
  - Wenn das Jahr 3 oder größer ist:
    - Schlage der Einlage 9 Prozent Zinsen zu.
  - Erhöhe das Jahr um 1
- Gebe die Einlage auf dem Bildschirm aus.

Sie sehen, dass die Abfrage in die Schleife eingebaut wurde. Diese Art der Kombination der Programmierelemente ist zunächst vielleicht verwirrend, aber bei längerem Nachdenken logisch. Diese Art, Aufgaben zu formalisieren, damit der Computer es versteht, ist die Aufgabe eines Programmierers.

Der Vollständigkeit halber zeige ich Ihnen, wie das Programm aussieht, das aus dem Algorithmus hervorgeht. Vielleicht werden Sie die Befehle nicht verstehen. Keine Sorge, die werden in den nächsten Kapiteln im Detail erläutert. Wichtig ist nur, dass Sie sehen, wie sich aus dem Algorithmus das fertige Programm entwickelt.

```
Einlage = 1000
Jahr = 1
while Jahr <= 7:
    if Jahr < 3:
        Einlage = Einlage + Einlage * 0.06
    if Jahr >= 3:
        Einlage = Einlage + Einlage * 0.09
    Jahr = Jahr + 1
print(Einlage)
```

Listing 1.3 Sparbriefe

Tatsächlich kann der Computer gar nicht viel mehr als Abläufe, Abfragen und Schleifen. Programmiersprachen kommen nur deshalb nicht mit fünf Befehlen aus, weil es noch einige Mechanismen gibt, die dem Programmierer helfen, Programmteile zu strukturieren, um sie mehrfach verwenden zu können.

1.2 Wie der Computer mit Daten umgeht

Wenn Computer und Menschen über Zahlen reden, meinen sie nicht ganz dasselbe. Der Mensch verwendet das Dezimalsystem, vermutlich, weil er zehn Finger hat. Computer dagegen kennen nur zwei Zustände. Entweder es ist Strom auf der Leitung oder nicht. Aber genauso wie die meisten Menschen mit größeren Zahlen als 10 umgehen können, kann auch der Computer nicht nur bis 2 zählen. Durch das Verwenden mehrerer Stellen können beinahe beliebige Zahlen dargestellt werden, ob im Zehner- oder im Binärsystem.

In den meisten Fällen kann es Ihnen egal sein, wie Ihr Computer seine Zahlen intern darstellt. Ich will Sie damit auch gar nicht länger belästigen. Aber Sie werden feststellen, dass die 2 und ihre Potenzen magische Zahlen für Computer sind. Acht Stellen sind ein Byte. Und da 2^8 256 ist, können darin 256 Zustände abgespeichert werden. Ein Kilobyte sind 1024 Byte, weil 1024 2^{10} sind. Zur leichteren Unterscheidung vom Kilo Zucker wird das Kilo für Bytes mit einem großen K abgekürzt.

Die unterschiedlichen Codierungen haben für die Nachkommazahlen fatale Folgen. Sie kennen vielleicht das Ergebnis, wenn Sie in Ihrem Taschenrechner 1 durch 3 teilen. Das erwartete Drittel erscheint als eine endlose Kolonne von Dreien hinter der 0 vor dem Komma. Multiplizieren Sie diese Zahl dann mit 3, werden die meisten Taschenrechner spontan eine Reihe von Neunen auf dem Display haben. Wenn das ein Außerirdischer sähe, der mit drei Fingern geboren ist, würde er sehr verblüfft sein. Denn vermutlich hätte er ein Zahlensystem, das auf der 3 basiert, und kann sich nicht vorstellen, dass ein Drittel mal 3 irgendetwas anderes als 1 ergibt. Und ganz ähnlich geht es dem Computer mit dem menschlichen Zehntel, das binär codiert ebenfalls ein endloser Bruch ist.

Spicker ×

Für den Computer mit seinen zwei Ziffern ist ein Zehntel ein richtig fieser endloser Bruch. Dadurch kommt der Computer beim Rechnen mit Dezimalstellen manchmal zu unsauberen Ergebnissen.

Der Computer und seine Texte

Der Computer speichert die Buchstaben natürlich auch in Bytes, die wie erwähnt 256 Zustände kennen. Die englischen Buchstaben umfassen 26 Zeichen. Hinzu kommen die Großbuchstaben und die zehn Ziffern. Das sind also 62 verschiedene Zeichen. Wenn noch ein paar Satzzeichen wie Punkt, Komma und Fragezeichen hinzukommen, sind es schnell mehr als 64 Zeichen, so dass 6 Bits für die Aufnahme aller Zeichen nicht ausreichen. Darum wurden zunächst 7 Bits verwendet, die einen Zeichenvorrat von 128 ermöglichen. Das letzte Bit wurde erst einmal auf 0 belassen. Diese Codierung wurde als ASCII-Zeichensatz (American Standard Code for Information Interchange) genormt.

Wenn Sie dieses Buch lesen, werden Sie vermutlich die deutsche Sprache verwenden und ahnen, dass die Amerikaner beim ASCII nicht viel Aufwand um die Umlaute getrieben haben. Darum wird das Thema wiederkommen.

Spicker ×

Buchstaben sind aus Sicht des Computers durchnummeriert. Erst bei der Ausgabe werden aus den Zahlen dann Buchstaben. Da aber Buchstaben für uns Menschen so wichtig sind, werden Buchstaben von Computersprachen besonders behandelt.

1.3 Sprachbarrieren zwischen Mensch und Computer

Entgegen anderslautenden Gerüchten verstehen Computer uns Menschen gar nicht. Das Herz des Computers, oder besser das Gehirn des Computers, ist der Prozessor, der auch gern CPU (Central Processing Unit) genannt wird. Dieser Prozessor kennt eine übersichtliche Zahl von Befehlen, die einfach durchnummeriert sind. Hinter jedem Befehl steht, mit welcher Adresse oder Wert gearbeitet werden soll, und auch das ist nichts als eine Zahl. Solch einen Zahlensalat kann sich aber kein Mensch merken.

Um dieses gegenseitige Unverständnis zu überbrücken, sind Programmiersprachen entstanden, die etwas dichter an die Vorstellung des Menschen gerückt sind. Ihre Befehle bestehen aus englischen Wörtern. Damit sie auch der Computer versteht, müssen sie in seine Zahlenbefehle übersetzt werden. Dazu gibt es zwei Techniken:

- Ein Compiler ist ein Übersetzer, der wie ein Übersetzungsbüro arbeitet. Er nimmt den gesamten Text und übersetzt ihn einmal in die Zielsprache des Computers. Der führt die Übersetzung dann direkt aus. Zur Ausführung des Programms wird dann weder der Originaltext noch der Compiler benötigt.
- Ein Interpreter übersetzt den Programmtext wie ein Simultandolmetscher. Er liest den Text dem Computer quasi vor und der führt die Befehle quasi parallel aus. Bei jedem Neustart des Programms werden der Interpreter und der Originaltext benötigt.

Alles so englisch hier!

Ja, die Computerei ist voller Anglizismen. Die Befehle sind englisch, die Bezeichnungen sind englisch, selbst das Wort Computer ist englisch. Wenn Ihr Englisch nicht so toll ist, muss Sie dies nicht beunruhigen. Sie müssen genauso wenig Englisch lernen wie Ihr Apotheker Latein. Denn obwohl die meisten Präparate in der Apotheke lateinische Namen tragen, wird sich Ihr Apotheker schwertun, Julius Caesars Schriften über den

gallischen Krieg im Original zu lesen. Und wozu auch? Seit Gründung der Europäischen Gemeinschaft hat es keine kriegerischen Auseinandersetzungen zwischen Franzosen und Italienern mehr gegeben, wenn wir mal von den Stadienkämpfen während der verschiedenen Fußballturniere absehen.

Viele Computerbegriffe sind längst übersetzt, sodass man sich damit nicht wehtut. Ein paar angelsächsische Besonderheiten sollten Sie aber im Hinterkopf behalten.

1. Das Dezimalkomma bei Zahlen ist ein Punkt. Wenn im Programm eine 1.2 steht, ist damit eine deutsche 1,2 gemeint.
2. Beim Datum nennen die Amerikaner den Monat vor dem Tag.
3. Die Amerikaner können sich nicht vorstellen, wozu jemand Umlaute gebrauchen kann, der kein Fan von Heavy Metal ist.

Aber das sind keine Gründe, ein Anglistik-Studium als Voraussetzung für das Erlernen einer Programmiersprache anzusehen.