Lesetext Programmiersprachen

L203

Aufgabe:

Lies den folgenden Text einmal für dich durch. Markiere Stellen, die dir wichtig erscheinen.

Beantworte die Fragen am Ende des Textes möglichst genau.

Zeit: 45‘ **Sozialform:** EA

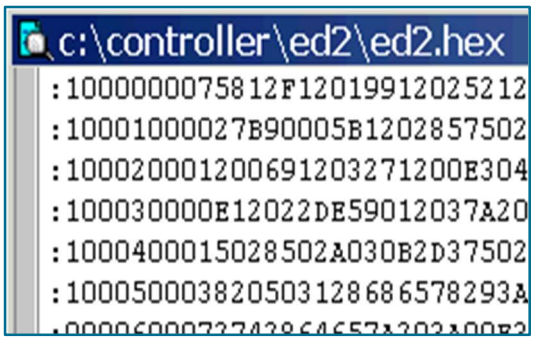
Was ist Programmieren? Um das zu erklären musst Du wissen, wie ein Computer funktioniert und was “programmieren” bedeutet.

Ein Computer besteht aus vielen Teilen, doch das Wichtigste ist der Prozessor, auch Central Processing Unit (CPU) genannt. Er berechnet die Aufgaben und koordiniert teilweise den Computer. Wenn man etwas programmiert, versteht man darunter, dem Computer Anweisungen zu geben, die er dann ausführt.

Ein Prozessor auf dem „Motherboard“

Wozu dienen Programmiersprachen?

Programmiersprachen sind dazu gemacht, einem Computer Anweisungen zu geben, damit er bestimmte Aufgaben ausführen kann. Solche Aufgaben sind etwa: Eine Meldung anzeigen, eine Rechnung ausführen, die dir selbst zu langweilig ist, oder eine bunte Grafik zeichnen. Leistungsfähige Computerprogramme - Textverarbeitungsprogramme, Bildbearbeitungsprogramme, Computerspiele - bestehen aus einer Vielzahl solcher Anweisungen, sogar bis zu mehreren Millionen. Gemeinsam allen Programmen, kleinen wie grossen, dass die Anweisungen nach ganz genauen Regeln aufgeschrieben werden müssen. In diesem Projekt wirst du einige Anweisungen der Programmiersprache Python kennen lernen und auch gleich ausprobieren, denn das geht mit Python ganz leicht. Am Ende wirst du dann aus ein paar solcher Anweisungen dein erstes Programm zusammenbasteln.



Maschinencode ist für Menschen nur sehr schwer zu verstehen.

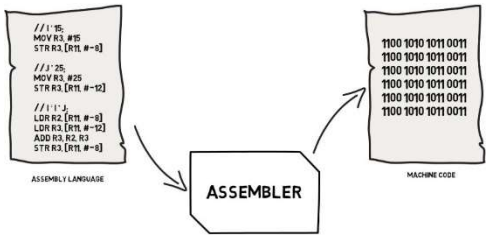
Nun ist es aber so, dass die CPU deines Computers, sein eigentliches Herzstück und Arbeitstier, nicht «Python kann». Die CPU ist ein Mikroprozessor also eine Maschine und versteht nur Maschinensprache, z.B 00001010011010110101001010100111. Maschinensprache braucht nur zwei Zeichen, 0 und 1. Das ist ihr ganzes Alphabet. Es ist so einfach, dass die vielen Millionen elektronischen Schalter in der CPU in rasender Geschwindigkeit Anweisungen abarbeiten können: Strom ein - Strom aus.

Damit das Programm interaktiv ist, wird anhand des Inputs (Eingabe), z.B. Tastatureingaben oder Mausposition, der Output (Ausgabe) erzeugt. Der Output ist das, was wir dann auf dem Bildschirm sehen oder als Blatt aus dem Drucker kommt. Was wäre ein Spiel ohne Eingabe, kein Spiel, weil es nicht interaktiv wäre? Es gibt viele verschiedene Eingabetypen, aber die wohl am häufigsten genutzte, ist die Tastatureingabe. Programmieren beinhaltet, einen Input durch logische und mathematische Operationen zu verarbeiten und so einen Output zu erzeugen.

Programmiersprachen sind Texte, die der Computer liest und ausführt. Aber Programmieren ist nicht so einfach, wie einen deutschen oder englischen Text zu schreiben. Mit Programmiersprachen kann man nichts erklären, sie dienen nur zum Geben von Anweisungen an den Computer und haben nur ein kleines “Vokabular”(vordefinierte Befehle). Wenn man einem Computer sagen will, dass der Himmel blau ist, dann kann man nicht schreiben “Himmel ist blau”. Ein Computer ist «dumm», er hat keine Intelligenz und weiss weder was ein Himmel ist, noch was blau bedeutet. Das führt dazu, dass etwas für Menschen so einfaches wie ”der Himmel ist blau” für einen Computer nicht zu verstehen ist.

Welche Programmiersprachen gibt es?

Ein Prozessor kann heute ungefähr 90 Befehle verstehen, die er dann selbst erledigt. Da diese Befehle für den Menschen kaum zu verstehen sind und so auch kaum jemand programmieren kann, wurden High-Level Programmiersprachen entwickelt. Gegenüber den einfacheren Low-Level Sprachen, auch Assembler Sprachen genannt, sind diese höheren Programmiersprachen gut zu verstehen und bieten deutlich mehr als 90 vordefinierte Befehle.



Assembler-Sprachen sind vom Prozessor abhängig. Läuft ein Assembler-Programm auf einem AMD Prozessor, wird dasselbe Programm auf einem Intel-Prozessor nicht funktionieren. Da Low-Levelsprachen (Assembler) sehr nah an der Maschinensprache (0100101000010) sind, ist die Rechengeschwindigkeit höher. Assemblersprachen findet man deswegen heute hauptsächlich nur noch in IoT-Systemen (Internet of Things) wie z. B: in einem Roboter oder sonstigen industriellen Controller-Systemen.

Die Grenze zwischen einfachen und höheren Programmiersprachen ist nicht eindeutig definiert. Eine höhere Programmiersprache ist eine Programmiersprache zur Abfassung eines Computerprogramms, die weit weniger abstrakt und weniger kompliziert als die Maschinensprachen ist. Mindestmerkmal ist, dass die Befehle höherer Programmiersprachen nicht unmittelbar von Mikroprozessoren verstanden und ausgeführt werden können. Die Befehle müssen durch Interpreter oder Compiler in Maschinensprache übersetzt werden.

Was ist ein Interpreter?

Damit dein Computer aber deine Python-Anweisungen verstehen und ausführen kann, braucht er einen Helfer, der ihm die Anweisungen dieser höheren Programmiersprache in Maschinensprache übersetzt. Dieser Helfer ist selbst ein Computerprogramm und bei der Software dabei, die du mit dem Python-System auf deinem Computer installiert hast: Der Name des Programms ist - wenig verwunderlich - Python und es wird im Computer-Fachchinesisch als Python-Interpreter bezeichnet.

Die Bezeichnung als „höhere“ Sprache bezieht sich in der Informatik nicht auf den Schwierigkeitsgrad, darin zu programmieren. Im Gegenteil sollen die höhere Programmiersprachen für die menschlichen Programmierer einfacher zu lesen und verstehen sein. Vielmehr bezieht sich das Attribut „höher“ auf die Abstraktionsebene der Programmiersprache. Vereinfacht kann man sagen, dass höhere Programmiersprachen mehr und komplexere logische Zusammenhänge mit weniger Text ausdrücken, der dann durch automatisierte Prozesse auf Maschinencode heruntergebrochen

wird. Die Lesbarkeit des Programmtextes wird so erhöht und das Programmieren, das Debbuging und die Wartung vereinfacht.

Es gibt über 1000 Programmiersprachen, doch es werden nur ca. 20 Sprachen häufig verwendet. Bekannt sind: *JavaScript, Java, PHP, Ruby, Phyton, COBOL, C, C++, C#, Visual Basic, BASIC, Batch, Java und Perl.*

Was braucht man zum Programmieren?

Nehmen wir mal an, du willst ein Programm programmieren. Was brauchst du dafür?

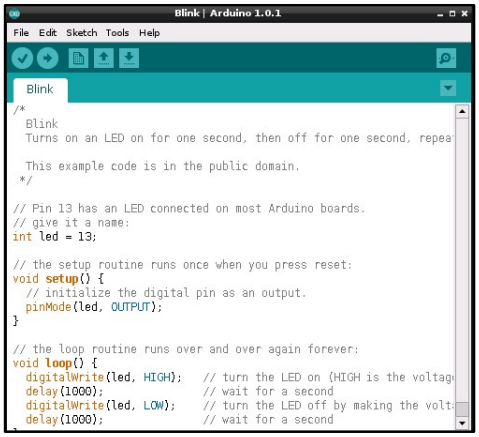
Erstmal einen Computer, der ist unverzichtbar, wenn du dein Programm ausführen willst. Aber was braucht man noch? Eigentlich fast nur Wissen über eine Programmiersprache. Fast, weil ich es voraussetze, dass ein heutiger Computer ein Betriebssystem mit einem Textverarbeitungsprogramm hat. Bei Windows können Sie zum Beispiel den Editor nehmen, dieser ist in jeder Windows Version vorhanden. Also hast Du bestimmt alles da, bloß was sollst Du jetzt in den Editor schreiben fragst Du dich vielleicht. Ganz einfach, Befehle. Befehle sind das, was mit der “Grammatik”, beim Programmieren Syntax genannt, eine Programmiersprache auszeichnen.

```python
import random

# VARIABLES
GAME_OVER = False
PLAYER_HEALTH = 5
PLAYER_FOOD_POUNDS = 500
MILES_TO_GO = 2000
CURRENT_DAY = 1
CURRENT_MONTH = 3
MONTHS_31_DAYS = [3, 5, 7, 8, 10, 12]
HEALTH_DECREASES_THIS_MONTH = 0


def should_decrease_health():
    global HEALTH_DECREASES_THIS_MONTH
    if HEALTH_DECREASES_THIS_MONTH < 2:
        random_day = random.randint(0, 30)
        if random_day < 2:
            return True
    return False
```

Python ist einfach, schlicht und gut lesbar

Aber wieso braucht eine Programmiersprache Grammatik (Syntax)? Schreibst du einen deutschen Text ohne Grammatik? Nein. Weil der Computer ohne diesen Syntax keinen Anfang und kein Ende eines Befehls erkennt, muss man ihm sagen, wo der Befehl anfängt und wo er aufhört. Das macht man mit Semikolons, Punkten, Absätzen und verschiedenen Klammern. Der Syntax ist recht unterschiedlich zwischen verschiedenen Programmiersprachen, aber wenn man eine Programmiersprache kann, ist es nicht mehr so schwierig eine Neue zu lernen.

Für «Arduino C» benutzt man ein Semikolon am Ende des Befehls, eine runde Klammer um Parameter (übergebene Werte, wie z.B. eine Zahl oder ein Buchstabe) zu markieren, geschweifte für Funktionen und Anführungszeichen um Text (Strings) erkenntlich zu machen. Das sind aber nur ein Paar der “Grammatikregeln”, aber wenn man das weiss, ist es schon die halbe Miete.



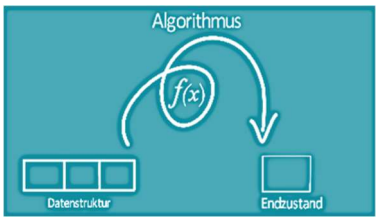
Arduino C ist für die Programmierung von Mikrokontrollern gedacht

Welche Elemente hat eine Programmiersprache?

Es gibt viele Elemente, die eine Programmiersprache hat. Alle zu erklären, wäre nicht wirklich hilfreich und würde dich nur verwirren. Stattdessen solltest du wissen, wie man nun etwas programmiert, dazu benötigst du nur Wissen über die Grundelemente. Wie du vielleicht weisst, rechnet ein Computer und versteht nichts, deshalb versucht ein Programmierer den Computer mathematische Bedingungen zu geben, um ihm zu erklären, dass er einen Befehl z.B. zehn Mal ausführen soll. Wenn man programmieren will, braucht man ein gutes mathematisches und logisches Verständnis.

Algorithmen und Datenstrukturen

Algorithmen und Datenstrukturen sind der Grundbestandteil der Informatik. Sie finden in fast allen Gebieten der Informatik Anwendung.



Datenstrukturen sind Objekte in der Informatik in denen Daten strukturiert werden. Es wird festgelegt, wie diese Daten angeordnet und gespeichert sind. Algorithmen bestimmen was mit diesen Daten gemacht wird. Sie legen ein Vorgehen fest, das aus einem Anfangszustand einen Endzustand macht. Beispiele für Datenstrukturen sind die einfachen Formen wie Strings oder Integer. Etwas komplexere Strukturen stellen die Arrays dar. Das Umwandeln dieser Daten in einen fertigen Endzustand wird dann mit einem Algorithmus erledigt.

Was ist ein Algorithmus?

Ein Algorithmus ist eine Beschreibung von Handlungsschritten. Man könnte auch Lösungsplan, Vorgehensweise, Lösung eines Problems oder Schritt-für-Schritt- Anleitung sagen. Algorithmen werden für die Verarbeitung von Datenstrukturen verwendet. Wichtig ist aber, dass ein Algorithmus hat nicht zwingend was mit einem Computer zu tun. Algorithmen gab es schon lange bevor es überhaupt Computer gab. Der Ursprung des Algorithmus liegt im menschlichen Verhalten. Wenn du jemanden einen Weg erklärst, dann ist die Erklärung ein Algorithmus.

Was sind Datenstrukturen?

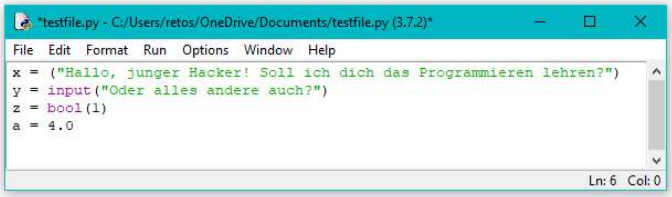
Datenstrukturen sind Objekte in denen Daten strukturiert und gespeichert werden. Die Datenstruktur beschreibt wie die Daten im Speicher abgelegt sind. Durch Algorithmen werden Sie dann organisiert. Man unterscheidet zwischen zwei Arten von Datenstrukturen: Die Statischen und die Dynamischen.

- Statische Datentypen haben ein festes Datenschema. Dieses kann nicht mehr geändert werden sobald es einmal festgelegt wurde.
  Beispiele: ***Integer, String, Boolean, Double Arrays***
- Dynamische Datenstrukturen erlauben die Anpassung des Speichers beim Hinzufügen oder Löschen von Daten während das Programm genutzt wird.
  Beispiele ***Listen, Bäume, Hashtabellen, Graphen, Schlagen, Stack***

Ein **Record** ist ein Verbund aus Daten. Zum Beispiel bilden die Daten wie Alter, Name, Adresse, Geburtstag und Geburtsort den demografischen Record einer Person.

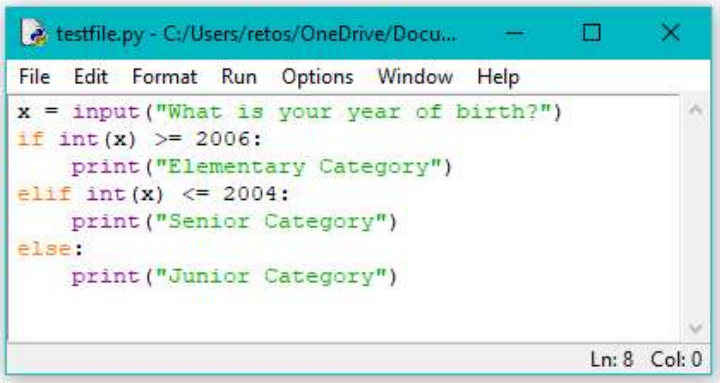
Was ist eine Variable?

Eine Variable ist eine Möglichkeit um im Programm eine Zahl, String (Buchstaben- und Zahlenwert mit dem nicht gerechnet werden kann) oder einen boolschen Wert (wahr oder falsch) zu speichern. In fast allen Programmiersprachen sind Variable der wichtigste Teil einer Programmiersprache, nur durch diese “Platzhalter” die durch einen eingegebenen Wert ersetzt werden können, ist es möglich interaktive Programme zu erstellen.



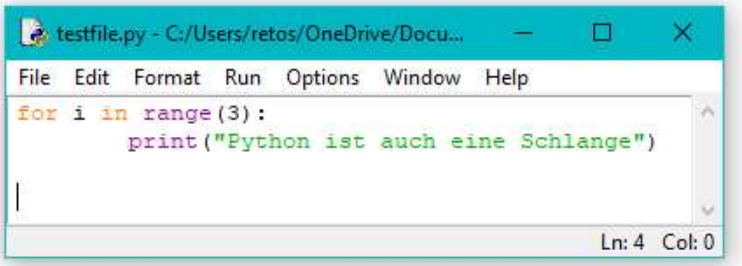
Was sind Verzweigungen?

Eine Verzweigung ist eine Abfrage mit einer Bedingung, trifft diese zu, so wird das, was innerhalb der Bedingung ist, ausgeführt. Um eine Bedingung zu erstellen gibt es Operatoren. Operatoren wären da: Gleich, ungleich, größer als, kleiner als, wahr und falsch.



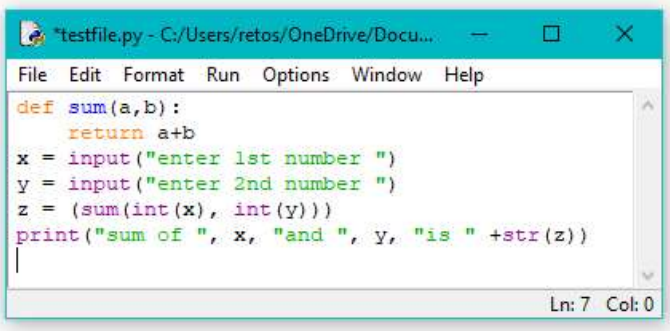
Was ist eine Schleife?

Eine Schleife enthält Code der solange ausgeführt wird, wie die Bedingung, die zur Schleife gehört wahr ist. Der Code kann also z.B. zehn Mal ausgeführt werden, das hat den Vorteil, das der Entwickler nicht zehn Mal den Code schreiben muss.



Was ist eine Funktion?

Eine Funktion ist eine Sammlung von Code, die jederzeit aufgerufen werden können. Z.B. für öfters anfallende Berechnungen schreibt man den Code in eine Funktion, so muss der Code nicht so oft geschrieben werden und kann einfach über den jeweiligen Funktionsnamen aufgerufen werden. Auch können Funktionen mit Parametern aufgerufen werden. Parameter sind Werte, wie z.B. eine Zahl. Diese Zahl kann von der Funktion wie eine Variable verwendet werden. So kann man mit derselben Funktion mehrere Rechnungen ausrechnen, die die gleiche Formel haben.



Fragen zum Textverständnis

1. Was ist und tut eigentlich eine Programmiersprache?
   Versuche, in eigenen Worten zu erklären.

2. Wortpaare finden

   Verbinde die Wortpaare, die zusammenpassen. Benutze unterschiedliche Farben.

Output	Grammatik
Syntax	Platzhalter
Computer	Anleitung
Variable	Rechner
Input	Tastatur
Algorithmus	Bildschirm
Schleife	Verzweigung
Bedingung	Wiederholung

3. Wortgruppen finden
   a) Suche Begriffe und Abkürzungen, die zusammenpassen und markiere diese mit einer gemeinsamen Farbe.
   b) Welcher dieser Begriffe ist ein Oberbegriff? Unterstreiche!

Sammlung von Code Gänsefüsschen
gut Programmiersprache Visual
Bravo Datentypen
Funktion wahr
Punkte
C++ gleich falsch Integer
Boolean
Parameter kleiner als Syntax
String
verschiedene Klammern
grösser als Absätze Variable
Java
ungleich Operatoren ungleich
und
Double Arrays
Semikolons gemacht
Python