Lektion 1
„Hallo Welt!“

Nach dieser Lektion können Sie ...

- grundlegende Operationen mit dem Programm Structorizer vornehmen,
- einfache Algorithmen mit Ein- und Ausgaben in Structorizer entwickeln und testen,
- das **Variable**nkonzept erklären,
- die grundlegenden **Datentypen** und ihre Eigenschaften benennen,
- die Grundrechenarten realisieren,
- das algorithmische Konzept der **Sequenz** erklären.

Bevor wir mit der Programmierung beginnen, werden wir uns mit Algorithmen beschäftigen. Diese Problemlösungen sind die Grundlage jedes Programms. Ein Computerprogramm setzt einen Algorithmus um. Daher ist es wichtig, dass Sie zunächst Probleme systematisch lösen, bevor Sie wild rumtippen und versuchen, ein Programm zu schreiben. Haben Sie die Lösungen, dann ist die Übersetzung in eine beliebige Programmiersprache i.d.R. recht einfach.

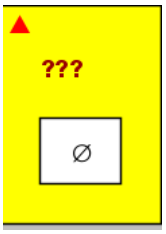
Structorizer

Das Programm, mit denen wir Probleme lösen, heißt Structorizer. Mit diesem Programm entwickeln Sie Algorithmen mit Nassi-Shneiderman-Diagrammen. Sie können Ihre Lösungen auch testen. Das hilft, um festzustellen, ob Ihre Lösung richtig ist.

Structorizer ist kostenfrei herunterladbar (https://structorizer.fisch.lu/), jedoch wird zum Betrieb das aktuelle Java Developement-Kit (JDK) benötigt. Dies ist ebenfalls kostenfrei (https://www.oracle.com/de/java/technologies/downloads/).

Hallo Welt! – Ihr erstes Programm

Klassischerweise ist das erste Programm, das Sie in einer Programmiersprache schreiben, das so genannte Hallo-Welt-Programm. Dieses hat selbst eine interessante Geschichte. Zur Programmierung starten Sie Structorizer.



Klicken Sie auf die drei Fragezeichen und geben Sie in „Diagrammname“ den Titel „HelloWorld“ ein.

Klicken Sie danach in das Programmfeld, dort steht jetzt Ø.

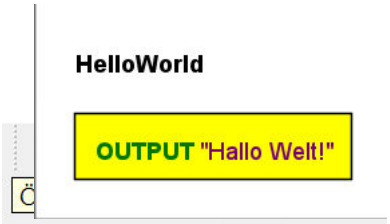
Wählen Sie danach die Option, eine Anweisung einzufügen (in Structorizer heißt das „Verarbeitung“).

In das Fenster, das sich öffnet, geben Sie im Feld „Einzugebender Text“ folgende Anweisung ein:

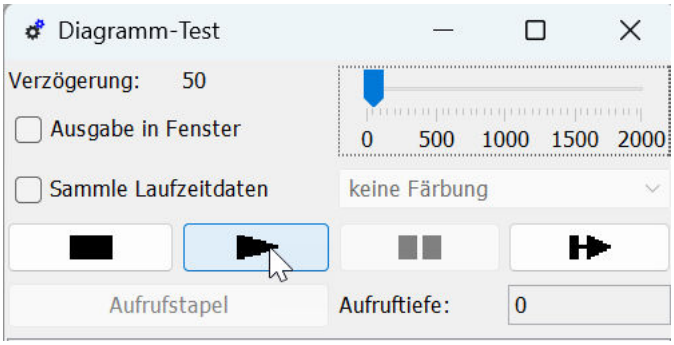
```
OUTPUT "Hallo Welt!"
```

Klicken Sie danach auf den Button „OK“.

Ihr Algorithmus müsste jetzt so aussehen:



Jetzt wird es Zeit, ihn zu testen. Klicken Sie auf das Debugger-Symbol. Das ist das Bild mit den beiden ineinandergreifenden Zahnrädern.



Es öffnet sich daraufhin ein Fenster. Dort gibt es einen Play-Button, auf den Sie bitte klicken. Jetzt wird der Algorithmus ausgeführt.

Structorizer müsste jetzt die Welt grüßen. Jetzt wissen Sie, wie man Algorithmen umbenennt, Anweisungen definieren und Algorithmen ausführen können.

Speichern Sie den Algorithmus unter dem Namen `01-01_HalloWelt.nsd` auf Ihrem H:-Laufwerk ab.

Hinweis

Die Dateinamen dieser Lektionen werden immer mit einem Zahlencode beginnen. Die erste Zahl steht für das Lektionen-AB und die zweite Zahl für den Algorithmus. Den Namen danach sollten Sie immer so wählen, dass klar ist, worum es in dem Algorithmus geht.

Wenn wir dann mit Python arbeiten, werde ich immer wieder auf die vorherigen Algorithmen zurückgreifen. Die Zahlencodes sind dann für die Identifikation des Algorithmus essentiell.

Aufgabe 1. Hallo ...

Ändern Sie das Programm „Hallo Welt“ so um, dass der Computer einen anderen Text ausgibt.

Sich selbst grüßen lassen – mit Eingaben arbeiten.

Eingaben sind wichtig für Programme. Schreiben Sie ein Programm mit den beiden Anweisungen

```
INPUT "Wie heißt du?", name
```

und

```
OUTPUT "Hallo ", name, "!"
```

Speichern Sie das Programm unter dem Name `01-02_HalloMitEingabe.nsd`.

Was macht das Programm, wenn Sie es laufen lassen?

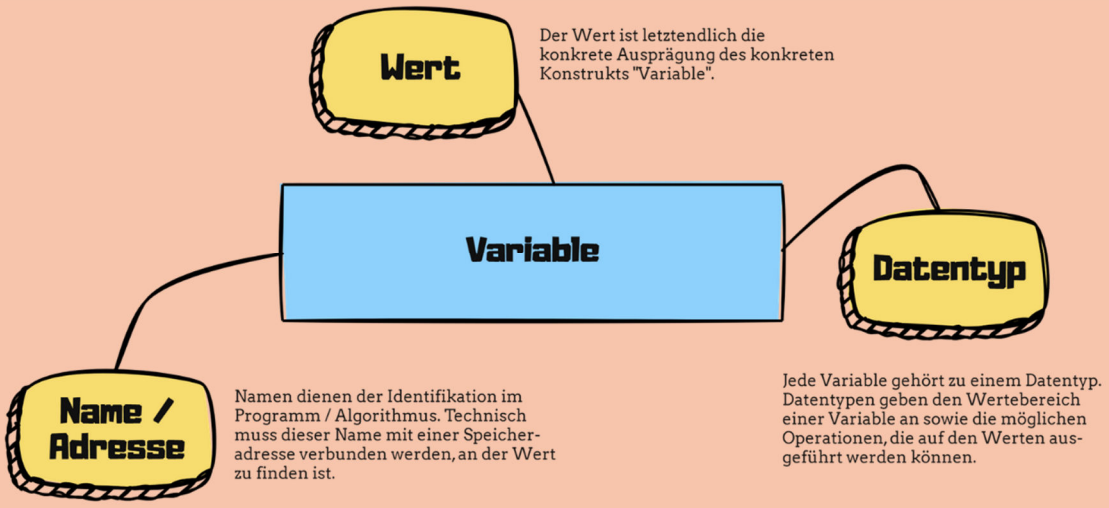
Aufgabe 2. Tageswiederholer.

Schreiben Sie ein Programm, das nach dem Wochentag fragt und dann die Information wiederholt (z.B. „Ach so, heute ist Freitag.“).

Theorie: Variablen

In dem Algorithmus gibt es die Variable `name`. Immer dann, wenn man sich in einem Algorithmus etwas merken muss, verwenden wir Variablen. Variablen haben drei Komponenten:

- Name/Adresse
- Wert
- Datentyp



Konventionen für Variablennamen:

- nur Kleinbuchstaben
- „sprechende Namen“, d.h., der Name gibt Auskunft über den Inhalt der Variable (`name` enthält einen Namen)

Man kann in Structorizer einer Variable einen Wert zuweisen, indem man den Operator `:=` verwendet, z.B.:

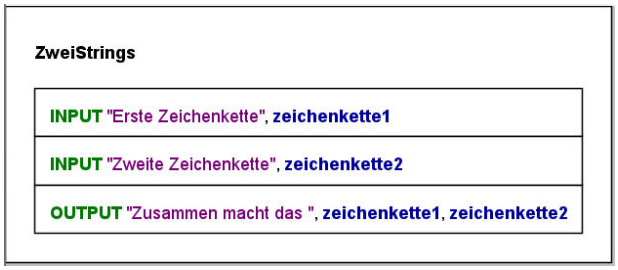
```
zahl := 42
zeichenkette := "Eine Zeichenkette"
```

Das bringt uns gleich zum Problem der Datentypen. Jeder Datentyp hat einen bestimmten Wertebereich und mögliche Operationen, die auf diesen Werten angewendet werden können. Hier eine kleine Übersicht über grundlegende Datentypen:

Datentyp	Wertebereich	Operationen
INTEGER	Ganze Zahlen	Grundrechenarten (+, -, *, /) Ganzzahldivision Modulo (%) Vergleich (>, <, =, >=, <=)
FLOAT	Kommazahlen	Siehe INTEGER
STRING	Zeichenketten	Konkatenation (Aneinanderreihung: +), Aussonderung, Längenbestimmung, Vergleich
BOOLEAN	Wahrheitswert (wahr, falsch)	NOT (nicht), AND (und), OR (oder)

Aufgabe 3. Zwei Zahlen mit Eingabe.
Fragen Sie in zwei **`INPUT`***-Aufforderungen nach den Zahlen a und b. Geben Sie danach die Ergebnisse folgender Operationen aus: Addition, Subtraktion, Multiplikation, Division. Vor der Ausgabe sollen die Werte der Variablen* `summe`, `differenz`, `produkt` *und* `quotient` *berechnet werden. Speichern Sie den Algorithmus als* `01-03_ZweiZahlen.nsd` ab.

Wenn man nach zwei Zeichenketten fragt und diese dann zusammen ausgibt, sieht der Algorithmus folgendermaßen aus:



Theorie: Sequenz

Anweisungen eines Algorithmus werden hintereinander abgearbeitet. Die Kunst der Algorithmenentwicklung ist es, die zur Verfügung stehenden Anweisungen so hintereinander abzuarbeiten, dass ein gegebenes Problem gelöst wird.

Aufgabe 4. Was ist Modulo?

Schreiben Sie einen Algorithmus, der nacheinander folgende Werte berechnet und ausgibt: `9 % 3`, `10 % 3`, `11 % 3`, `12 % 3`. *Speichern Sie den Algorithmus unter dem Namen* `01-04_Modulo.nsd` *ab. Geben Sie an, was mit Modulo berechnet wird.*

Aufgabe 5. Prozentrechner.

Entwickeln Sie einen Algorithmus, der zunächst nach einem Betrag und dann nach einem Prozentsatz fragt und danach den Prozentwert berechnet (z.B. Eingabe: 200, Prozentsatz: 10, Ausgabe: 20). Speichern Sie den Algorithmus unter dem Namen `01-05_Prozent.nsd` *ab.*

Aufgabe 6. Projekt.

Wir schreiben ein Programm, in dem zwei Zeiten addiert werden. Es werden für jede Uhrzeit getrennt jeweils Stunden, Minuten und Sekunden eingegeben. Das Ergebnis soll die Anzahl der Tage, Stunden, Minuten und Sekunden angeben.

a) *Algorithmenentwicklung fängt oft mit Papier und Bleistift an. Ein Problem kann nur gelöst werden, wenn man es analysiert hat. Gehen Sie mit folgenden Schritten vor:*
   - *Addieren Sie zunächst zwei Zeiten, wobei nur Minuten und Sekunden gegeben sind. Überlegen Sie, wie Sie mittels der Modulo-Operation entsprechende Probleme des Überlaufs (mehr als 60 Sekunden) lösen können.*
   - *Erweitern Sie die Lösung auf das Gesamtproblem.*

b) *Setzen Sie Ihre Problemlösung mit Structorizer um. Gehen Sie davon aus, dass als Eingabewerte nur gültige Werte angegeben werden.*

c) *Definieren Sie Testfälle:*
   - *Diese sollten Extremfälle enthalten.*
   - *Diese sollten Fälle zwischen den Extrema enthalten, die besondere Lösungen liefern (nur Stunden etc.).*
   - *Diese sollten Fälle enthalten, die Sie willkürlich auswählen.*

Speichern Sie Ihre fertige Lösung unter dem Namen `01-06_ZweiZeiten.nsd` *ab.*