Gliederung des Kapitels 1.1

1.1 Algorithmen und Sprachen

~~1.1.1 Darstellung von Algorithmen~~
~~1.1.2 Grundlegende Datenbereiche~~
~~1.1.3 Realisierte Abbildung~~
~~1.1.4 (Künstliche) Sprachen~~
~~1.1.5 Grammatiken~~
~~1.1.6 BNF, Syntaxdiagramme~~
~~1.1.7 Sprachen zur Beschreibung von Sprachen~~
1.1.8 Historische Anmerkungen
1.1.9 Übungsaufgaben

1.1.8 Historische Anmerkungen

Zum Begriff Algorithmus: Hiermit verbindet man das griechische Wort arithmós (Zahl), vor allem aber den arabisch-persischen Mathematiker Mohamad Ibn Musa Al-Chwarismi; dieser lebte von 780 bis ca. 850 n.Chr., stammte aus der Region südöstlich des Kaspischen Meeres, arbeitete am Hof des Kalifen von Bagdad und hat neben anderen das "Kitab al-jabr w'al-muqabala" (das Buch über die "Regeln der Wiedereinsetzung und Reduktion") geschrieben; es behandelt die Lösungen von linearen und quadratischen Gleichungen; im Titel tritt das hier mit "Wiedereinsetzung" übersetzte Wort "algebra" erstmals in der Mathematik auf. In der lateinischen Fassung wird der Autor mit "Algorithmi" bezeichnet. Hieraus entstand das Wort Algorithmus als Begriff für "exaktes Rechenverfahren".

Algorithmen sind als mathematische Lösungsverfahren recht alt. Der Euklidische Algorithmus zur Berechnung des größten gemeinsamen Teilers natürlicher Zahlen

while b≠0 do r := a mod b, a := b; b := r od

stammt aus der Zeit um 300 v. Chr. Das Newtonsche Verfahren zur Berechnung einer einfachen Nullstelle von stetig differenzierbaren Funktionen f (wähle einen Anfangswert für x und die Genauigkeit δ geeignet)

while |f(x)| > δ do x := x - f(x) / f'(x) od

wird seit ca. 1670 verwendet. Das Gaußsche Eliminationsverfahren zur Lösung linearer Gleichungssysteme Algorithmus wird seit Anfang des 19. Jahrhunderts eingesetzt.

Die formale Definition für "Algorithmus " erfolgte 1936 unabhängig voneinander in drei Arbeiten, und zwar über den Lambda-Kalkül (dieser λ-Kalkül bildet die Grundlage der funktionalen Programmierung) von Alanzo *Church* (1903-1995), über μ-rekursive Funktionen (dies sind einfache Programme mit den Konstruktoren ";", "if-then" und "while") von Steven Cole *Kleene* (1909-1994) und über eine auf Zeichen arbeitende Maschine (die Turingmaschine) von Alan Mathison *Turing* (1912-1954).

Ab nun wurde es möglich, Aussagen über Algorithmen herzuleiten und "Unmöglichkeitsbeweise" zu führen. In der Folgezeit wurden weitere Kalküle als gleichwertig zu den drei oben genannten Darstellungen für Algorithmen nachgewiesen.

Zur Darstellung der Grundbereiche: Für Boolean und character sind sie recht alt (Lateinisches Alphabet plus Satzzeichen) und für Zeichen durch den ASCII-Code Anfang der 1950er Jahre fixiert.

Das dezimale Stellenwertsystem für natürliche Zahlen hat sich ab dem 7. Jahrhundert in Indien entwickelt und gelangte durch die Araber bis Ende des 12. Jahrhunderts nach Europa. Das mathematische Standardwerk "liber abbaci", das "Buch der Rechenkunst", von Leornardo Pisano (bekannt als Fibonacci) aus dem Jahre 1202 verwendet und verbreitet dieses System im Westen. Hier treten auch erstmals negative Zahlen auf. Somit gibt es bereits seit 700 Jahren die uns geläufige Darstellung für den Datentyp integer.

Gottfried Wilhelm Leibniz beschreibt 1697 das Binärsystem, das vieles vereinfacht. Aber erst nach 1930 werden diese Ideen technisch für den Bau von Rechenmaschinen umgesetzt.

Erweitert man das Dezimalsystem auf Brüche, so erhält man die Darstellung des Datentyps real. Diese Erweiterung nahm erstmals Al-Kasi, Direktor der Sternwarte von Samarkant, 1427 vor.

Die Darstellung im Zweierkomplement erfolgte mit der Entwicklung digitaler Rechenautomaten in den 1940er Jahren.

Zur Programmierung: Charles Babbage (1791-1871) entwirft ab 1838 die "Analytical Engine", eine modern anmutende Rechenmaschine mit Steuerwerk, Rechenwerk und Programmspeicher. Seine Assistentin ist Ada Augusta countess of Lovelace (1815-1852), die in Ergänzung eines von ihr übersetzten Artikels die Verwendung von Programmen anregt und hierfür erste Programme schreibt. So wurde sie zur ersten Programmiererin. Diese Arbeiten geraten jedoch in Vergessenheit und werden erst nach dem Bau der ersten Computer wieder entdeckt.

Die Programmierung beginnt ab 1940 mit Abfolgen von Maschinenbefehlen, ab 1956 mit Programmen in den Programmiersprachen FORTRAN, ALGOL, LISP, APL und COBOL. Das Programmieren wird das Kerngebiet für die neue Wissenschaft "Informatik".

Zu Grammatiken und BNF: Die erste Arbeit hierzu stammt 1959 von Noam Chomsky. Hierin sind die Typ 0, 1, 2 und 3 Grammatiken und Sprachen und diverse Eigenschaften beschrieben. Es folgen viele Arbeiten, vor allem über kontextfreie Grammatiken. Die Backus-Naur-Form entstand im Rahmen der Entwicklung der Programmiersprache ALGOL ab 1958; sie ist nach ihren wesentlichen Erfindern, dem Amerikaner J. Backus und dem Dänen P. Naur, benannt. Die Syntax der meisten heutigen Programmiersprachen wurde in EBNF formuliert. Die Syntaxdiagramme wurden als Veranschaulichung von BNF und kontextfreien Grammatiken in den 1960er Jahren vorgeschlagen.

1.1.9 Übungsaufgaben

Aufgabe 9:
Definiere die Syntax der BNF mit Hilfe der BNF.

Aufgabe 9:
Definiere die Syntax der BNF mit Hilfe der BNF.

1.1.9 Übungsaufgaben

Aufgabe 1: *Multiplikation*
Beschreiben Sie die Multiplikation zweier natürlicher Zahlen.

a) Verwenden Sie eine iterierte Addition.
b) Beschreiben Sie den zeichenweise arbeitenden Algorithmus, den Sie aus der Grundschule kennen.
c) Nehmen Sie an, Sie hätten als Operation die Bildung des Quadrats einer Zahl zur Verfügung, d.h., es gibt den Operator square(x), der zu einer natürlichen Zahl x das Quadrat x^2 liefert. Wie kann man dann die Multiplikation durchführen?

Aufgabe 2: *Berechnung von div und mod*
Der Euklidische Algorithmus (Beispiel 1.1.2.3) benutzt die Operation mod, die zu zwei natürlichen Zahlen a und b den Rest liefert, der bei der ganzzahligen Division a div b von a durch b übrig bleibt. Es gilt: $a = b \cdot (a \text{ div } b) + a \text{ mod } b$.
Geben Sie zwei Algorithmen an, die a div b und a mod b gleichzeitig berechnen, und zwar

a) einmal zeichenweise (z.B. als array dargestellt) und
b) einmal ohne die Darstellung mit Ziffern zu verwenden.

Prüfen Sie Ihre Algorithmen mit einigen Ablaufprotokollen.

Aufgabe 3: *Darstellung zu einer Basis*
Geben Sie einen Algorithmus an, der zu zwei natürlichen Zahlen x und b die Ziffern der Darstellung von x zur Basis b ausgibt. (Die Ziffern sollten Sie in der Form (i) ausgeben, wobei i die eigentliche Ziffer ist, $0 \leq i < b$.)

Aufgabe 4: *Zeichenverschiebung und quadratische Codierung*
Es sei pos: **Â** → {0, 1, 2, ..., 127} die Funktion, die jedem Zeichen seine Nummer im ASCII-Code zuordnet (siehe 1.1.2.6).
Es sei val: {0, 1, 2, ..., 127} → **Â** die Funktion, die jeder Zahl zwischen 0 und 127 das zugehörige Zeichen bzgl. des ASCII-Codes zuordnet.
a) Schreiben Sie einen Algorithmus, der zunächst eine Zahl k und danach eine Folge von Zeichen (das Ende der Folge sei durch das Zeichen '&' bestimmt) einliest und die um k Stellen im ASCII-Code verschobene Folge dieser Zeichen ausgibt.
b) Wir wollen die Zeichen durch folgende Vorschrift verschlüsseln:
Wenn das Zeichen α die Nummer $pos(\alpha) = i$ besitzt, dann berechnen wir $j = i^2 \text{ mod } 128$ und ersetzen α durch das Zeichen $\beta = val(j)$.
Beispiel: α = 'U', pos(U) = i = 85, $j = 85^2 \text{ mod } 128 = 7225 \text{ mod } 128 = 57$, β = val(57) = '9'.
Schreiben Sie einen Algorithmus, der diese Verschlüsselung vornimmt, der also die Abbildung $\alpha \rightarrow \beta = val((pos(a))^2 \text{ mod } 128)$ berechnet.
Untersuchen Sie, ob diese Abbildung injektiv ist (siehe Folie 138).
Diskutieren Sie Vor- und Nachteile dieser Abbildung.

Aufgabe 5: *28 Zahldarstellungen natürlicher Zahlen*
a) Schreiben Sie die Zahlen 5, 26, 144, 5040 jeweils zur Basis 2, 3, 7, 16, -2 und -3.
b) Schreiben Sie diese vier Zahlen als Viertupel (r_1, r_2, r_3, r_4) wobei r_i die Reste bzgl. der Zahlen $q_1 = 3$, $q_2 = 11$, $q_3 = 17$ und $q_4 = 26$ entsprechend Hinweis 3 nach 1.1.2.8 sind. Multiplizieren Sie 5 und 26 sowie 144 und 5040 in dieser Reste-Darstellung und prüfen Sie Ihre Resultate.

Aufgabe 6: *10 Zahldarstellungen rationaler Zahlen*
a) Schreiben Sie die Zahlen 3.4, 169.52 und -888.88 zur Basis 2 und 12 mit 9 Nachkommastellen (vgl. 1.1.2.12) sowie in der Gleitpunktdarstellung mit Mantissenlänge 11 und Exponentenlänge 5 (vgl. 1.1.2.14 bis 16).
b) Schreiben Sie -8.86 und -0.14 zur Basis 2 mit dem Zwei-Komplement und addieren Sie sie in dieser Darstellung.

Aufgabe 7: *Datentypdeklarationen*
Beschreiben Sie folgende Daten mit Hilfe der in Kapitel 1.1. vorgestellten Datentypen und Konstruktoren:
a) Liste der Nummern der Großbuchstaben im ASCII_Code.
b) Menge der Vornamen aller Freunde und Freundinnen.
c) Zuordnung der Geburtsjahre zu jedem/r Freund/Freundin.
d) Datentyp für die Wochentage und Datentyp für fünf selbst zu definierende Wetterlagen (kühl, regnerisch, sonnig, ...).
e) Zuordnung von Wochentag und Wetterlage zu der bei dieser Situation geeigneten Kleidung.
f) Jede Teilmenge einer angeordneten Menge M mit n Elementen kann man durch einen n-stelligen Booleschen Vektor beschreiben, dessen i-te Komponenten genau dann true ist, wenn das i-te Element in der Teilmenge liegt. Definieren Sie auf diese Weise den Datentyp "Menge der Teilmengen von M".

Aufgabe 8: *Terminierung?*
Ermitteln Sie, für welche Eingabewerte folgende Programme terminieren und welche Funktion sie realisieren.

```
program P1 is
k, m: integer;
begin  read (m);  k := 1;
       while m > k do  k := k+2; m := m+1 od;
       write (k)
end;
```

```
program P2 is
k, m: integer;
begin  read (m);  k := -m;
       while m > 0 do  k:=k+2; m:= k od;
       write (k)
end;
```


Aufgabe 9: *Terminierung?*
Ermitteln Sie, für welche Eingabewerte folgendes Programm terminiert (zu "pos" siehe Aufgabe 4).

```
program unbekannt is
k: integer; X: character; DB: array [0..3] of character;
begin
       for k:=0 to 3 do DB[k] := 'Z' od;
       read (X);
       while X ≠ 'Z' do
              k := pos (X);
              while DB[k] ≠ 'Z' do  k := (k+1) mod 4  od;
              DB [k] := X
       od;
       for k:=0 to 3 do write (DB[k]) od;
end;
```

Aufgabe 10: *Operationen auf Sprachen (vgl. 1.1.4.6)*
Es sei A die zweielementige Menge A = {a, b}.
(1) Bilde zur Menge L = {a, bb} die Mengen LL und LLL.
(2) Skizzieren Sie ein Verfahren, das zu einem Wort $w \in A^*$ feststellt, ob $w \in L^*$ gilt oder nicht.
(3) Sei J = {ab}. Beschreiben Sie die Mengen J^* und $A^* \setminus J^*$.
(4) Sei K = {b, ab}. Beschreiben Sie die Menge $(K \cup L)^*$.
(5) Beschreiben Sie die Menge $\{(abb)^n \mid n>0\}$ mit Hilfe der Sprachen J und K und der Sprachoperationen Vereinigung, Differenz, Durchschnitt, Konkatenation und/oder Iteration.
(6) Sei H = {ba, bb}. Geben Sie einen Algorithmus für das Wortproblem der Menge $(H \cup J)^*$ an, d.h., geben Sie ein Verfahren an, das zu jedem Wort $w \in A^*$ feststellt, ob $w \in (H \cup J)^*$ gilt oder nicht.

Aufgabe 11: *Erzeugte kontextfreie Sprachen*
Es seien V = {S, A} und Σ = {0, 1}. Welche Sprachen L_i werden folgenden kontextfreien Grammatiken $G_i = (V, \Sigma, P_i, S)$ erzeugt mit

$P_1 = \{ S \to 0, S \to 1 \}$
$P_2 = \{ S \to 1, S \to S, S \to 0, A \to 0 \}$
$P_3 = \{ S \to 0S, S \to 00 \}$
$P_4 = \{ S \to S1, S \to 00, S \to 1, A \to 1, A \to AS \}$
$P_5 = \{ S \to 0A0, S \to 1, A \to 0S0 \}$
$P_6 = \{ S \to 0AS, S \to 0, A \to 1A1, A \to 1 \}$
$P_7 = \{ S \to AA, A \to 0A0, A \to 1 \}$
$P_8 = \{ S \to AS, A \to \varepsilon, A \to 0S1, S \to A \}$
$P_9 = \{ S \to SAS, A \to 10, A \to 0S1A, S \to A \}$

Aufgabe 12: *Grammatiken konstruieren*

Konstruieren Sie Grammatiken $G_i = (V_i, \Sigma, P_i, S)$ mit $\Sigma = \{0, 1\}$, die folgende Sprachen L_i erzeugen (hier ist w^R das gespiegelte Wort w, d.h., wenn $w = a_1a_2...a_n$ ist, dann ist $w^R = a_n..a_2a_1$):

$L_1 = \Sigma^* \setminus \{\varepsilon\} = \Sigma^+$

$L_2 = \{ w \in \Sigma^* \mid w \text{ hat eine gerade Länge} \}$

$L_3 = \{ 0^n10^n \mid n > 0 \}$

$L_4 = \{ (0^n10^n)^m \mid n > 0, m > 0 \}$

$L_5 = \{ ww^R \mid w \in \Sigma^* \}$

$L_6 = \{ w^2 \mid w \in \Sigma^* \}$

$L_7 = \{ 0^n10^n10^n \mid n > 0 \}$

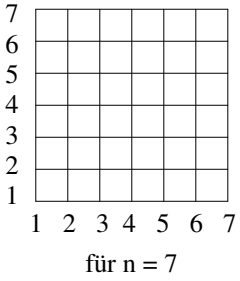
$L_8 = \{ w \in A^* \mid w \text{ besitzt gleich viele Zeichen 0 und 1} \}$

$L_9 = \{ 0^n \mid \text{Es gibt ein } k \geq 0 \text{ mit } n = 2^k \}$

Aufgabe 13: *Charakterisierung von "diagonalen Wegen"*

Es sei ein n×n-Gitter gegeben:

Es sollen genau die Wege beschrieben werden, die vom Punkt (1,1) unten links zum Punkt (n,n) oben rechts führen. Hierfür bedeute das Zeichen a "gehe zum nächsten Punkt rechts" und b bedeute "gehe zum nächsten Punkt nach oben". Das Wort abab..ab = $(ab)^n$ beschreibt dann einen solchen "diagonalen Weg" ebenso wie das Wort a^nb^n.



für n = 7

a) Beschreiben Sie die Menge aller Wörter, die einen Weg vom Punkt (1,1) zum Punkt (n,n) beschreiben.

b) Geben Sie eine Grammatik an, die für alle n alle diese Wörter erzeugt.

Aufgabe 14: *Eindeutige kontextfreie Grammatiken?*
Betrachten Sie folgende Grammatiken $G_i = (V, \Sigma, P_i, S)$ mit $V = \{S, A, B\}$ und $\Sigma = \{0, 1, 2, 3\}$, welche sind eindeutig? Begründen Sie Ihr "ja" oder geben Sie ein mehrdeutiges Wort an.
$P_1 = \{ S \to 0, S \to A, A \to 0 \}$
$P_2 = \{ S \to 1, S \to S \}$
$P_3 = \{ S \to 0SS, S \to 1 \}$
$P_4 = \{ S \to S1, S \to 0, A \to AA, A \to 1, A \to AS \}$
$P_5 = \{ S \to 00B01S11, S \to 00B01S10S11, S \to 2, B \to 3 \}$
$P_6 = \{ S \to A0S, S \to A, A \to B1A, A \to B, B \to 2S2, B \to 3 \}$
$P_7 = \{ S \to AB, A \to 0A1, A \to \varepsilon, B \to 0B, B \to \varepsilon \}$
$P_8 = \{ S \to A1B, S \to B1A, A \to 0A0, A \to 1, B \to 0B, B \to 0 \}$
$P_9 = \{ S \to AS, A \to 1B, B \to 0S, B \to 1AS, S \to 0B, S \to 2 \}$
Anmerkung: P_5 hat etwas mit der ein- und zweiseitigen Fallunterscheidung, P_6 etwas mit arithmetischen Ausdrücken zu tun.

Aufgabe 15: *Eindeutige Grammatiken?*
Geben Sie zu folgenden kontextfreien Grammatiken
$G_i = (V, \Sigma, P_i, S)$ mit $V = \{S, A, B\}$ und $\Sigma = \{0, 1, 2, 3\}$
möglichst einfache Syntaxdiagramme an:
$P_1 = \{ S \to 0SS, S \to 1 \}$
$P_2 = \{ S \to AA, S \to 0, A \to AA, A \to 1, A \to AS \}$
$P_3 = \{ S \to 0B1S3, S \to 0B1S2S3, S \to 00, B \to 11 \}$
$P_4 = \{ S \to A0S, S \to A, A \to B1A, A \to B, B \to 2S2, B \to 3 \}$
$P_5 = \{ S \to ABS, A \to 0A1, A \to \varepsilon, B \to 0B, B \to \varepsilon, S \to 2 \}$
$P_6 = \{ S \to A1B, S \to B1A, A \to 0A0, A \to 1, B \to 0B, B \to 0 \}$

Aufgabe 2^4:
Definieren Sie die Syntax der BNF mit Hilfe der BNF.