

## Basisinformationen zu AMPL

AMPL (A Mathematical Programming Language) ist eine algebraische Modellierungssprache für mathematische Optimierungsprobleme. AMPL wurde an den Bell Laboratories entwickelt. AMPL bietet die Möglichkeit, Modelle für mathematische Optimierungsprobleme in abstrakter Form aufzustellen sowie die resultierenden Optimierungsprobleme durch Aufruf externer Solver einer Lösung zuzuführen. Eine weitere Stärke von AMPL ist die Trennung von Modell- und Datenteil.

Detaillierte, regelmässig auf den neuesten Stand gebrachte Informationen zu AMPL sind im WWW unter <http://www.ampl.com> verfügbar. Als Referenz für den Sprachumfang von AMPL bietet sich das folgende Buch an:

R. Fourer, D.M. Gay und B.W. Kernighan, AMPL: A Modelling Language for Mathematical Programming, 1993, Boyd & Fraser Publishing Company, Massachusetts. (Die neuen Auflagen ist bei Duxbury Press erhältlich. Neue ISBN lautet 0-534-38809-4. Die neueste erhältliche Auflage ist die zweite von Duxbury herausgegebene.)

Einige Kurzbeschreibungen zum Sprachumfang sind frei im WWW erhältlich (für Links siehe die WWW-Seite von AMPL.)

AMPL wird in verschiedenen Versionen von verschiedenen Anbietern zum Kauf angeboten, die sich im Leistungsumfang als auch im Umfang der inkludierten Solver unterscheiden.

Die jeweils neueste Version der Student Edition von AMPL ist frei erhältlich und kann aus dem WWW ohne Einschränkungen heruntergeladen werden (Adresse <http://www.ampl.com>). Es gibt Versionen sowohl für Windows als auch Linux/Unix Systeme. Beachten Sie, daß es im Moment jedoch nur eine Testversion einer Windows Version mit integriertem Benutzerinterface gibt. (Am besten die bewährte Kommandozeilenversion verwenden.) Lesen Sie auch die entsprechenden Informationsfiles und beachten Sie, daß Sie neben AMPL auch mindestens einen Solver herunterladen und installieren müssen (z.B. CPLEX für lineare und ganzzahlige lineare Programme und MINOS für lineare und nichtlineare Programme). (Auch diese Solver sind auf der AMPL Seite verfügbar — natürlich nur in eingeschränkter Studentenversion). Die Studentenversion von AMPL unterscheidet sich von der Vollversion nur durch die Tatsache, daß nur Probleme mit maximal 300 Variablen und maximal 300 Nebenbedingungen gelöst werden können. (Dies liegt daran, daß die Versionen der Solver, die in den Studentenversionen inkludiert sind, auf diese Problemgröße eingeschränkt wurden.)

Das Hauptanwendungsgebiet von AMPL ist zwar die Modellierung komplexerer Optimierungsprobleme, aber AMPL kann auch als Schnittstelle zu Solvern verwendet werden, um vorliegende Optimierungsprobleme zu lösen. Das folgende Beispiel soll dies an Hand eines kleinen linearen Programms illustrieren.

### Beispiel 1: (Direkte Verwendung, eigentlich untypisch für AMPL)

Kommandofolge (interaktiv)

```
var XB;
var XC;
maximize Profit: 25*XB + 30*XC;
subject to Zeit: (1/200)*XB + (1/140)*XC <= 40;
subject to BLimit: 0 <= XB <= 6000;
subject to CLimit: 0 <= XC <= 4000;

solve;
display XB,XC;
quit;
```

## Beispiel 2: Eine typischere Anwendung von AMPL

Interaktive Befehlsabfolge:

```
model;
param n; # gegebene Daten
set Perioden := 1..n;
set Produkte;
set Maschinen;
param Bedarf {Produkte} >= 0;
param maxBedarf {p in Produkte} >= Bedarf[p];
param Erloes {Produkte};
param Herstellung {Produkte, Maschinen} >= 0;
param Kapazitaet {Maschinen} >= 0, integer;
# gesuchte Variablen:
var Produktion {Produkte} >= 0, integer;


---


maximize Gewinn: sum {p in Produkte}
    Produktion[p] * Erloes[p] # Zielfunktion
subject to Kap {m in Maschinen}: # Nebenbedingungen
    sum {p in Produkte} Produktion[p] * Herstellung[p,m]
        <= Kapazitaet[m];
Bedarfsdeckung {p in Produkte}: Bedarf[p] <= Produktion[p] <= maxBedarf[p];


---


```

*Daten:*

```
data;
param n := 12;
set Produkte := Zimtsterne Vanillekipferln;
set Maschinen := Mixer Herd;
param:
    Bedarf  maxBedarf :=
    Zimtsterne  30.5  200.9
    Vanillekipferln  79  82 ;
param Kapazitaet :=
    Mixer  10
    Herd  20;
param Herstellung: # zweidimensionale Tabellen
    Herd  Mixer :=
    Zimtsterne  12.3  4
    Vanillekipferln  9.5  5 ;
oder (transponierte Reihenfolge von Zeilen und Spalten:
param Herstellung (tr):
    Zimtsterne Vanillekipferln :=
    Herd  12.3  9.5
    Mixer  4  5 ;
oder:
param Herstellung :=
    Zimtsterne Herd  12.3
    Zimtsterne Mixer  4
    Vanillekipferln Herd  9.5
    Vanillekipferln Mixer  5 ;


---


```

**Einige weitere nützliche Befehle:**

```
solve;
display Gewinn, {p in Produkte} Produktion[p];
include Dateiname; model Datei.mod; data Datei.dat; commands Datei.run;
let Kapazitaet["Herd"] := 30;
reset; reset model; reset data;
drop Bedingung; restore Bedingung;
fix Variable; unfix Variable;
```