V.M. GLUSHKOV INSTITUTE
OF CYBERNETICS OF THE NAS OF UKRAINE

NEOS: SERVER AND SOLVERS

LECTURE 3/SURVEY OF OPTIMIZATION

Course developer: **B**utenko S. (Prof., Texas A&M University, PhD)
Course advisor: **S**tetsyuk P.I. (Doctor of Sciences, Institute of Cybernetics)

Department of Post-Graduate Studies

Course Instructor : Volovyk E. I.

AGENDA

- NEOS Server: overview, structure, general characteristics, etc.)
- GUROBI optimizer
- MINOS optimizer
- BARON optimizer
- CPLEX optimizer
- GAMS (General Algebraic Modeling System)
- AMPL (A Mathematical Programming Language)
- Optimization Problem Examples using AMPL with different optimizers

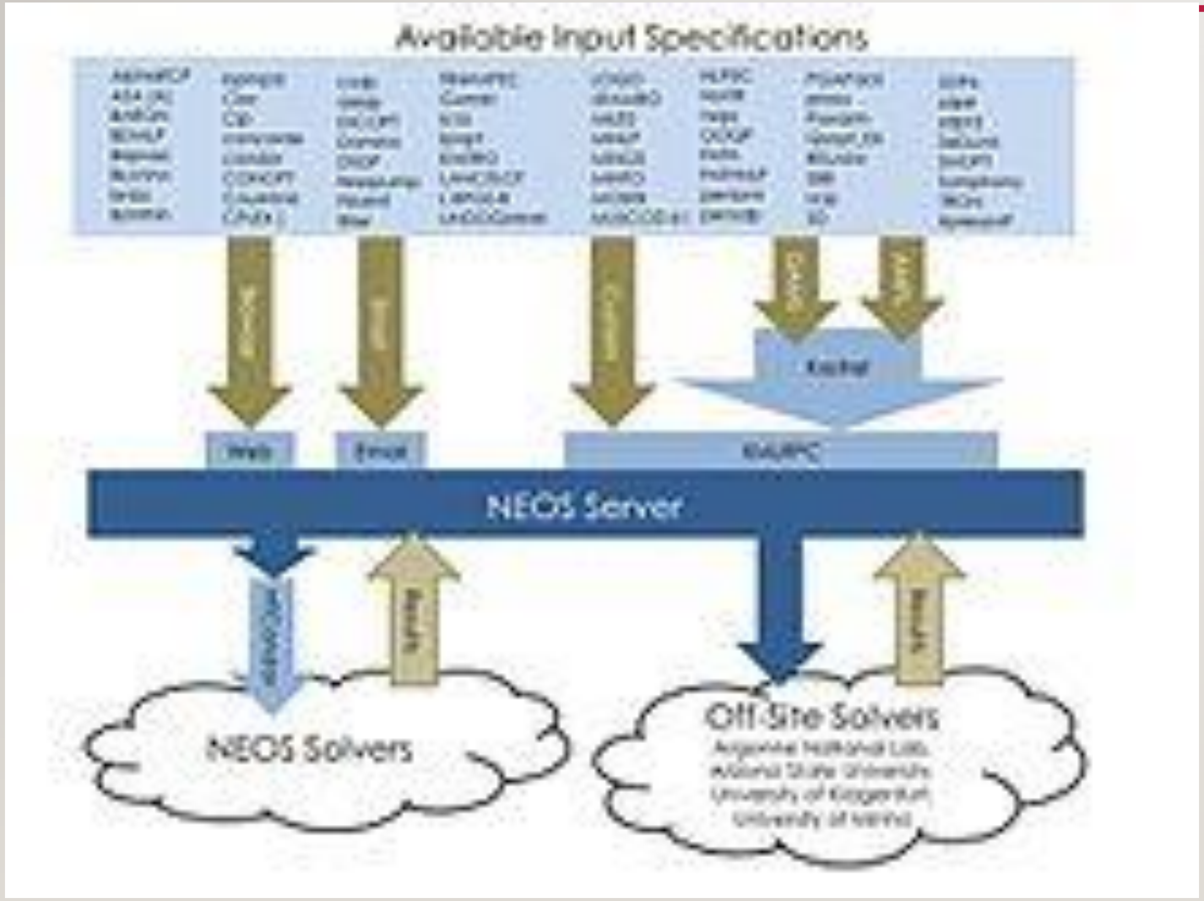
NEOS SERVER

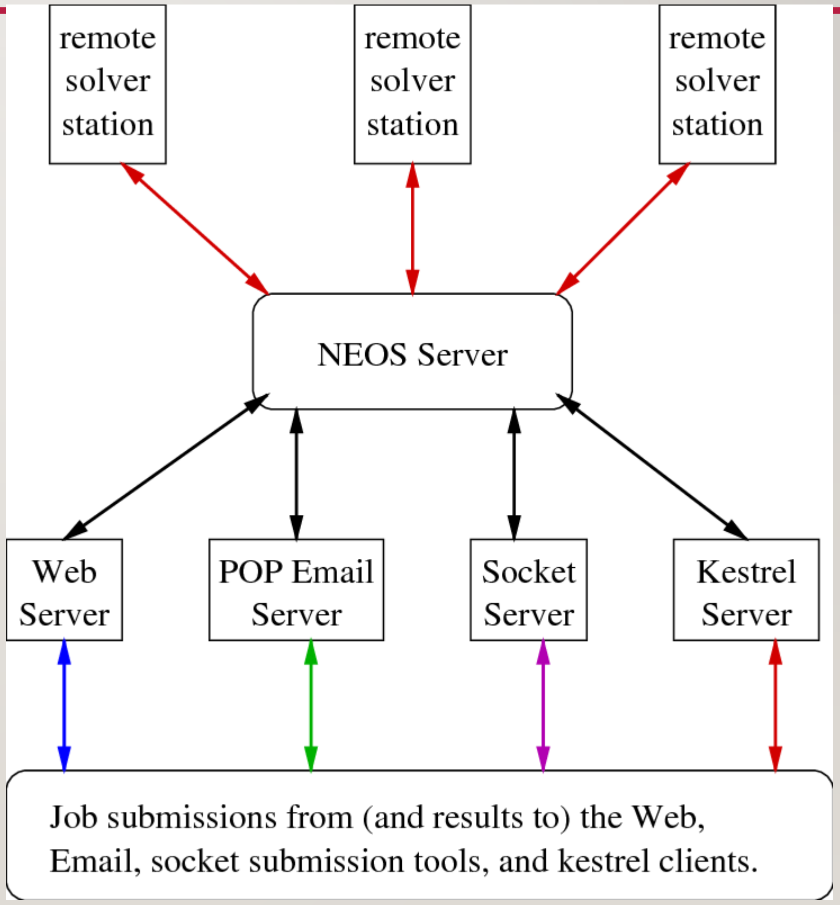
The NEOS (Network-Enabled Optimization System) Server is an Internet-based client-server application that provides free access to a library of optimization solvers. Its library of solvers includes more than 60 commercial, free and open source solvers, which can be applied to mathematical optimization problems of more than 12 different types, including linear programming, integer programming and nonlinear optimization.

NEOS SERVER (https://neos-server.org/neos/)

- Developed in 1996 by the Optimization Technology Center of Argonne National Laboratory(IL) and Northwestern University (IL);
- Objective: to develop a method to share optimization software resources over the Internet;
- Managed by the Wisconsin Institute for Discovery at the University of Wisconsin-Madison;
- Most of the solvers are hosted by the University of Wisconsin in Madison;
- Remote solvers are hosted by partner organizations: Arizona State University, the University of Klagenfurt in Austria, and the University of Minho in Portugal.

NEOS SERVER STRUCTURE





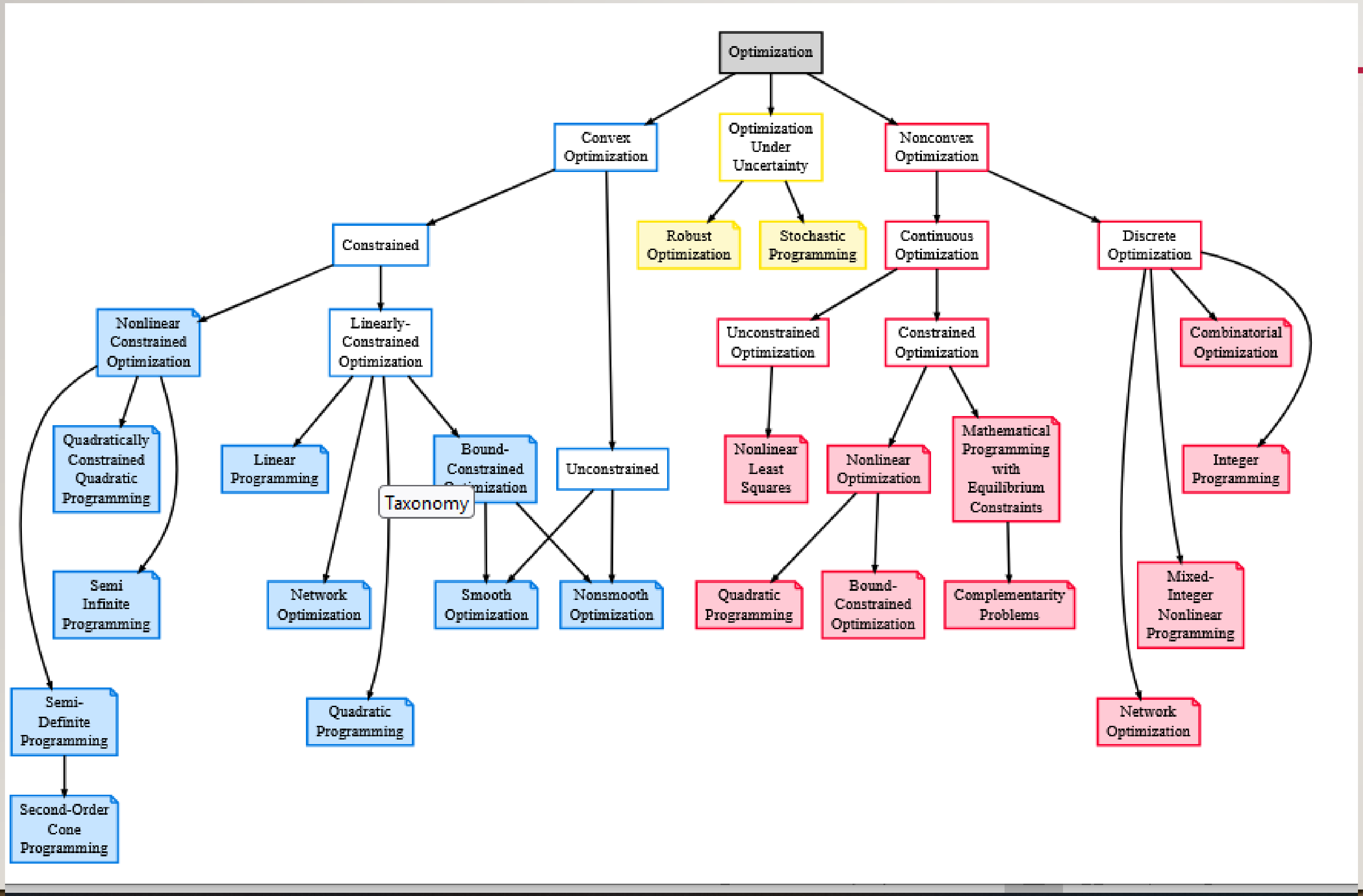
CURRENT NEOS ADMINISTRATORS

(UW-Madison)

- Michael Ferris
- Ben Huebner
- Jeff Linderoth
- Elizabeth Wong

The development of the NEOS Server was partially supported by various grants from the Office of Advanced Scientific Computing of the Department of Energy and the National Science Foundation.

OPTIMIZATION PROBLEM TYPES


Optimization
Convex Optimization
Optimization Under Uncertainty
Nonconvex Optimization
Constrained
Robust Optimization
Stochastic Programming
Continuous Optimization
Discrete Optimization
Nonlinear Constrained Optimization
Linearly-Constrained Optimization
Unconstrained Optimization
Constrained Optimization
Combinatorial Optimization
Quadratically Constrained Quadratic Programming
Linear Programming
Bound-Constrained Optimization
Unconstrained
Nonlinear Least Squares
Nonlinear Optimization
Mathematical Programming with Equilibrium Constraints
Integer Programming
Taxonomy
Semi Infinite Programming
Network Optimization
Smooth Optimization
Nonsmooth Optimization
Quadratic Programming
Bound-Constrained Optimization
Complementarity Problems
Mixed-Integer Nonlinear Programming
Semi-Definite Programming
Quadratic Programming
Network Optimization
Second-Order Cone Programming

- Continuous Optimization
- Discrete Optimization
- Unconstrained Optimization
- Constrained Optimization
- Optimization Under Uncertainty

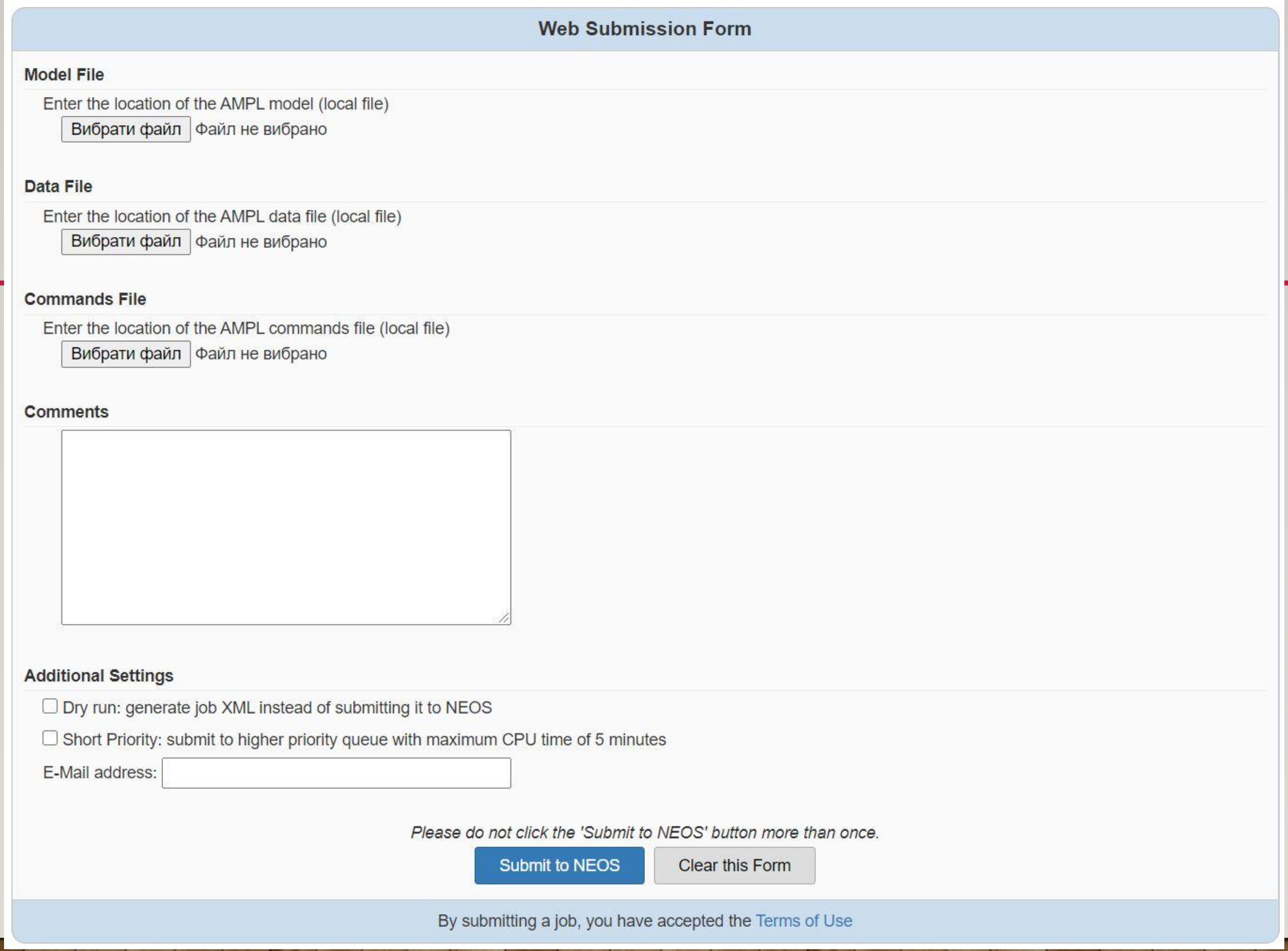
NEOS AVAILABLE SOLVERS

Several popular commercial solvers are available free-of-charge through the NEOS Server:

- AMPL
- Artelys Knitro for Knitro
- COIN-OR
- COPT
- FICO for the Xpress Optimization Suite

- GAMS
- Gurobi
- IBM for CPLEX
- Lindo Systems, Inc. for LINDOGlobal

- Mosek
- Octeract
- OpenSolver



Web Submission Form

Model File

Enter the location of the AMPL model (local file)

Вибрати файл Файл не вибрано

Data File

Enter the location of the AMPL data file (local file)

Вибрати файл Файл не вибрано

Commands File

Enter the location of the AMPL commands file (local file)

Вибрати файл Файл не вибрано

Comments

Additional Settings

☐ Dry run: generate job XML instead of submitting it to NEOS

☐ Short Priority: submit to higher priority queue with maximum CPU time of 5 minutes

E-Mail address:

Please do not click the 'Submit to NEOS' button more than once.

Submit to NEOS Clear this Form

By submitting a job, you have accepted the Terms of Use

GUROBI OPTIMIZER

GUROBI
OPTIMIZATION

- It is a special kind of software called a "solver."
- Bob Bixby, Zonghao Gu and Ed Rothberg founded Gurobi in 2008.
- Supports modeling languages: AMPL, GAMS, MPL.
- Supports interface of the programming languages: C , C++, Java, Python, MATLAB, etc.
- Large linear and quadratic programming models, mixed integer programming, nonlinear programming,
- Uses a unique combination of Genetic Algorithms and classical non-linear optimization methods

MINOS OPTIMIZER

- MINOS (Modular In-core Nonlinear Optimization System);
- a Fortran software package for solving linear and nonlinear mathematical optimization problems;
- developed by Bruce Murtagh and Michael Saunders at Stanford University;
- remains one of the top-used solvers on the NEOS Server and in GAMS.
- **Problem types supported**: Linear, quadratic, and smooth nonlinear objectives and constraints in continuous variables.
- **Special features**: Linear constraints are handled separately from nonlinear ones, for greater efficiency.

BARON OPTIMIZER

- Branch-and-Reduce Optimization Navigator from the Optimization Firm, (USA);
- the product of nearly 30 years of academic research recognized by the INFORMS Computing Society Prize and the Beale-Orchard-Hays Prize for excellence in computational mathematical programming;
- **Problem types supported**: Linear Programming, Non-Linear Programming, mixed-integer programming (MIP), and Mixed-Integer Nonlinear Programming (MINLP) problems.
- Supports modeling languages: AMPL, GAMS, AIMMS.

CPLEX OPTIMIZER

- CPLEX was the first commercial linear optimizer on the market to be written in the C programming language;
- The name CPLEX itself comes from C-Simplex resulted in CPLEX.
- High-performance optimization solver for linear, mixed-integer and quadratic programming;
- **Robert E. Bixby** originally developed CPLEX Optimization in 1987;
- **Problem types supported**: Linear Programming, Non-Linear Programming, Integer Programming, Mixed-Integer programming (MIP), and Quadratic Programming problems.
- Supports interface of the programming languages: C , C++, Java, Python, MATLAB, Excel,
- Supports modeling languages: AMPL, GAMS, AIMMS, OptimJ, TOMLAB.

GAMS (GENERAL ALGEBRAIC MODELING SYSTEM)

- started as a Research Project at the World Bank in 1976;
- 1987 GAMS becomes a commercial product;
- the first algebraic modeling language; similar to commonly used fourth-generation programming languages;
- a high-level modeling system for mathematical optimization;
- designed for modeling and solving linear, nonlinear, and mixed-integer optimization problem;
- allows the users to implement a sort of hybrid algorithm combining different solvers;
- among the most popular input formats for the NEOS Server.

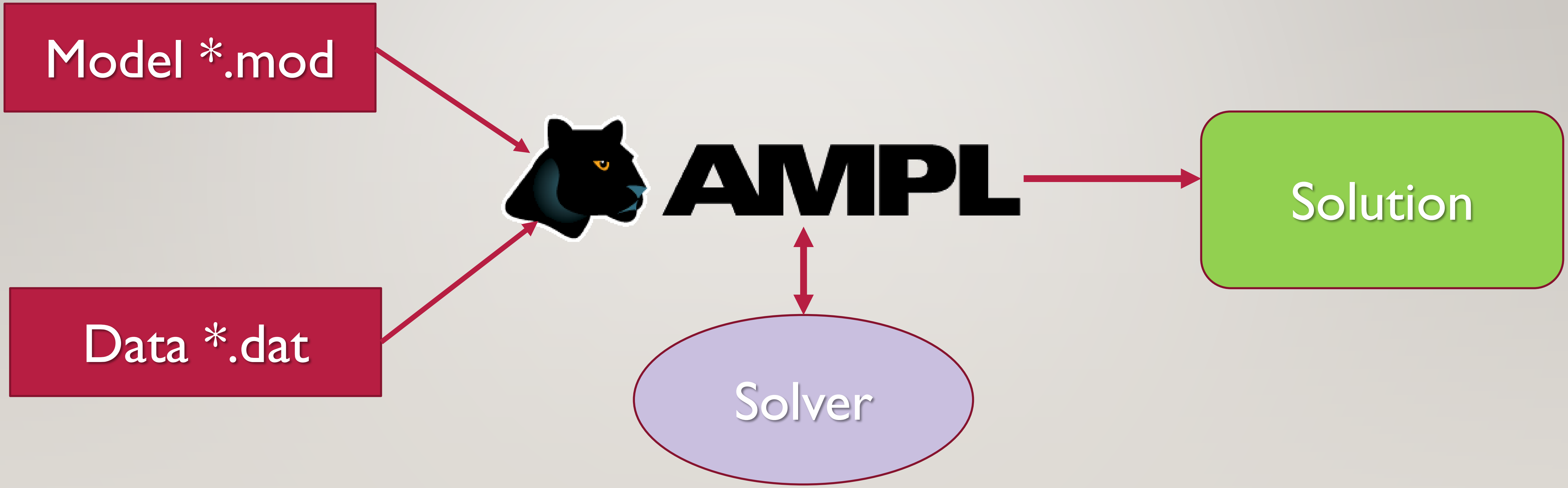
AMPL (A MATHEMATICAL PROGRAMMING LANGUAGE)

- developed by Robert Fourer, David Gay, and Brian Kernighan at Bell Laboratories;
- 1985 - AMPL was designed and implemented;
- supports dozens of solvers, both open source and commercial software;
- its syntax similar to the mathematical notation of optimization problems;
- available for many popular 32- and 64-bit operating systems including Linux, macOS, Solaris, AIX, and Windows;
- supports a wide range of problem types: Linear programming, Quadratic programming, Nonlinear programming, Mixed-integer programming, Global optimization, etc.

AMPL REFERENCES

- https://ampl.com/ provides free downloadable “student” versions of AMPL and representative solvers that run on Windows, Unix/Linux, and Mac OS X.
- https://ampl.com/resources/the-ampl-book/chapter-downloads/ - book «*AMPL: A Modeling Language for Mathematical Programming*» (R. Fourer, D. Gay, B. Kernighan)
- https://ampl.com/try-ampl/download-a-free-demo/ – AMPL IDE

HOW AMPL WORKS



INSTALLING AND RUNNING AMPL

- Create a folder you will use to store all your AMPL-related files.
- "Unzip" the distribution file by double-clicking on it.
- Move all the files from the resulting folder into your AMPL folder.
- To run immediately, double-click "ampl" executable.

AMPL FILE TYPES AND BASIC COMMANDS

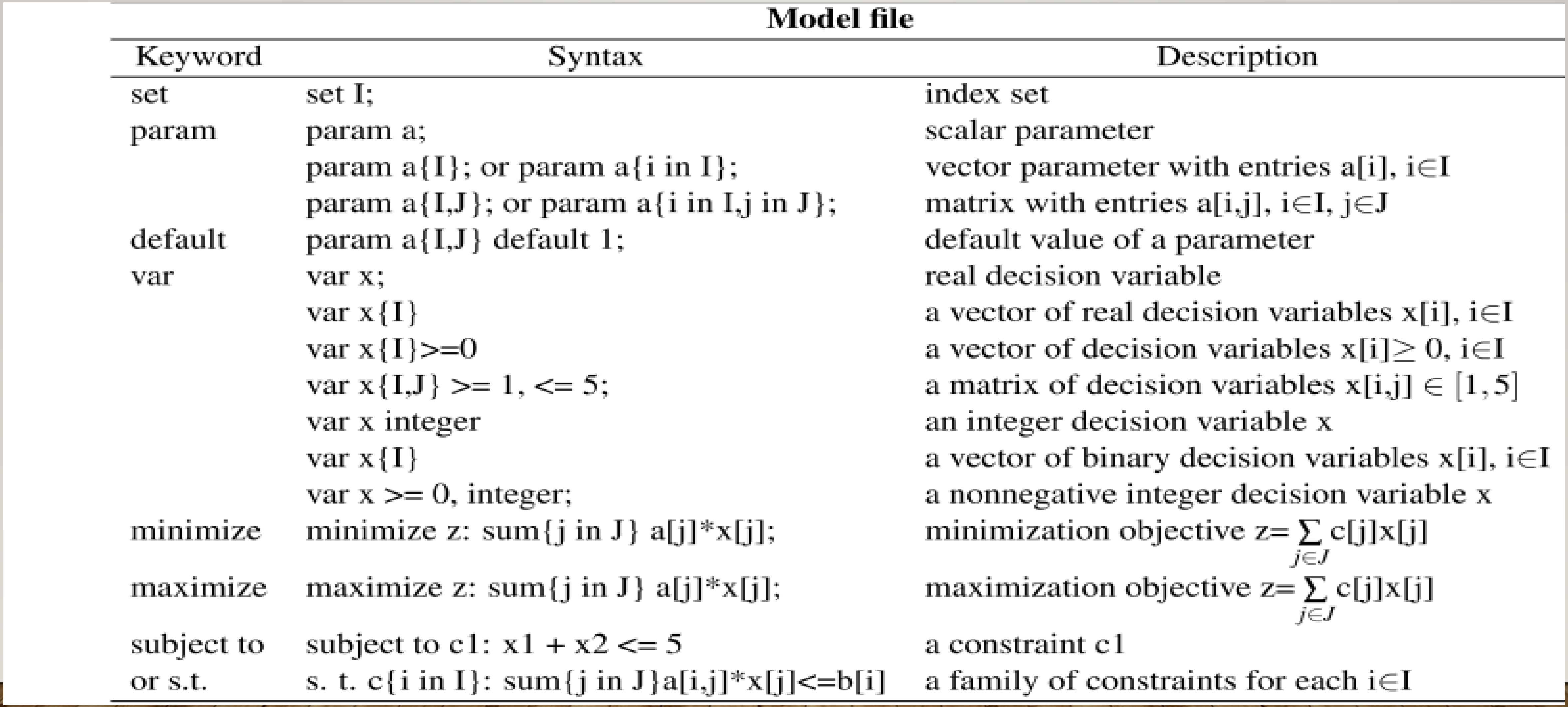
- **.mod** (model) file contains the mathematical formulation of the problem in general form, with sets and parameters used to represent the inputs;
- **.dat** (data) file contains the description of the sets and values of the parameters describing the instance of the problem being solved;
- **.run** file lists the instructions for AMPL to load the model and data files, interact with optimization solvers, and output the solution and related information.

The files can be created and modified using any text editor.

AMPL BASICS

- AMPL is case-sensitive;
- AMPL ignores whitespace;
- Your model will go into a text-file (.mod or/and .dat);
- You will type commands like *solve* at the *ampl:* prompt ;
- Comments are denoted by a number sign (#);
- Every line must end with a semicolon (;);

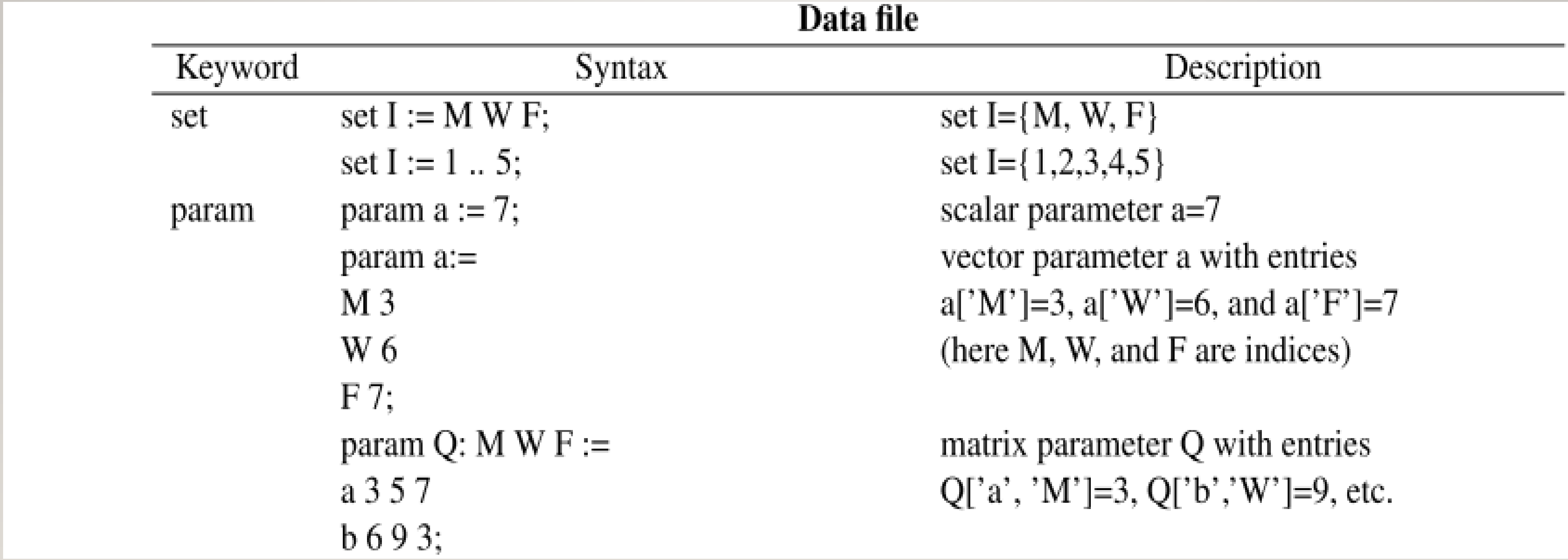
BASICS OF AMPL SYNTAX (MODEL FILE)



Model file

Keyword	Syntax	Description
set	set I;	index set
param	param a;	scalar parameter
	param a{I}; or param a{i in I};	vector parameter with entries a[i], i∈I
	param a{I,J}; or param a{i in I,j in J};	matrix with entries a[i,j], i∈I, j∈J
default	param a{I,J} default 1;	default value of a parameter
var	var x;	real decision variable
	var x{I}	a vector of real decision variables x[i], i∈I
	var x{I}>=0	a vector of decision variables x[i]≥ 0, i∈I
	var x{I,J} >= 1, <= 5;	a matrix of decision variables x[i,j] ∈ [1, 5]
	var x integer	an integer decision variable x
	var x{I}	a vector of binary decision variables x[i], i∈I
	var x >= 0, integer;	a nonnegative integer decision variable x
minimize	minimize z: sum{j in J} a[j]*x[j];	minimization objective $z=\sum_{j\in J} c[j]x[j]$
maximize	maximize z: sum{j in J} a[j]*x[j];	maximization objective $z=\sum_{j\in J} c[j]x[j]$
subject to	subject to c1: x1 + x2 <= 5	a constraint c1
or s.t.	s. t. c{i in I}: sum{j in J}a[i,j]*x[j]<=b[i]	a family of constraints for each i∈I

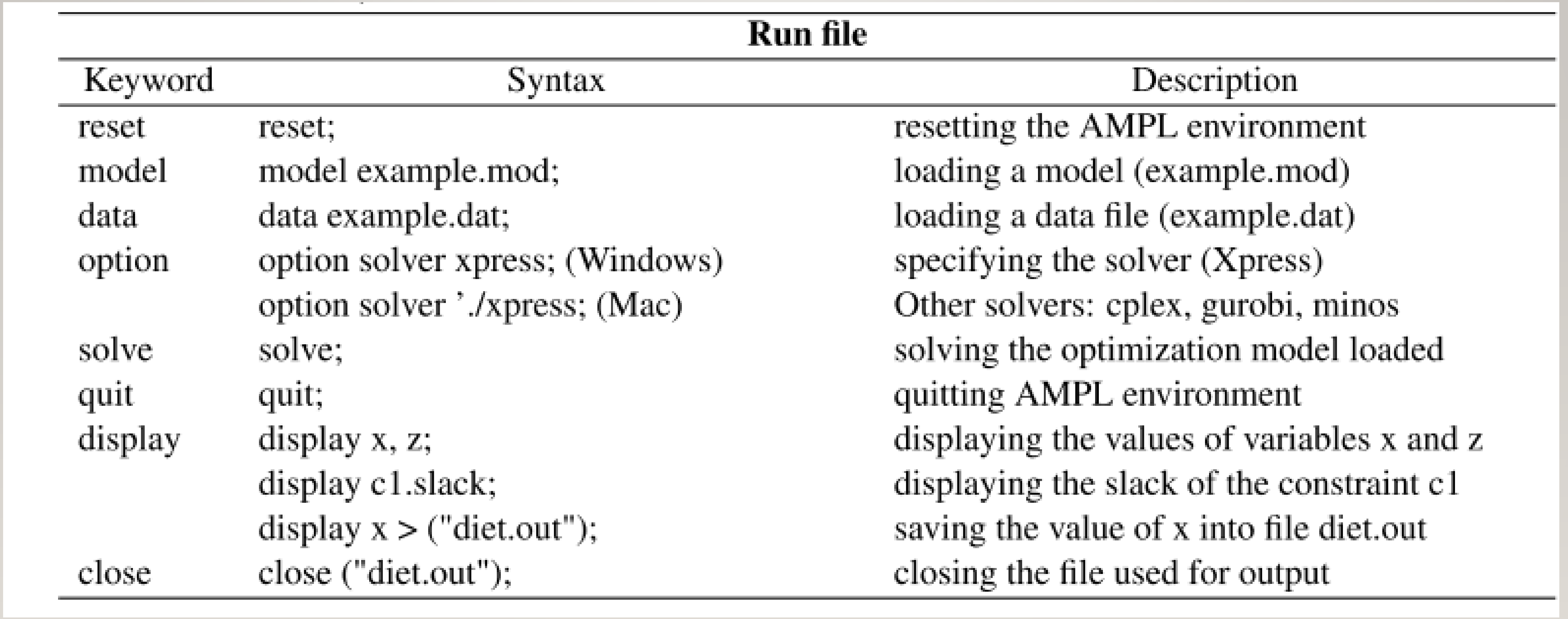
BASICS OF AMPL SYNTAX (DATA FILE)



Data file

Keyword	Syntax	Description
set	set I := M W F;	set I={M, W, F}
	set I := 1 .. 5;	set I={1,2,3,4,5}
param	param a := 7;	scalar parameter a=7
	param a:=	vector parameter a with entries
	M 3	a['M']=3, a['W']=6, and a['F']=7
	W 6	(here M, W, and F are indices)
	F 7;	
	param Q: M W F :=	matrix parameter Q with entries
	a 3 5 7	Q['a', 'M']=3, Q['b','W']=9, etc.
	b 6 9 3;	

BASICS OF AMPL SYNTAX (RUN FILE)



Run file

Keyword	Syntax	Description
reset	reset;	resetting the AMPL environment
model	model example.mod;	loading a model (example.mod)
data	data example.dat;	loading a data file (example.dat)
option	option solver xpress; (Windows)	specifying the solver (Xpress)
	option solver ’./xpress; (Mac)	Other solvers: cplex, gurobi, minos
solve	solve;	solving the optimization model loaded
quit	quit;	quitting AMPL environment
display	display x, z;	displaying the values of variables x and z
	display c1.slack;	displaying the slack of the constraint c1
	display x > ("diet.out");	saving the value of x into file diet.out
close	close ("diet.out");	closing the file used for output

BASIC OF AMPL SYNTAX (MODEL FILE)

- Keywords: **set**, **node**, **arc**, **var**, **param**, **minimize**, **maximize**, **subject to (s.t.)**
- Reserved keywords: **for**, **if**, **elseif**, **else**, **while**, **file**, **system**
- Arithmetic Operations : **+ - * / ^**
- Functions : **abs()**, **sin()**, **cos()**, **log()**, **sqrt()**, **exp()**
- Commands: **display**, **write**, **print**, **option**, **include**, **quit , etc.**
- Comments: **#** and **/* */**
- Letters: ‘a’, “ABC”

EXAMPLE 1 : REVENUE MAXIMIZATION PROBLEM

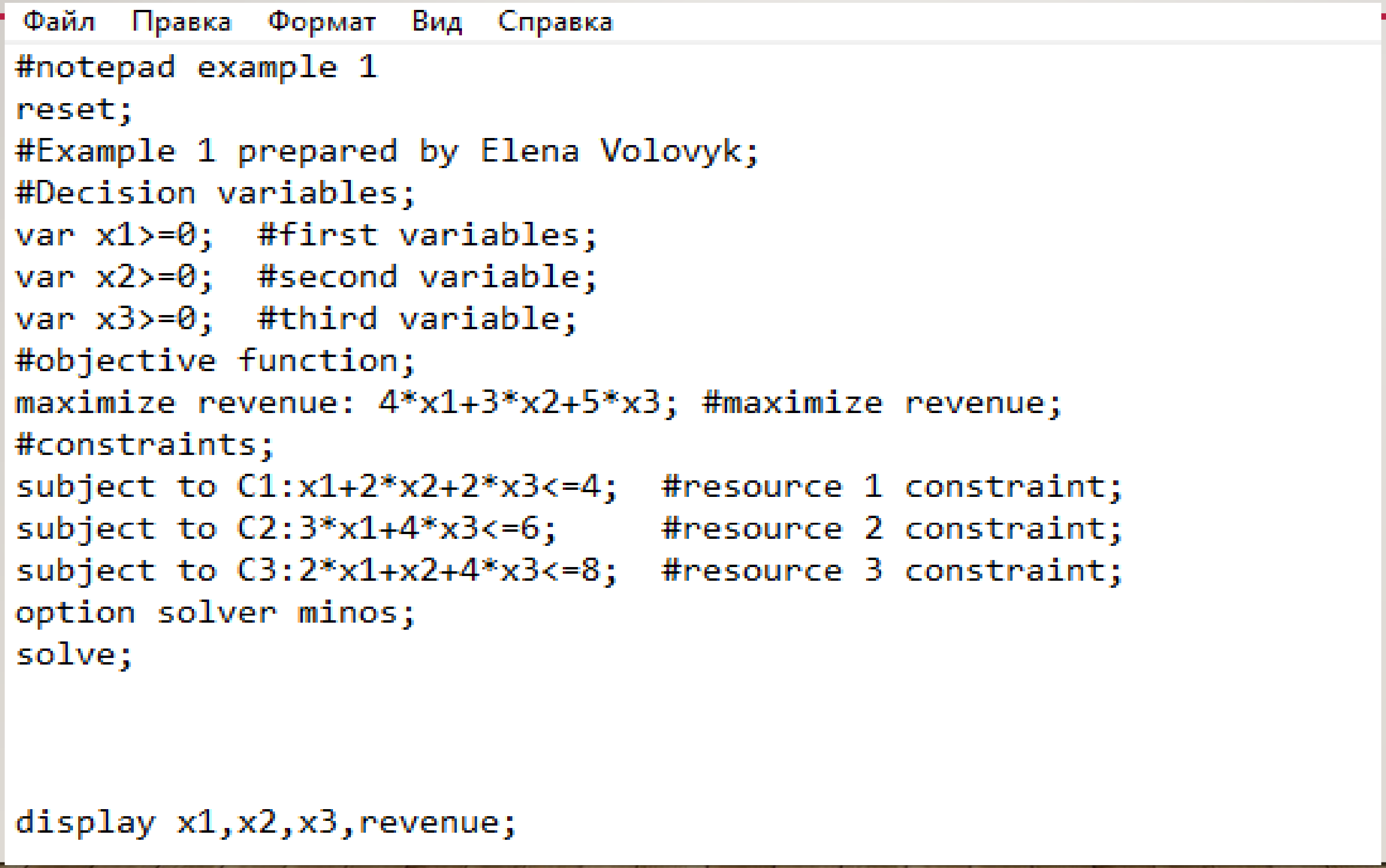
Suppose a manufacturing company makes 3 products using 3 resources. The table below provides the information on the number of units of each resource used to make one unit of each product, the selling price of one unit of each product (last row), and the available supply of each resource, expressed in the number of units (last column). The objective of the company is to maximize the revenue by producing an optimal number of units of each product, while not exceeding the resource limitations.

	Product 1	Product 2	Product 3	Supply
Resource 1	1	2	2	4
Resource 2	3	0	4	6
Resource 3	2	1	4	8
Price:	4	3	5	

COMPLETE LINEAR PROGRAMMING FORMULATION

Maximize	$4x_1 + 3x_2 + 5x_3$	(revenue)
Subject to (s.t.)	$x_1 + 2x_2 + 2x_3 \leq 4$	(resource 1 constraint)
	$3x_1 + \quad 4x_3 \leq 6$	(resource 2 constraint)
	$2x_1 + 4x_2 + 3x_3 \leq 8$	(resource 3 constraint)
	$x_1, x_2, x_3 \geq 0$	(nonnegativity constraints)

AMPL CODE FOR EXAMPLE 1

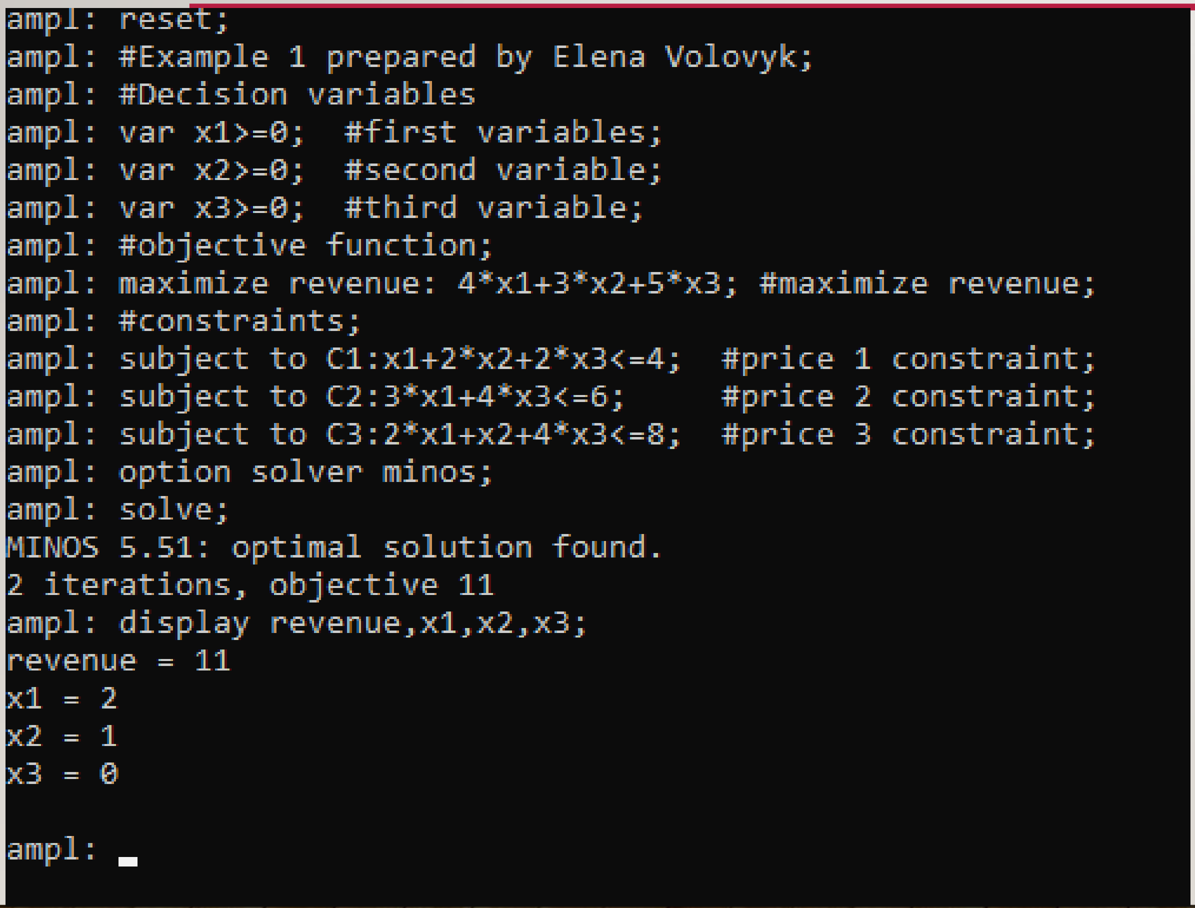


```
#notepad example 1
reset;
#Example 1 prepared by Elena Volovyk;
#Decision variables;
var x1>=0;  #first variables;
var x2>=0;  #second variable;
var x3>=0;  #third variable;
#objective function;
maximize revenue: 4*x1+3*x2+5*x3; #maximize revenue;
#constraints;
subject to C1:x1+2*x2+2*x3<=4;  #resource 1 constraint;
subject to C2:3*x1+4*x3<=6;     #resource 2 constraint;
subject to C3:2*x1+x2+4*x3<=8;  #resource 3 constraint;
option solver minos;
solve;



display x1,x2,x3,revenue;
```

AMPL SOLVER'S RESULTS FOR EXAMPLE 1 (MINOS)



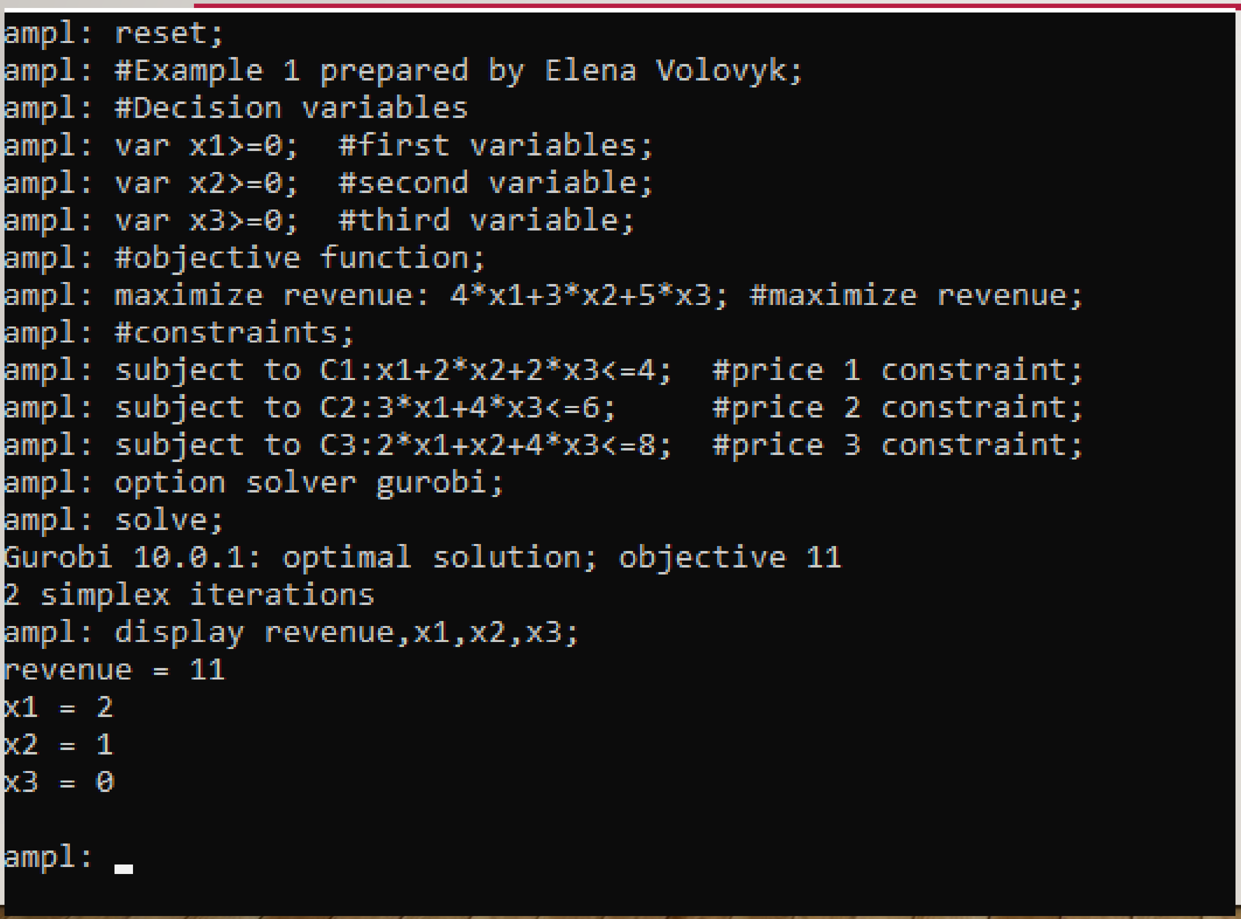
```
ampl: reset;
ampl: #Example 1 prepared by Elena Volovyk;
ampl: #Decision variables
ampl: var x1>=0;  #first variables;
ampl: var x2>=0;  #second variable;
ampl: var x3>=0;  #third variable;
ampl: #objective function;
ampl: maximize revenue: 4*x1+3*x2+5*x3; #maximize revenue;
ampl: #constraints;
ampl: subject to C1:x1+2*x2+2*x3<=4;  #price 1 constraint;
ampl: subject to C2:3*x1+4*x3<=6;     #price 2 constraint;
ampl: subject to C3:2*x1+x2+4*x3<=8;  #price 3 constraint;
ampl: option solver minos;
ampl: solve;
MINOS 5.51: optimal solution found.
2 iterations, objective 11
ampl: display revenue,x1,x2,x3;
revenue = 11
x1 = 2
x2 = 1
x3 = 0

ampl:
```

CONCLUSION:

With the stated constraints, the company may reach the maximum revenue of $11 by producing the optimal number of product1= 2, product2=1 and product3=0

AMPL SOLVER'S RESULTS FOR EXAMPLE 1 (GUROBI)



```
ampl: reset;
ampl: #Example 1 prepared by Elena Volovyk;
ampl: #Decision variables
ampl: var x1>=0;  #first variables;
ampl: var x2>=0;  #second variable;
ampl: var x3>=0;  #third variable;
ampl: #objective function;
ampl: maximize revenue: 4*x1+3*x2+5*x3; #maximize revenue;
ampl: #constraints;
ampl: subject to C1:x1+2*x2+2*x3<=4;  #price 1 constraint;
ampl: subject to C2:3*x1+4*x3<=6;     #price 2 constraint;
ampl: subject to C3:2*x1+x2+4*x3<=8;  #price 3 constraint;
ampl: option solver gurobi;
ampl: solve;
Gurobi 10.0.1: optimal solution; objective 11
2 simplex iterations
ampl: display revenue,x1,x2,x3;
revenue = 11
x1 = 2
x2 = 1
x3 = 0

ampl: _
```

CONCLUSION:

With the stated constraints, the company may reach the maximum revenue of $11 by producing the optimal number of product1= 2, product2=1 and product3=0

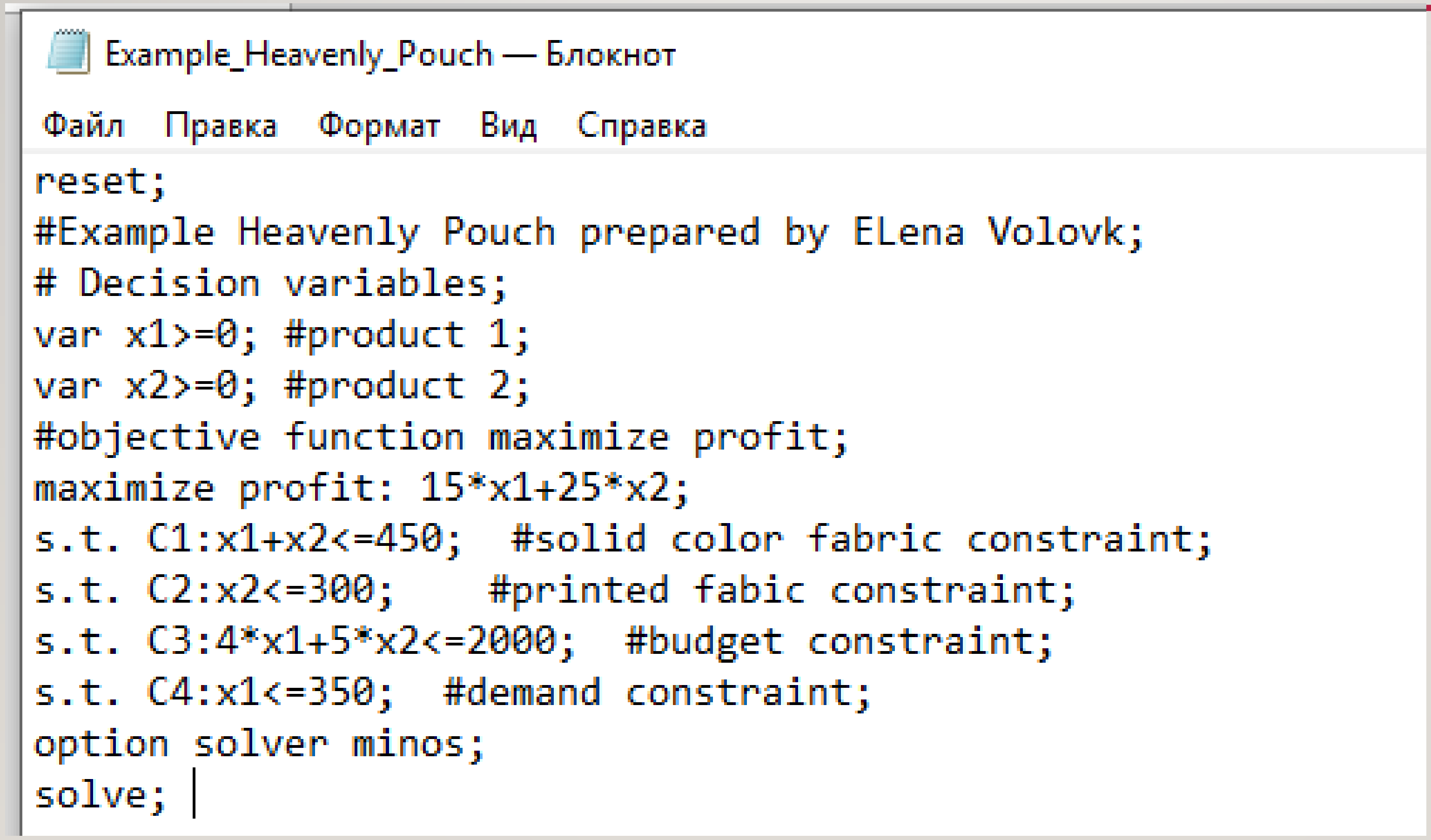
EXAMPLE 2: HEAVENLY POUCH Inc. PROBLEM

Heavenly Pouch, Inc. produces two types of baby carriers, non-reversible and reversible. Each non-reversible carrier sells for $23, requires 2 meters of a solid color fabric, and costs $8 to manufacture. Each reversible carrier sells for $35, requires 2 meters of a printed fabric as well as 2 meters of a solid color fabric, and costs $10 to manufacture. The company has 900 meters of solid color fabrics and 600 meters of printed fabrics available for its new carrier collection. It can spend up to $4,000 on manufacturing the carriers. The demand is such that all reversible carriers made are projected to sell, whereas at most 350 non-reversible carriers can be sold. Heavenly Pouch is interested in formulating a mathematical model that could be used to maximize its profit (e.g., the difference of revenues and expenses) resulting from manufacturing and selling the new carrier collection.

COMPLETE LINEAR PROGRAMMING FORMULATION

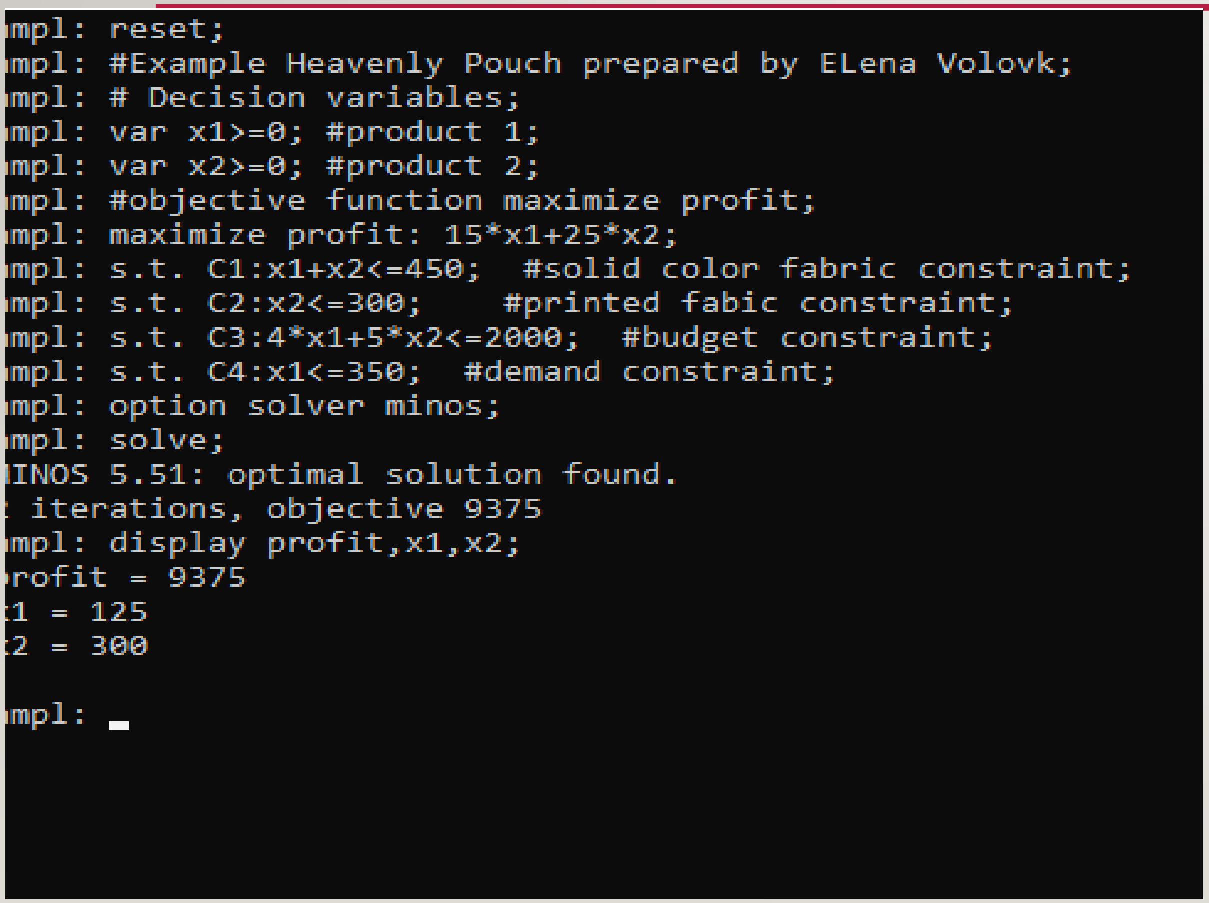
Maximize	$15x_1 + 25x_2$	(profit)
Subject to (s.t.)	$x_1 + x_2 \leq 450$	(solid color fabric constraint)
	$x_2 \leq 300$	(printed fabric constraint)
	$4x_1 + 5x_2 \leq 2000$	(budget constraint)
	$x_1 \leq 350$	(demand constraint)
	$x_1, x_2 \geq 0$	(nonnegativity constraints)

AMPL CODE FOR THE HEAVENLY POUCH PROBLEM


Example_Heavenly_Pouch — Блокнот
Файл Правка Формат Вид Справка

```
reset;
#Example Heavenly Pouch prepared by ELena Volovk;
# Decision variables;
var x1>=0; #product 1;
var x2>=0; #product 2;
#objective function maximize profit;
maximize profit: 15*x1+25*x2;
s.t. C1:x1+x2<=450;  #solid color fabric constraint;
s.t. C2:x2<=300;    #printed fabic constraint;
s.t. C3:4*x1+5*x2<=2000;  #budget constraint;
s.t. C4:x1<=350;  #demand constraint;
option solver minos;
solve;
```

AMPL SOLVER'S RESULTS FOR EXAMPLE 2



```
mpl: reset;
mpl: #Example Heavenly Pouch prepared by ELena Volovk;
mpl: # Decision variables;
mpl: var x1>=0; #product 1;
mpl: var x2>=0; #product 2;
mpl: #objective function maximize profit;
mpl: maximize profit: 15*x1+25*x2;
mpl: s.t. C1:x1+x2<=450;  #solid color fabric constraint;
mpl: s.t. C2:x2<=300;    #printed fabic constraint;
mpl: s.t. C3:4*x1+5*x2<=2000;  #budget constraint;
mpl: s.t. C4:x1<=350;  #demand constraint;
mpl: option solver minos;
mpl: solve;
INOS 5.51: optimal solution found.
 iterations, objective 9375
mpl: display profit,x1,x2;
rofit = 9375
1 = 125
2 = 300

mpl: _
```

CONCLUSION:

With the stated constraints, the company Heavenly Pouch Inc. may increase its profit up to the maximum of $9,375 provided they will produce 125 units of product 1 and 300 units of product 2.

EXAMPLE 3: MAXIMUM BOX VOLUME

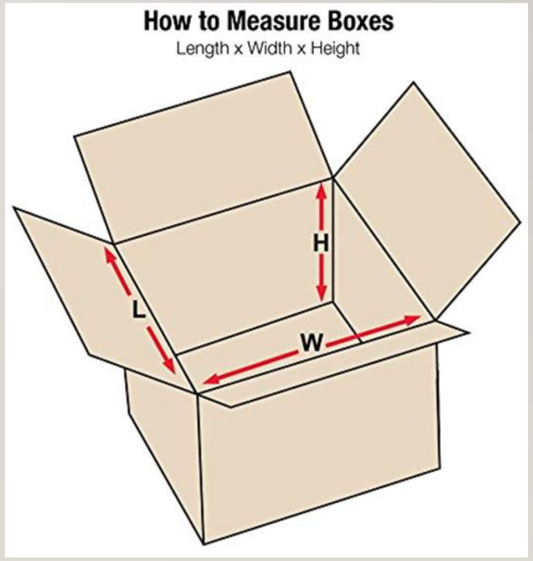
Create a box with an open top and square base with a surface area of 300 square centimeters .What height will produce a box with the maximum possible volume?

MAXIMIZE V= x^2h

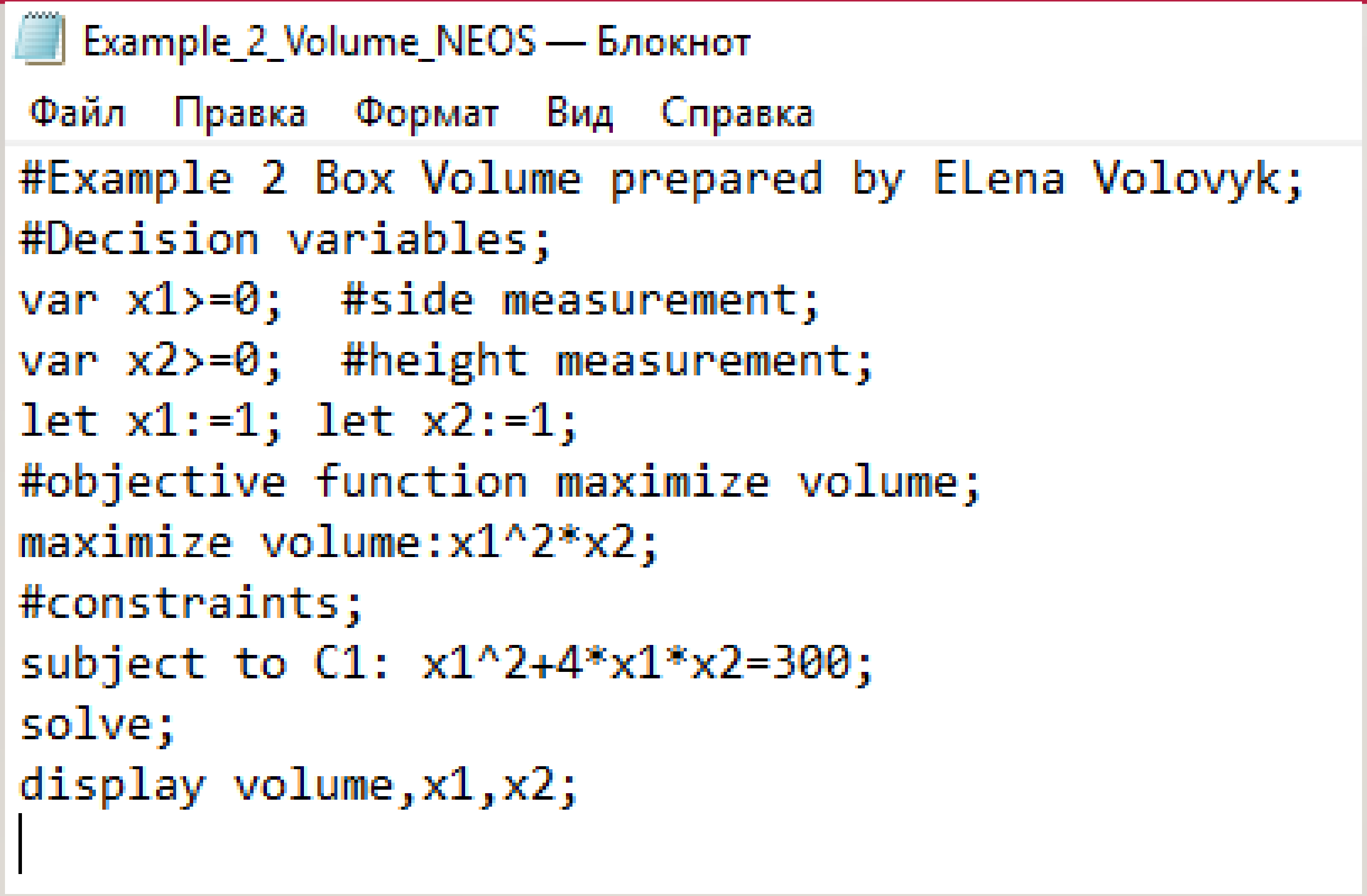
Subject to $x^2 + 4xh = 300$

$x > 0$

$h > 0$

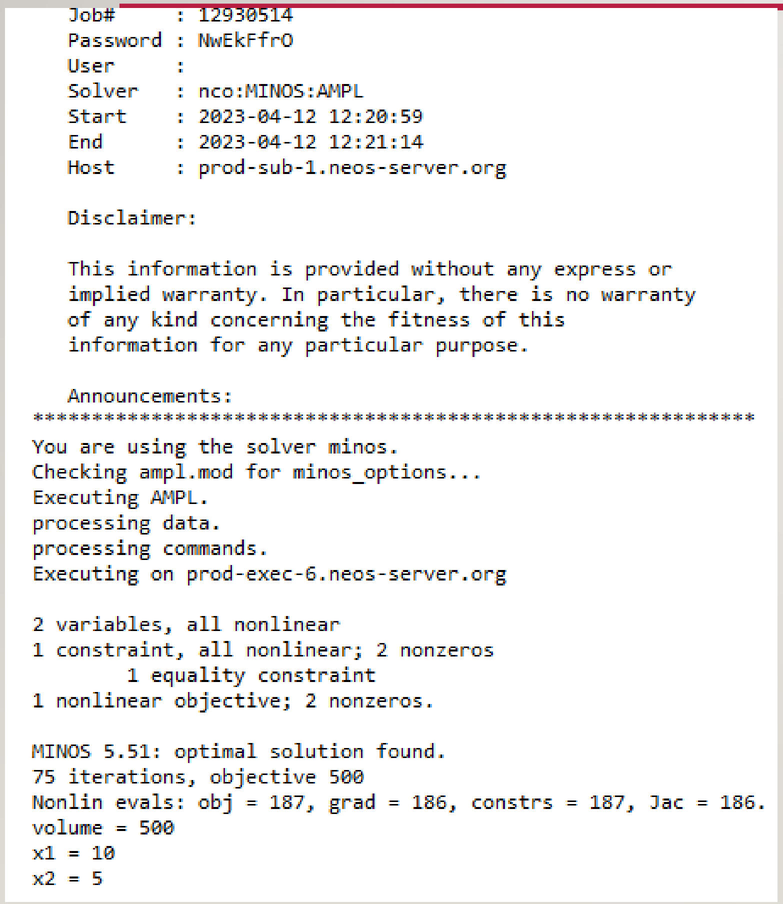


AMPL CODE FOR MAXIMUM BOX VOLUME EXAMPLE


Example_2_Volume_NEOS — Блокнот
Файл Правка Формат Вид Справка

```
#Example 2 Box Volume prepared by ELena Volovyk;
#Decision variables;
var x1>=0;  #side measurement;
var x2>=0;  #height measurement;
let x1:=1; let x2:=1;
#objective function maximize volume;
maximize volume:x1^2*x2;
#constraints;
subject to C1: x1^2+4*x1*x2=300;
solve;
display volume,x1,x2;
```

NEOS RESULTS FOR EXAMPLE 3



```
   Job#     : 12930514
   Password : NwEkFfrO
   User     :
   Solver   : nco:MINOS:AMPL
   Start    : 2023-04-12 12:20:59
   End      : 2023-04-12 12:21:14
   Host     : prod-sub-1.neos-server.org

   Disclaimer:

   This information is provided without any express or
   implied warranty. In particular, there is no warranty
   of any kind concerning the fitness of this
   information for any particular purpose.

   Announcements:
************************************************************
You are using the solver minos.
Checking ampl.mod for minos_options...
Executing AMPL.
processing data.
processing commands.
Executing on prod-exec-6.neos-server.org

2 variables, all nonlinear
1 constraint, all nonlinear; 2 nonzeros
        1 equality constraint
1 nonlinear objective; 2 nonzeros.

MINOS 5.51: optimal solution found.
75 iterations, objective 500
Nonlin evals: obj = 187, grad = 186, constrs = 187, Jac = 186.
volume = 500
x1 = 10
x2 = 5
```

CONCLUSION:

The maximum volume of a box with a squared base, provided that the surface areas is 300 square centimeters is 500 cube centimeters with the side measurement of 10 centimeters and the height measurement 5 centimeters.

Thank you!