Instructions:
In many real life applications, we often need to solve one single problem again and again with different inputs. For instance, in the data-fitting problem in Lab 02, the costs or number of warehouses may change as more data becomes available. So will have to rewrite the AMPL model again every time data changes. In order to avoid rewriting models and to reduce errors introduced by manually changing the model description, we can separate the data from the model. Then the same model can be used again and again just by changing the data. AMPL (and also other languages) have provisions for separating data from the model. It is an important concept in modeling.

Another benefit of separating the model and data is that we can write small models using loops and sets in compact notation. It enables us to solve problems with many variables and constraints without having to write each and every constraint explicitly. Structured problems with thousands of constraints may be written in only a few lines.

In AMPL one can write an abstract model as a '.mod' file while keeping the data separately in a '.dat' file. A typical AMPL session will then look like:

```
ampl: model ex1.mod;
ampl: data ex1.dat;
ampl: option solver gurobi;
ampl: solve;
```

The model file typically declares the variables, objectives, constraints and parameters. Parameters can be the constant coefficients, right-hand-sides, bounds, and even the dimension of the problem. Parameters of an AMPL model can be declared using the keyword "param" the model file.

Let us learn by an example. Consider the following optimisation problem with N variables, one constraint and bounds on variables.

$$\begin{aligned} \min c_1x_1 + c_2x_2 + \ldots + c_Nx_N, & \\ x_1 + x_2 + \ldots + x_N \leq b, & \\ x_i \leq u_i, \quad i = 1, \ldots, N, & \\ x_i \geq l_i, \quad i = 1, \ldots, N. & \end{aligned}$$

here 'N, c, l, u' are parameters that are given to us. We can write it in AMPL model file (say ex1.mod) as

```
param N;
param b;
set S := 1..N;
param c{S};
param lb{S};
param ub{S};

var x{S};

minimize cost: sum{i in S}c[i]*x[i];
s.t. con1: sum{i in S} x[i] <= b;
s.t. ubcon{i in S}: x[i] <= ub[i];
s.t. lbcon{i in S}: x[i] >= lb[i];
```

Pay special note to the 'sum' keyword. It makes long expressions compact. In the data file (say ex1.dat), we can set the value of parameters as

```
param N := 7;
param b := 30;

param c:=
    1  23
    2  12
    3  -6
    4  3.4
    5  8.4
    6  40
    7  11;

param lb:=
    1  1
    2  1
    3  0
    4  0
    5  7
    6  0
    7  2;

param ub:=
    1  4
    2  3
    3  2
    4  2
    5  30
    6  30
    7  13;
```

If we want to change values of 'b' we can change ex1.dat without disturbing our model file. Similarly, if we want to change the number of variables, we can just change N and other parameters in the data file.
The keyword 'set' is used to denote sets in AMPL. One can declare a set in a model file and then initialize variables and parameters according to the elements of the set. For instance, S is a set in the above example. A set does not need to have only numbers in it. In the above example, we could have declared S in the model file and defined it in the data file.

```
# This line is in .mod file
set S;

# This line is in .dat file
set S := 'apple' 'boy' 'cat' 'dog' 'ear' 'fan' 'go';
```

Notice that '{ }' is used in the definition of a set and the '[ ]' is used to access an element of a set.

Style: As the model and data files become more and more complicated, it becomes important that they be readable and easily understandable. Certain rules can be followed to make your files easy to understand. Some examples are provided below:

- Use comments to explain things that may not be clear from the model. Assume that the reader is not an expert in modeling. Write comments whereever you use a special trick or an approximation or an unusual content.

- Write a comment explaining every variable and paramter. See, for instance Exercise 3 on page 5.
- Keep names of variables that are intuitive and related to the problem. For instance, if your variables denote show much steel and cement should be used, then the names of the variables could be `steel` and `cement`.
- Use capital letters to denote sets and parameters. Use small letters to denote variables. For instance, the parameter cost of a unit product could be denoted as `COST`, etc.

Exercise 1: Consider the model described above.

$$\begin{aligned} \min\, & c_1x_1 + c_2x_2 + \ldots + c_Nx_N, \\ & x_1 + x_2 + \ldots + x_N \leq b, \\ & x_i \leq u_i, \quad i = 1, \ldots, N, \\ & x_i \geq l_i, \quad i = 1, \ldots, N. \end{aligned}$$

1. Write the model file ex1.mod and data file ex1.dat as described above.
2. Solve the problem using gurobi solver.
3. [**R**] Report the optimal solution and the value of the cost.
4. Now suppose we want to solve the same problem but with 14 variables. Copy ex1.dat as ex1b.dat and modify it suitably. You may assume that

$$\begin{aligned} & b = 60, \\ & c = [23, 12, -6, 3.4, 8.4, 40, 11, 23, 12, -6, 3.4, 8.4, 40, 11] \\ & lb = [1, 1, 0, 0, 7, 0, 2, 1, 1, 0, 0, 7, 0, 2], \\ & ub = [4, 3, 2, 2, 30, 30, 13, 4, 3, 2, 2, 30, 30, 13] \end{aligned}$$

5. [**R**] Solve the new problem and report the solution.

Exercise 2: Fitting Revisited. (Adapted from Bradley, Hax, and Magnanti, Addison-Wesley, 1977) The selling prices of a number of warehouses in Powai overlooking the lake were given in the following table, along with the size of the lot and its elevation.

Warehouse i	Selling price P_i	Lot size (sq. ft.) L_i	Elevation (feet) E_i
1	155000	12000	350
2	120000	10000	300
3	100000	9000	100
4	70000	8000	200
5	60000	6000	100
6	100000	9000	200

You have been asked by Laxmi Warehousing Company to construct a model to forecast the selling prices of other warehouses in Powai from their lot sizes and elevations. The company feels that a linear model of the form $P = b_0 + b_1 L + b_2 E$ would be reasonably accurate and easy to use. Here b_1 and b_2 would indicate how the price varies with lot size and elevation, respectively, while b_0 would reflect a base price for this section of the city. You would like to select the "best" linear model in some sense. If you knew the three parameters b_0, b_1 and b_2, the six observations in the table would each provide a forecast of the selling price as follows:

$$\hat{P}_i = b_0 + b_1 L_i + b_2 E_i \quad i = 1, 2, \ldots, 6.$$

However, since b_0, b_1 and b_2 cannot, in general, be chosen so that the actual prices P_i are exactly equal to the forecast prices $\hat{P}_i$ for all observations, you would like to minimize the absolute value of the residuals $R_i = P_i - \hat{P}_i$.
In this lab, we will solve this problem for 60 data points instead of 6.

1. First create ex2.mod to model this problem for N data points and $SPRICE$, $LSIZE$ and $ELEV$ as the parameters. You may assume that $b_0 \geq 0$ and b_1, b_2 do not have bounds on them.
2. Create ex2.dat using data from the above table for 6 warehouses.
3. Solve the problem.
4. Now create an ex2b.dat using data from Moodle for 60 data points.
5. [**R**] Solve the problem and report the solution.

Exercise 3: Staff scheduling revisited. Recollect this problem from lab02. It is the beginning of monsoon semester at IIT Bombay, and our department needs a system adminstrator to be working every weekday (Mon-Fri) from 8AM to 10PM. There are six candidates available who can do this job, but they are also busy doing other activities during the week. Their availability and wage-rate is listed in the table below

	Wage-rate	Mon	Tue	Wed	Thu	Fri
KC	150	6	0	6	0	6
DH	152	0	6	0	6	0
HB	148	4	8	4	0	4
SC	146	5	5	5	0	5
KS	166	3	0	3	8	0
NK	176	0	0	0	6	2

Each candidate has a different qualification and hence they have a different wage-rate. According to the contract, KC, DH, HB and SC, must work at least 8 hours every week. KS and NK must work 7 hours every week. There should be exactly one administrator on duty every weekday during the work hours.

We would like to create a model for this problem. Notice that we will need a parameter that changes with both days and candidates. We need a 2-dimensional data structure for available hours. In AMPL, we do this by declaring a parameter

```
set CANDS;           # The set of all candidates.
set DAYS;            # The set of all working days in a week.
param AVAIL_HOURS{CANDS, DAYS}; # Availability of candidate on each day.

var hours{CANDS, DAYS} >= 0; # Hours allocated to each candidate on each day.
```

The data file can then have

```
set CANDS := ‘KC’ ‘DH’ ‘HB’ etc.;
set DAYS  := ‘MON’ ‘TUE’ ‘WED’ etc.;
param AVAIL_HOURS:
    MON  TUE  WED THU FRI =
KC    6    0    6   0   6
DH    0    6    0   6   0
HB    4    8    4   0   4
SC  ...
KS  ...
NK    0    0    0   6   2;
```

To access AVAIL_HOURS for a particular candidate on a day, we can use the syntax `AVAIL_HOURS[KC, WED]`. To sum up all hours used by `KC`, we can do `sum{d in DAYS} hours[KC,d]`. To sum up all hours used by all candidates, we can do `sum{c in CANDS, d in DAYS}hours[c,d]` etc.

1. You are required to find number of hours each candidate must be alloted each day so that the cost of running the facility is minimized. Write an AMPL model for a set `CANDS` of candidates and a set `DAYS` of the week. Your model file should not have ANY data, not even the total number of hours facility is open or the minimum hours that a candidate must work for.
2. Create an appropriate data file using the above data.
3. Solve this problem using Gurobi.

4. [**R**] Report the number of variables, constraints and nonzeros in the constraints of your model. You can use `display _nvars;` to display number of variables in AMPL. Similarly `_ncons` is the number of constraints, and `_snzcons` is the number of nonzeros in the constraint matrix. You can get more statistics from the table A-13 on page 493 of the AMPL book (page 502 of the pdf file on Moodle).

5. [**R**] Report the solution and the total cost.

Exercise 4: Alloy Mix Problem. Bindu Metal Co. produces 100 quintals of metallic alloy product of composition: 40% tin, 35% zinc, and 25% lead. In order to produce this alloy, it can mix 10 available alloys: A-1, A-2, ..., A-10. The alloys have the following composition and costs.

	A-1	A-2	A-3	A-4	A-5	A-6	A-7	A-8	A-9	A-10
%age of tin	80	75	75	60	55	55	40	35	30	30
%age of zinc	15	15	10	20	25	10	50	15	30	55
%age of lead	5	10	15	20	20	35	10	50	40	15
in-house stock (Quintals)	10	12	0	08	15	15	12	10	10	9
in-house cost (per Quintal)	35	50	58	60	44	39	45	55	35	40
purchase cost	72	95	110	125	88	74	95	115	60	84

The company can either use the stocks of raw material alloys available in the company or purchase them from market. The stocks available in the company are limited while the quantity available in market is unlimited. You are given the task of finding how much of each raw-material should be put in the final mix to obtain 100 Quintals of the product.

1. [**R**] Formulate an LP model that can be solved to find the optimal mix of raw-material alloys.
2. Write down the model in AMPL using separate model and data files.
3. [**R**] Solve and report the optimal value and the optimal solution.