Instructions: In this session, we will learn to use AMPL modeling language and software. We will start with very simple exercises in order to gain familiarity with the syntax of the language. Since we will be using AMPL in the next few labs as well, it is important to understand and learn its correct usage.

AMPL Modeling Language

AMPL is a modeling language for writing optimization problems. It is not a solver as it does not really solve the optimization problems. It translates human-readable mathematical notation into arrays and other data-structures which are understood by the solvers. It also translates the solutions obtained from the solvers back into a human-readable form.

There are many solvers that work with AMPL. For instance, CPLEX and GUROBI are solvers for linear programming, convex quadratic programming (objective function is quadratic and constraints are linear), second-order cone programming and their integer counterparts. MINOS, SNOPT and CONOPT are solvers for general convex nonlinear optimization problems. These can not be used to solve problems with integer constraints.

A typical modeling exercise in AMPL takes the following steps

1. Write the model in AMPL's syntax (or AMPL code) using a text editor
2. Select a solver and other options
3. Ask the solver to solve the model
4. Retrieve the solution and other information

Writing an AMPL model

You will have to write all your AMPL-programs using a text editor. The easiest to use text-editor in Linux is 'gedit'. You can start it by clicking on the topmost icon on the left panel and typing 'gedit' and then clicking on 'gedit' icon. Gedit can be used to write text files including optimisation models for AMPL. Other commonly used editors are vim and emacs. They are more advanced (and much more powerful) than gedit, but require some practice and training. You can start them by typing vim or emacs in the terminal. You may use 'Notepad' on Windows. Do not use 'MS-Office or Word' to write any AMPL models. They will not work.

AMPL Syntax and Usage Guide

AMPL user manual is available as a book online on this link: `http://ampl.com/resources/the-ampl-book/chapter-downloads/`. You may refer to it for additional help.

Exercise 1: A Tiny AMPL Example
Create an 'ex1.mod' AMPL model file with the following contents. Do not copy and paste. Type everything yourself.

```
# This line is a comment. Anything written after a hash sign is a comment.
# We do not need a semicolon after a comment.
# Declare variables:
var x1;
var x2;
var y1 <= 1000;

# Declare objective:
minimize cost: 3*x1 + 4*x2 + 5*y1;

# This is the first constraint
s.t. con1: 6*x1 + 8*x2 + 10*y1 >= 200;

# And the second one
s.t. con2: 6*x2 + 5*y1 <= 8;

# Our test model is complete. Now save the file as ex1.mod
```

Now we can run this model. Go back to the terminal window. If you have closed it, then open another terminal and run setup_a.sh as described earlier. Do not close this terminal. Before running AMPL, we first need to change the working directory in the terminal to the directory where you have saved your file. Suppose you saved your AMPL file in a directory 'lab06' in the home folder. First type

```
cd
```

This will change the working directory to the home directory. Then type

```
cd lab06
```

You may use the command

```
ls
```

to see the contents of the directory. Then start AMPL (by typing `ampl` in the command-prompt or the terminal). Solve this model using 'CPLEX' solver by writing the following commands.

```
ampl: reset;
ampl: option solver cplex;
ampl: model ex1.mod;
ampl: solve;
ampl: display x1, x2, y1;
ampl: display con1.body, con1.lb, con1.ub;
ampl: display con2.body, con2.lb, con2.ub;
ampl: display cost;
```

1. Write the optimisation problem that is being solved.
2. Report the values of all four `display` commands above.
3. Check if the reported solution satisfies both the constraints in the model.

4. Now suppose we want to add a lower bound on `y1`: `y1>=10`. Change `var y1 <= 1000;` line to `var y1 >= 10, <= 1000;` in 'ex1.mod' file to add this bound. Solve the problem. (Remember to use the 'reset' command.)

5. Report the values of all three variables and the activities of both constraints at the new optimal point.