```python
############################################################################
# File Name: model12_cplex.py                                              #
# Author: Geonsik Yu, Purdue University, IE Dept                           #
# LP problem (Model 12: Alloy Blending) from:                              #
# https://sites.math.washington.edu/~burke/crs/407/models/m12.html         #
############################################################################
import cplex

## STEP 1. Set up what we need. -----------------------------------------------------------
## Declare variable names
variables = ["x1", "x2", "x3", "x4", "x5"]
## Delare a list of coefficients  of each variable in the objective function (same order)
obj_coeffs = [19.0, 17.0, 23.0, 21.0, 25.0]
## Delare a list of upperbounds of each variable
upperbounds = [cplex.infinity, cplex.infinity, cplex.infinity, cplex.infinity, cplex.infinity]
## Delare a list of lowerbounds of each variable
lowerbounds = [0.0, 0.0, 0.0, 0.0, 0.0]
## Declare contraint names
constraint_names = ["Tin(%)", "Zinc(%)", "Lead(%)"]
## Delare a list of RHS constants of each constraints
righthand = [40.0, 35.0, 25.0]
## Delare a list of inequality directions of each constraints
senses = ['E', 'E', 'E']
## Set coefficients of each variables in each constraints:
lin_expr = [cplex.SparsePair(ind=variables, val=[60.0, 25.0, 45.0, 20.0, 50.0]),
            cplex.SparsePair(ind=variables, val=[10.0, 15.0, 45.0, 50.0, 40.0]),
            cplex.SparsePair(ind=variables, val=[30.0, 60.0, 10.0, 30.0, 10.0])]

## STEP 2. Generate LP problem object -----------------------------------------------------
## Generate an LP problem
problem = cplex.Cplex()
## Set objective as minimization
problem.objective.set_sense( problem.objective.sense.minimize )
## Set variables and objective function
problem.variables.add( obj=obj_coeffs, ub=upperbounds, lb=lowerbounds, names=variables )
## Set constraints
problem.linear_constraints.add(lin_expr = lin_expr, senses = senses, rhs = righthand, names = constraint_names)
## Solve the problem
problem.solve()

## STEP 3. Print out results --------------------------------------------------------------
numrows = problem.linear_constraints.get_num()
numcols = problem.variables.get_num()

print("Solution status = "+ repr(problem.solution.get_status())+ ": " +repr(problem.solution.status[problem.solution.get_status()]))
print("Solution value  = "+ repr(problem.solution.get_objective_value()))

x = problem.solution.get_values()
shadow_price = problem.solution.get_dual_values()
for i in range(numcols):
    print("Variable " + variables[i] + ": Value = " + repr(x[i]))
for i in range(numrows):
    print("Constraint " + constraint_names[i] + ": Shadow Price = " + repr(shadow_price[i]))
```