**DIEGO FERNANDO MARQUEZ BETANCUR**

TRABAJO 2. SOLUCIÓN DE PL EN PYOMO

**1. Problema de producción de Vestidos: (Código Python-pyomo).**

# -\*- coding: utf-8 -\*-

from pyomo.environ import \*

# Dominio del Modelo - Vestidos

vestido = ConcreteModel()

vestido.i = Set(initialize=['Q','R','S','T'], doc='Fabricante')

vestido.j = Set(initialize=['A','B','C','D','E'], doc='Modelo')

#Vectores

vestido.Disp = Param(vestido.i, initialize={'Q':325,'R':300,'S':275,'T':275},doc='Fabricantes cuyas disponibilidades Disp(j)')

vestido.Cant = Param(vestido.j, initialize={'A':150,'B':100,'C':75,'D':250,'E':200}, doc='Demandas mÃ­nimas de vestidos de mujer Cant(i)')

#Matriz

tablac = {

('Q', 'A'): 28,

('Q', 'B'): 35,

('Q', 'C'): 43,

('Q', 'D'): 22,

('Q', 'E'): 15,

('R', 'A'): 30,

('R', 'B'): 32,

('R', 'C'): 45,

('R', 'D'): 18,

('R', 'E'): 10,

('S', 'A'): 25,

('S', 'B'): 35,

('S', 'C'): 48,

('S', 'D'): 20,

('S', 'E'): 13,

('T', 'A'): 33,

('T', 'B'): 27,

('T', 'C'): 40,

('T', 'D'): 25,

('T', 'E'): 27,

}

vestido.C = Param(vestido.i, vestido.j, initialize=tablac, doc='Utilidades (Uij) por vestido varÃ­an de acuerdo con cada fabricante')

#V.Decision:

vestido.x = Var(vestido.i, vestido.j, bounds=(0.0,None), doc='Cantidad de vestidos modelo i comprados a tienda j')

# s.a: - Restricciones:

def supply\_rule(vestido, i):

return sum(vestido.x[i,j] for j in vestido.j) <= vestido.Disp[i]

vestido.supply = Constraint(vestido.i, rule=supply\_rule, doc='No exceder la capacidad de cada fabricante i')

def demand\_rule(vestido, j):

return sum(vestido.x[i,j] for i in vestido.i) >= vestido.Cant[j]

vestido.demand = Constraint(vestido.j, rule=demand\_rule, doc='Satisfacer la demanda de cada modelo j')

#Funcion Objetivo:

def objective\_rule(vestido):

return sum(vestido.C[i,j]\*vestido.x[i,j] for i in vestido.i for j in vestido.j)

vestido.objective = Objective(rule=objective\_rule, sense=maximize, doc='FunciÃ³n objetivo')

# Funcion para llamar al solucionador de problema (NEOS)

instance = vestido

opt = SolverFactory("cbc")

solver\_manager = SolverManagerFactory('neos')

results = solver\_manager.solve(instance, opt=opt)

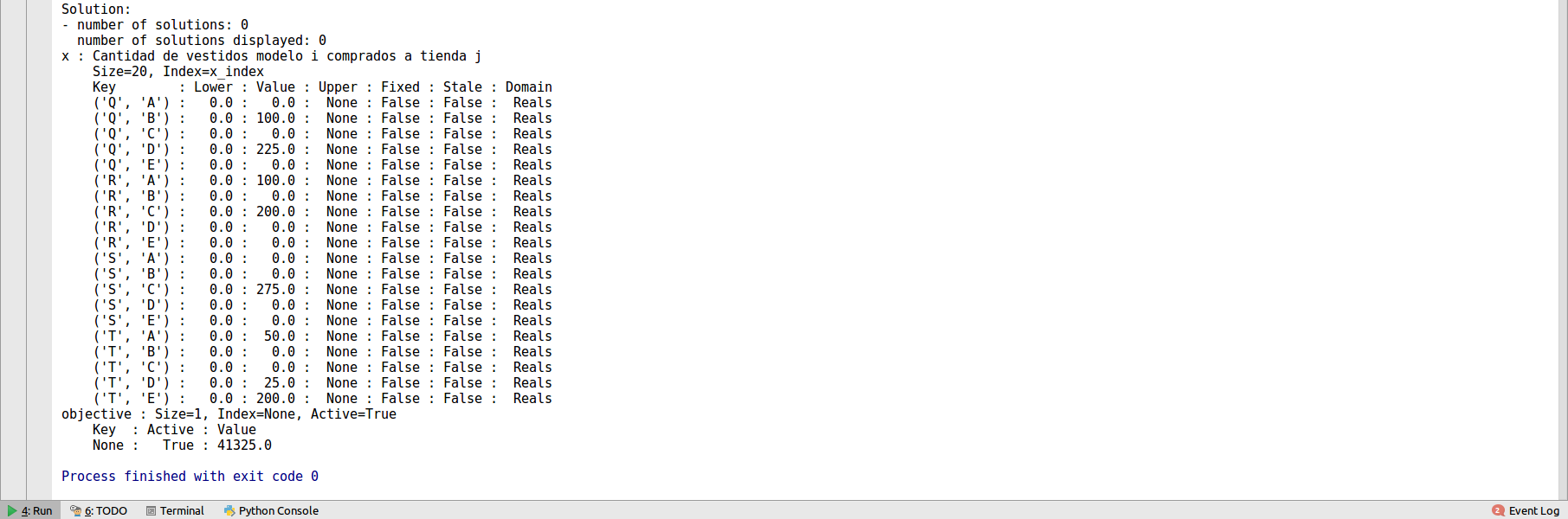
results.write()

vestido.x.display()

vestido.objective.display()

**2. Resultados del problema:**





**La solución óptima del problema indica que:**

('Q', 'A') = 0

('Q', 'B') = 100

('Q', 'C') = 0

('Q', 'D') = 225

('Q', 'E') = 0

('R', 'A') = 100

('R', 'B') = 0

('R', 'C') = 200

('R', 'D') = 0

('R', 'E') = 0

('S', 'A') = 0

('S', 'B') = 0

('S', 'C') = 275

('S', 'D') = 0

('S', 'E') = 0

('T', 'A') = 50

('T', 'B') = 0

('T', 'C') = 0

('T', 'D') = 25

('T', 'E') = 200

**Maximización para el costo de esta asignación óptima es de $41325**