In [1]:

```
#Problem 3-2
using JuMP, Cbc, NamedArrays

computers = [:Pear, :Apricot, :Mango]
labor =  Dict(zip(computers,[1,2,1.8]))
chips =  Dict(zip(computers,[2,5,6]))
sell =  Dict(zip(computers,[400,900,1000]))
production = Dict(zip(computers,[500,400,300]))
M = Dict(zip(computers,[2500, 1500, 1200]))

m = Model()

@variable(m, x[computers] >= 0)
@variable(m, y[computers] >=0)

@objective(m, Max, sum(sell[i]*x[i] for i in computers))

@constraint(m, sum(chips[i] * x[i] for i in computers ) <= 3000) #chip constraint
@constraint(m, sum(labor[i] * x[i] for i in computers) <= 1200) #labor constraint
@constraint(m, prod[i in computers], x[i] >= production[i]*y[i])

#@constraint(m, rhs[i in computers], x[i] >= 1000*y[i])
#@constraint(m, bin[i in computers], y[i] == 0)

@constraint(m, bound[i in computers], x[i] <= M[i]*y[i])
@constraint(m, nonzero[i in computers], x[i] >= 0)


set_optimizer(m, Cbc.Optimizer)
optimize!(m)
```

```
Presolve 2 (-9) rows, 3 (-3) columns and 6 (-15) elements
0  Obj -0 Dual inf 2700 (3)
3  Obj 550000
Optimal - objective value 550000
After Postsolve, objective 550000, infeasibilities - dual 0 (0), primal 0 (0)
Optimal objective 550000 - 3 iterations time 0.002, Presolve 0.00
```

In [ ]: