```julia
using JuMP
using HiGHS

# Model creation
model = Model(HiGHS.Optimizer)

# Decision variables
@variable(model, x[1:6] >= 0)

# Parameters
production = [350, 240, 550, 200, 300, 600]
costs = [30, 40, 35, 45, 38, 50]

# Objective
@objective(model, Min, sum(costs[i] * x[i] for i in 1:6))

# Chemical balance constraints
@constraint(model, x[1] <= 1000)  # For 10 AM, chemical limit can be
set initially
for t in 2:6
    @constraint(model, sum(production[1:t]) - sum(x[1:t]) <= 1000)  #
Chemical balance without name
end

# Capacity limits thru a loop
@constraint(model, x[1] <= 650)
for t in 2:6
    @constraint(model, x[t] <= 650)
end

# No overnight storage constrait
@constraint(model, sum(production) - sum(x) == 0)

# Solve
optimize!(model)

# Print
optimal_value = objective_value(model)
decision_variables = value.(x)

println("Optimal objective value: ", optimal_value)
println("Decision variables (amount of Chemical X to recycle): ",
decision_variables)

# the final number isn't correct :(
```

```
Running HiGHS 1.7.2 (git hash: 5ce7a2753): Copyright (c) 2024 HiGHS
under MIT licence terms
Coefficient ranges:
  Matrix [1e+00, 1e+00]
  Cost   [3e+01, 5e+01]
```

```
  Bound  [0e+00, 0e+00]
  RHS    [1e+02, 2e+03]
Presolving model
5 rows, 6 cols, 24 nonzeros  0s
4 rows, 6 cols, 18 nonzeros  0s
Presolve : Reductions: rows 4(-9); columns 6(-0); elements 18(-15)
Solving the presolved LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0     0.0000000000e+00 Pr: 4(3360) 0s
          1     7.8550000000e+04 Pr: 0(0) 0s
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Simplex   iterations: 1
Objective value     :  7.8550000000e+04
HiGHS run time      :          0.01
Optimal objective value: 78550.0
Decision variables (amount of Chemical X to recycle): [650.0, 290.0,
650.0, 0.0, 650.0, 0.0]
```