Question 3

The CRUD chemical plant produces as part of its production process a noxious compound called chemical X. Chemical X is highly toxic and needs to be disposed of properly. Fortunately, CRUD is linked by a pipe system to the FRESHAIR recycling plant that can safely reprocess chemical X. On any given day, the CRUD plant will produce the following amount of chemical X (the plant operates between 9am and 3pm only):

Time	9-10AM	10-11AM	11AM-12PM	12-1PM	1-2PM	2-3PM
Chemical X (in liters)	300	240	600	200	300	600

Becuase of environmental regulation, at no point in time is the CRUD plant allowed to keep more than 1000 litres on site and no chemical X is allowed to be kept overnight. At the top of every hour, at most 650 litres of chemical X can be sent to the FRESHAIR recycling plant. The cost of recycling chemical X is different for every hour:

Time	10AM	11AM	12PM	1PM	2PM	3PM
$/Chemical X (in liters)	30	40	35	45	38	5 0

You need to decide how much chemical to send from the CRUD plant to the FRESHAIR recycling plant at the top of each hour, so that you can minimize the total recycling cost but also meet all of the evironmental constraints. Formulate this problem as an LP.

In [ ]:
```
# declaring model
using JuMP, HiGHS

chemical = Model(HiGHS.Optimizer)
```

Out[ ]:
```
A JuMP Model
Feasibility problem with:
Variables: 0
Model mode: AUTOMATIC
CachingOptimizer state: EMPTY_OPTIMIZER
Solver name: HiGHS
```

In [ ]:
```
# vector for amount of chemical X produced each hour
production = [300 240 600 200 300 600]
# vector for cost of sending chemical X each hour
cost = [30 40 35 45 38 50]
```

Out[ ]:
```
1×6 Matrix{Int64}:
 30  40  35  45  38  50
```

In [ ]:
```
# declaring variable for current amount of chemical X at the end of each hour
@variable(chemical, 1000 >= current[1:6] >= 0)
# declaring variable for amount of chemical X sent to FRESHAIR
@variable(chemical, 650 >= sent[1:6] >= 0)
```

Out[ ]:
```
6-element Vector{VariableRef}:
 sent[1]
 sent[2]
 sent[3]
 sent[4]
 sent[5]
 sent[6]
```

In [ ]:
```
# constraints

# at the end of the final hour, amount of chemical X must be 0
@constraint(chemical, endcond, current[6] == 0)
# conservation
@constraint(chemical, flow1, 300 - sent[1] == current[1])
@constraint(chemical, flow2, current[1] + 240 - sent[2] == current[2])
@constraint(chemical, flow3, current[2] + 600 - sent[3] == current[3])
@constraint(chemical, flow4, current[3] + 200 - sent[4] == current[4])
@constraint(chemical, flow5, current[4] + 300 - sent[5] == current[5])
@constraint(chemical, flow6, current[5] + 600 - sent[6] == current[6])
```

Out[ ]:
flow6 : $current_5 - current_6 - sent_6 = -600.0$

In [ ]:
```
# defining objective function
@objective(chemical, Min, sum(cost[i]*sent[i] for i in 1:6))

print(chemical)
```

$$
\begin{aligned}
\min \quad & 30sent_1 + 40sent_2 + 35sent_3 + 45sent_4 + 38sent_5 + 50sent_6 \\
\text{Subject to} \quad & current_6 = 0.0 \\
& -current_1 - sent_1 = -300.0 \\
& current_1 - current_2 - sent_2 = -240.0 \\
& current_2 - current_3 - sent_3 = -600.0 \\
& current_3 - current_4 - sent_4 = -200.0 \\
& current_4 - current_5 - sent_5 = -300.0 \\
& current_5 - current_6 - sent_6 = -600.0 \\
& current_1 \geq 0.0 \\
& current_2 \geq 0.0 \\
& current_3 \geq 0.0 \\
& current_4 \geq 0.0 \\
& current_5 \geq 0.0 \\
& current_6 \geq 0.0 \\
& sent_1 \geq 0.0 \\
& sent_2 \geq 0.0 \\
& sent_3 \geq 0.0 \\
& sent_4 \geq 0.0 \\
& sent_5 \geq 0.0 \\
& sent_6 \geq 0.0 \\
& current_1 \leq 1000.0 \\
& current_2 \leq 1000.0 \\
& current_3 \leq 1000.0 \\
& current_4 \leq 1000.0 \\
& current_5 \leq 1000.0 \\
& current_6 \leq 1000.0 \\
& sent_1 \leq 650.0 \\
& sent_2 \leq 650.0 \\
& sent_3 \leq 650.0 \\
& sent_4 \leq 650.0 \\
& sent_5 \leq 650.0 \\
& sent_6 \leq 650.0
\end{aligned}
$$

In [ ]:

```
# solving the model
optimize!(chemical);

# outputs detailed information about the solution process
@show solution_summary(chemical);

value.(sent)
```

```
Solving LP without presolve or with basis
Model   status      : Optimal
Objective value     :  8.8050000000e+04
HiGHS run time      :          0.00
solution_summary(chemical) = * Solver : HiGHS

* Status
  Termination status : OPTIMAL
  Primal status      : FEASIBLE_POINT
  Dual status        : FEASIBLE_POINT
  Message from the solver:
  "kHighsModelStatusOptimal"

* Candidate solution
  Objective value      : 8.80500e+04
  Objective bound      : 0.00000e+00
  Relative gap         : Inf
  Dual objective value : 8.80500e+04

* Work counters
  Solve time (sec)   : 3.38403e-03
  Simplex iterations : 0
  Barrier iterations : 0
  Node count         : -1
```

Out[ ]:
```
6-element Vector{Float64}:
 300.0
  40.0
 650.0
   0.0
 650.0
 600.0
```