Question 2

Hackensack Blended Whiskey Company uses three grades of whiskey to make blended varieties. They do not produce their own whiskey, and instead purchase three varieties to make their blends: Standard, Choice, and Prime. These unblended grades can be used to make the following two brands of whiskey, with associated characteristics:

- Scottish Club: Must contain at least 60% Prime and at most 20% Standard. Scottish Club sells for $6.80 per liter.
- Johnny Gold: Must contain at least 15% Prime and at most 60% Standard. Johnny Gold sells for $5.70 per liter.

The amount of available raw whiskey and their associated costs are:

Whiskey	Available (no. of liters)	Cost per liter (in $)
Standard	1,200	4.00
Choice	2,500	5.00
Prime	2,000	7.00

Hackensack doesn't want to produce more whiskey than it knows it can sell. It estimates its current demand for each type of blended whiskey as:

Blended Whiskey	Demand (in liters)
Scottish Club	1,000
Johnny Gold	600

Hackensack can increase its demand by advertising its blended whiskeys. For each dollar spent on advertising any type of its blended whiskeys, that type of whiskey's demand will increase by 1.25 liters.

Formulat an LP to maximize the total profit (revenue minus cost):

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

# defining model
whiskey = 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 raw whiskey
raw_whiskey = [:Standard, :Choice, :Prime]

# vector for blended whiskey
blended_whiskey = [:ScottishClub, :JohnnyGold]
```

Out[ ]:
```
2-element Vector{Symbol}:
 :ScottishClub
 :JohnnyGold
```

In [ ]:
```
# declaring variables
@variable(whiskey, x[raw_whiskey, blended_whiskey] >= 0)
@variable(whiskey, advertising >= 0)
@variable(whiskey, advertising2 >= 0)
```

Out[ ]:

$$advertising2$$

In [ ]:
```
# scottish club constraints
@constraint(whiskey, SCBound1, x[:Prime, :ScottishClub] >= 0.6*sum(x[i,:Scott
@constraint(whiskey, SCBound2, x[:Standard, :ScottishClub] <= 0.2*sum(x[i,:Sc

# johnny gold constraints
@constraint(whiskey, JGBound1, x[:Prime, :JohnnyGold] >= 0.15*sum(x[i,:Johnny
@constraint(whiskey, JGBound2, x[:Standard, :JohnnyGold] <= 0.6*sum(x[i,:John

# raw whiskey constraints
@constraint(whiskey, SumBound, sum(x[:Standard, i] for i in blended_whiskey)
@constraint(whiskey, SumBound2, sum(x[:Choice, i] for i in blended_whiskey) <
@constraint(whiskey, SumBound3, sum(x[:Prime, i] for i in blended_whiskey) <=

# demand constraints
@constraint(whiskey, demand, sum(x[i, :ScottishClub] for i in raw_whiskey) <=
@constraint(whiskey, demand2, sum(x[i, :JohnnyGold] for i in raw_whiskey) <=
```

Out[ ]:
demand2 :

$$x_{Standard,JohnnyGold} + x_{Choice,JohnnyGold} + x_{Prime,JohnnyGold} - 1.25advertising2 \leq 600.0$$

In [ ]:
```
# formulate the objective function
revenue = 6.80*sum(x[i, :ScottishClub] for i in raw_whiskey) + 5.70*sum(x[i,
cost = 4.00*sum(x[:Standard, i] for i in blended_whiskey) + 5.00*sum(x[:Choic
@objective(whiskey, Max, revenue - cost)
```

Out[ ]:

$$2.8x_{Standard,ScottishClub} + 1.7999999999999998x_{Choice,ScottishClub} - 0.20000000$$
$$+ 1.7000000000000002x_{Standard,JohnnyGold} + 0.7000000000000002x_{Choice,JohnnyGold} - 1$$
$$- advertising - advertising2$$

In [ ]:
```
# solving the model
optimize!(whiskey);

# outputs detailed information about the solution process
@show solution_summary(whiskey);
value.(x)
```

```
Presolving model
9 rows, 8 cols, 26 nonzeros
9 rows, 8 cols, 26 nonzeros
Presolve : Reductions: rows 9(-0); columns 8(-0); elements 26(-0)
Solving the presolved LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0    -1.1199987222e+01 Ph1: 7(10.8); Du: 4(11.2) 0s
          8    -1.6133333333e+03 Pr: 0(0) 0s
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Simplex   iterations: 8
Objective value     :  1.6133333333e+03
HiGHS run time      :          0.00
solution_summary(whiskey) = * Solver : HiGHS

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

* Candidate solution
  Objective value      : 1.61333e+03
  Objective bound      : 1.61333e+03
  Relative gap         : Inf
  Dual objective value : 1.61333e+03

* Work counters
  Solve time (sec)   : 1.16395e-03
  Simplex iterations : 8
  Barrier iterations : 0
  Node count         : -1
```

Out[ ]:
```
2-dimensional DenseAxisArray{Float64,2,...} with index sets:
    Dimension 1, [:Standard, :Choice, :Prime]
    Dimension 2, [:ScottishClub, :JohnnyGold]
And data, a 3×2 Matrix{Float64}:
 200.0  1000.0
 200.0   416.667
 600.0   250.0
```