Assignment 2

Q.The Weigelt Corporation has three branch plants with excess production capacity. Fortunately, the corporation has a new product ready to begin production, and all three plants have this capability, so some of the excess capacity can be used in this way. This product can be made in three sizes–large, medium, and small–that yield a net unit profit of $420, $360, and $300, respectively. Plants 1, 2, and 3 have the excess capacity to produce 750, 900, and 450 units per day of this product, respectively, regardless of the size or combination of sizes involved. The amount of available in-process storage space also imposes a limitation on the production rates of the new product. Plants 1, 2, and 3 have 13,000, 12,000, and 5,000 square feet, respectively, of in-process storage space available for a day's production of this product. Each unit of the large, medium, and small sizes produced per day requires 20, 15, and 12 square feet, respectively. Sales forecasts indicate that if available, 900, 1,200, and 750 units of the large, medium, and small sizes, respectively, would be sold per day. At each plant, some employees will need to be laid off unless most of the plant's excess production capacity can be used to produce the new product. To avoid layoffs if possible, management has decided that the plants should use the same percentage of their excess capacity to produce the new product. Management wishes to know how much of each of the sizes should be produced by each of the plants to maximize profit. Solve the problem using lpsolve, or any other equivalent library in R. We have to maximize the profit for the Weigelt corporation. Let us suppose that xij represents the number of units of size i to be produced by the plant j. For the sake of representation, I have taken i= 1,2,3 for the three sizes large, medium and small respectively. So, we can write our LPP problem as follows:

Maximize: Z= 420 *(x11 + x12 + x13) + 360* (x21 + x22 + x23) + 300 *(x31 + x32 + x33) Subjected to the following constraints: x11 + x21 + x31 <= 750 x12 + x22 + x32 <= 900 x13 + x23 + x33 <= 450 20x11 + 15x21 + 12x31 <= 13000 20x12 + 15x22 + 12x32 <= 12000 20x13 + 15x23 + 12x33 <= 5000 x11 + x12 + x13 <= 900 x21 + x22 + x23 <= 1200 x31 + x32 + x33 <= 750 900x11 + 900x21 + 900x31 - 750x12 - 750x22 - 750x32 = 0 450x11 + 450x21 + 450x31 - 750x13 - 750x23 - 750x33 = 0 x11, x12, x13, x21, x22, x23, x31, x32, x33 must be an integer.

I used "lpSolveAPI" library in R to solve the above LPP problem. R code is shown below. The optimal solution came out to be as follows: x11= 530, x12= 0, x13= 1, x21= 160, x22= 688, x23= 8, x31= 0, x32= 140, x33= 405 The optimal profit for the above solution = $694,680

```
#installing tha package and calling the library
library(lpSolveAPI)
```

```
## Warning: package 'lpSolveAPI' was built under R version 4.1.2
```

```
#initializing a lpp problem
lpmodel= make.lp(0,9)
lpmodel
```

```
## Model name:
##   a linear program with 9 decision variables and 0 constraints
```

```
# setting the objective function
set.objfn(lpmodel,  c(420,420,420,360,360,360,300,300,300))
lp.control(lpmodel, sense= "max", basis.crash= "mostfeasible", presolve= c("knapsack",
"mergerows"), timeout= 1000)
```

```
## $anti.degen
## [1] "none"
##
## $basis.crash
## [1] "mostfeasible"
##
## $bb.depthlimit
## [1] -50
##
## $bb.floorfirst
## [1] "automatic"
##
## $bb.rule
## [1] "pseudononint" "greedy"       "dynamic"      "rcostfixing"
##
## $break.at.first
## [1] FALSE
##
## $break.at.value
## [1] 1e+30
##
## $epsilon
##       epsb       epsd      epsel     epsint epsperturb   epspivot
##      1e-10      1e-09      1e-12      1e-07      1e-05      2e-07
##
## $improve
## [1] "dualfeas" "thetagap"
##
## $infinite
## [1] 1e+30
##
## $maxpivot
## [1] 250
##
## $mip.gap
## absolute relative
##    1e-11    1e-11
##
## $negrange
## [1] -1e+06
##
## $obj.in.basis
## [1] TRUE
##
## $pivoting
## [1] "devex"    "adaptive"
##
## $presolve
## [1] "knapsack"  "mergerows"
##
## $scalelimit
## [1] 5
```

```
##
## $scaling
## [1] "geometric"   "equilibrate"  "integers"
##
## $sense
## [1] "maximize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 1000
##
## $verbose
## [1] "neutral"
```

```
#setting the constraint of decision variables to be integer
set.type(lpmodel, columns = seq(1,9), type = "integer")

# constraint for excess capacity at each plant
add.constraint(lpmodel, xt= rep(c(1,0,0),3), type = "<=", rhs = 750)
add.constraint(lpmodel, xt= rep(c(0,1,0),3), type = "<=", rhs = 900)
add.constraint(lpmodel, xt= rep(c(0,0,1),3), type = "<=", rhs = 450)

# constraint for sales forecast of each type of product
add.constraint(lpmodel, xt= c(1,1,1,0,0,0,0,0,0), type = "<=", rhs = 900)
add.constraint(lpmodel, xt= c(0,0,0,1,1,1,0,0,0), type = "<=", rhs = 1200)
add.constraint(lpmodel, xt= c(0,0,0,0,0,0,1,1,1), type = "<=", rhs = 750)

# constraint for available in process storage space at each plant
add.constraint(lpmodel, xt= c(20,0,0,15,0,0,12,0,0), type = "<=", rhs = 13000)
add.constraint(lpmodel, xt= c(0,20,0,0,15,0,0,12,0), type = "<=", rhs = 12000)
add.constraint(lpmodel, xt= c(0,0,20,0,0,15,0,0,12), type = "<=", rhs = 5000)

# constraint for same percentage of excess capacity to be used in all plants
add.constraint(lpmodel, xt= c(900,-750,0,900,-750,0,900,-750,0), type = "=", rhs = 0)
add.constraint(lpmodel, xt= c(450,0,-750,450,0,-750,450,0,-750), type = "=", rhs = 0)

# solving the LPP model
write.lp(lpmodel, 'model.lp', type = 'lp')
solve.lpExtPtr(lpmodel)
```

```
## [1] 0
```

```
#getting the output of LPP models
get.variables(lpmodel)
```

```
## [1] 530   0   1 160 688   8   0 140 405
```

```
get.objective(lpmodel)
```

```
## [1] 694680
```

```
get.constraints(lpmodel)
```

```
##  [1]   690   828   414   531   856   545 13000 12000  5000     0     0
```