```
using CP;

 //general parameters

 {string} DrugCategories=...;
 int numofDrugs=...;
 range n=1..numofDrugs;
 int numofPharmacies=...;
 range p=1..numofPharmacies;
 int months=...;
 range m=1..months;
 float fsD[n]=...;
 int pd[m][p][n]=...; //actual values of demand during each month in last year

 //pharmacy parameters

 float hP[p]=...;
 float oP[p]=...;
 float fsP[p]=...;
 float cc[p]=...; //compensation cost
 int ss[n]=...; //safety stock

int sq[m][p][n]=...; //shortage quantity
 int eq[m][p][n]=...; //extra quantity

//warehouse (Hikma) parameters

 int quarters=...;
 range q=1..quarters;
 float hW[q]=...;
 int qD[q][n]=...; // quarter demand

 //decision variables


dvar int+ Q[m][p][n]; // quantity to be ordered in each month by each pharmacy for
each drug

 //Stochastic formulation

float lf=...; // lower factor to find stochastic demand during lead time
float uf=...; // upper factor to find stochastic demand during lead time
float md[i in m,j in p,k in n] = round(lf*pd[i][j][k]); // lowest value of demand
during lead time (minimum d)
execute { writeln("md = ",md); }
float Md[i in m,j in p,k in n] = round(uf*pd[i][j][k]); // highest value of demand
during lead time (maximum d)
execute { writeln("Md = ",Md); }

 //Random generations

 execute{
  var now = new Date();
    Opl.srand(now.getTime()%Math.pow(2,31));  //generation function srand to get diff
#s in each run
```

```
}

int scale=10; //number of decimal points

float cd[i in 1..12 , j in 1..3 , k in 1..2]=  round(md[i][j][k]+1 / scale *
(rand(scale+1)) * (Md[i][j][k]-md[i][j][k]));

execute
{

writeln("current demand= ",cd);

}

assert forall(i in 1..12 , j in 1..3 , k in 1..2) as1:md[i][j][k]<=cd[i][j][k];
assert forall(i in 1..12 , j in 1..3 , k in 1..2) as2:cd[i][j][k]<=Md[i][j][k];



 //objective function

 dexpr float zP= sum(i in m, j in p , k in n)( (hP[j]*((Q[i][j][k]/2) + ss[k]))+
(oP[j])+ (sq[i][j][k] * cc[j]) ) ;

 dexpr float zW= sum(i in q , j in n)( (hW[j]*qD[i][j]) ) ;

 dexpr float zT= zP + zW;

 minimize zT;


//constraints

subject to{

 // pharmacies floorspace

forall(i in m , j in p) {

   sum(k in n)(fsD[k]*Q[i][j][k])<=fsP[j];

   }

 forall(i in m, j in p , k in n) {

(0.85*pd[i][j][k])+sq[i-1][j][k]<= Q[i][j][k];

}


   forall(i in m, j in p , k in n) {

   ((Q[i][j][k] + ss[k])<=cd[i][j][k])==((sq[i][j][k])==(cd[i][j][k]-(Q[i][j][k] +
ss[k])));
}
```

```
forall(i in m, j in p , k in n) {

   ((Q[i][j][k] + ss[k])>=cd[i][j][k])==((eq[i][j][k])==((Q[i][j][k] + ss[k])-
cd[i][j][k]));
}




}
```