Profit Optimisation

This is very similar to the demand fill optimisation example. Except with profit we want to maximise plus explicit weights are now needed. For example a product may have a gross profit of $4 per unit when there is demand but may need to sell at a marked down price thereafter and may instead make a loss of $1 per unit from that point on. It is weights like these that we are interested in when considering profit optimisation models.

Below we use a simple example. That us humans can quickly and easily solve. The approach below highlights the importants of this algorithms ability to loop through previous solves to give it a chance to find a better solution to the problem after the first passthrough.

```
   ]<xo=:(3 3) $ 2 0 0 1 1 0 1 0 1
┌─────┐
│2 0 0│
│1 1 0│
│1 0 1│
└─────┘
   ]xp=:4 2 2
4 2 2
   ]<xt=:(4,3) $ 0 NB. start with all zeros
┌─────┐
│0 0 0│
│0 0 0│
│0 0 0│
│0 0 0│
└─────┘
   ]<wdf=:10 1 1 NB. demand fill weights
┌──────┐
│10 1 1│
└──────┘
   ]<wde=:_10 0 0 NB. demand exceeded weights
┌───────┐
│_10 0 0│
└───────┘
   eval=:3 : 0
csy=.+/"2 y
df=. xp <. csy
de=.0 >. csy-df
+/ (de*wde), (df*wdf)
)
   NB. eval xt
   bs=:3 : '(}:xt) ,~ xo{~ (] i. >./) {{eval y, }: xt}}"1 xo' NB. best solve finder
   solver=: 3 : 0
xt=:bs 1
eval xt
)
   NB. first try:
   solver"0 i.4
20 40 31 22
   <xt
┌─────┐
│1 1 0│
│1 1 0│
```

```
│2 0 0│
│2 0 0│
└─────┘
   xp - +/"2 xt
_2 0 2
   NB. second try:
   solver"0 i.4
33 44 44 44
   <xt
┌─────┐
│1 1 0│
│1 1 0│
│1 0 1│
│1 0 1│
└─────┘
   xp - +/"2 xt
0 0 0
```