In [28]:
```
from pulp import *
import pandas as pd
import numpy as np
```

In [171]:
```
= ['PadFab','FlecFab','MetRod','PVCRod','MetLeg','PVCLeg','Package']
= ['MetalCut','PVCCut','FabCut','FabSew','KitAssm']
= ['36x30M','30x24M','30x24PVC','24x18PVC','36x30Flec','36x30Pad','30x24F
   '24x18PVCFlec']
```

In [172]:
```
M = W + P
R = W + C + P

raw = W + C
```

In [173]:
```
T = range(1,8)
```

In [174]:
```
T
```

Out[174]:
```
range(1, 8)
```

In [175]:
```
## Reading resources data into a 30*30 matrix (from PabFab to 24x18PVCFl
## its been converted to its sparse form, by putting 0s for not listed r
```

In [176]:
```
df_1 = pd.read_csv('Bill Of Resources.csv', index_col=0)
resources_bill_dict = makeDict([R,R],df_1.to_numpy())
```

In [177]:
```
## Reading supply data into a dictionary
```

In [178]:
```
df_2 = pd.read_csv('supply_data.csv', index_col=0)
supply_dict = makeDict([R,T],df_2.to_numpy())
```

In [179]:
```
supply_dict
```

Out[179]:
```
{'PadFab': {1: 3000, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'FlecFab': {1: 5000, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetRod': {1: 6500, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'PVCRod': {1: 10500, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetLeg': {1: 200, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'PVCLeg': {1: 400, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'Package': {1: 400, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetalCut': {1: 180, 2: 180, 3: 180, 4: 180, 5: 150, 6: 0, 7: 0},
 'PVCCut': {1: 240, 2: 240, 3: 240, 4: 240, 5: 240, 6: 0, 7: 0},
 'FabCut': {1: 240, 2: 240, 3: 240, 4: 300, 5: 300, 6: 0, 7: 0},
 'FabSew': {1: 480, 2: 480, 3: 0, 4: 360, 5: 360, 6: 0, 7: 0},
 'KitAssm': {1: 240, 2: 240, 3: 240, 4: 240, 5: 240, 6: 0, 7: 0},
 '36x30M': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24M': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24PVC': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '24x18PVC': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30Flec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30Pad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24Flec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24Pad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '24x18Flec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '24x18Pad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30MPad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30MFlec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24MPad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24MFlec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24PVCPad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '30x24PVCFlec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '24x18PVCPad': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '24x18PVCFlec': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0}}
```

In [180]:
```
## Reading Scrapping cost into a 30*7 matrix
## (Rows: from PabFab to 24x18PVCFlec in order, Columns: column[0] corre
```

In [181]:
```
df_3 = pd.read_csv('scr_cost.csv', index_col=0)
scrcost_dict = makeDict([R,T],df_3.to_numpy())
scrcost_dict
```

Out[181]:
```
{'PadFab': {1: 0.02, 2: 0.02, 3: 0.02, 4: 0.02, 5: 0.02, 6: 0.02, 7: 0
.02},
 'FlecFab': {1: 0.02, 2: 0.02, 3: 0.02, 4: 0.02, 5: 0.02, 6: 0.02, 7:
0.02},
 'MetRod': {1: 0.01, 2: 0.01, 3: 0.01, 4: 0.01, 5: 0.01, 6: 0.01, 7: 0
.01},
 'PVCRod': {1: 0.01, 2: 0.01, 3: 0.01, 4: 0.01, 5: 0.01, 6: 0.01, 7: 0
.01},
 'MetLeg': {1: 0.01, 2: 0.01, 3: 0.01, 4: 0.01, 5: 0.01, 6: 0.01, 7: 0
```

```
.01},
 'PVCLeg': {1: 0.03, 2: 0.03, 3: 0.03, 4: 0.03, 5: 0.03, 6: 0.03, 7: 0
.03},
 'Package': {1: 0.05, 2: 0.05, 3: 0.05, 4: 0.05, 5: 0.05, 6: 0.05, 7:
0.05},
 'MetalCut': {1: 0.7, 2: 0.7, 3: 0.7, 4: 0.7, 5: 0.7, 6: 0.7, 7: 0.7},
 'PVCCut': {1: 0.5, 2: 0.5, 3: 0.5, 4: 0.5, 5: 0.5, 6: 0.5, 7: 0.5},
 'FabCut': {1: 0.5, 2: 0.5, 3: 0.5, 4: 0.5, 5: 0.5, 6: 0.5, 7: 0.5},
 'FabSew': {1: 0.5, 2: 0.5, 3: 0.5, 4: 0.5, 5: 0.5, 6: 0.5, 7: 0.5},
 'KitAssm': {1: 0.5, 2: 0.5, 3: 0.5, 4: 0.5, 5: 0.5, 6: 0.5, 7: 0.5},
 '36x30M': {1: 10.0, 2: 10.0, 3: 10.0, 4: 10.0, 5: 10.0, 6: 10.0, 7: 1
0.0},
 '30x24M': {1: 10.0, 2: 10.0, 3: 10.0, 4: 10.0, 5: 10.0, 6: 10.0, 7: 1
0.0},
 '30x24PVC': {1: 7.5, 2: 7.5, 3: 7.5, 4: 7.5, 5: 7.5, 6: 7.5, 7: 7.5},
 '24x18PVC': {1: 7.5, 2: 7.5, 3: 7.5, 4: 7.5, 5: 7.5, 6: 7.5, 7: 7.5},
 '36x30Flec': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5}
,
 '36x30Pad': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5},
 '30x24Flec': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5}
,
 '30x24Pad': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5},
 '24x18Flec': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5}
,
 '24x18Pad': {1: 5.5, 2: 5.5, 3: 5.5, 4: 5.5, 5: 5.5, 6: 5.5, 7: 5.5},
 '36x30MPad': {1: 12.0, 2: 12.0, 3: 12.0, 4: 12.0, 5: 12.0, 6: 12.0, 7
: 12.0},
 '36x30MFlec': {1: 12.0, 2: 12.0, 3: 12.0, 4: 12.0, 5: 12.0, 6: 12.0,
7: 12.0},
 '30x24MPad': {1: 12.0, 2: 12.0, 3: 12.0, 4: 12.0, 5: 12.0, 6: 12.0, 7
: 12.0},
 '30x24MFlec': {1: 12.0, 2: 12.0, 3: 12.0, 4: 12.0, 5: 12.0, 6: 12.0,
7: 12.0},
 '30x24PVCPad': {1: 11.0,
  2: 11.0,
  3: 11.0,
  4: 11.0,
  5: 11.0,
  6: 11.0,
  7: 11.0},
 '30x24PVCFlec': {1: 11.0,
  2: 11.0,
  3: 11.0,
  4: 11.0,
  5: 11.0,
  6: 11.0,
  7: 11.0},
 '24x18PVCPad': {1: 11.0,
  2: 11.0,
  3: 11.0,
  4: 11.0,
  5: 11.0,
  6: 11.0,
  7: 11.0},
 '24x18PVCFlec': {1: 11.0,
  2: 11.0,
  3: 11.0,
  4: 11.0,
  5: 11.0,
  6: 11.0,
  7: 11.0}}
```

In [182]:

```
## Reading inventory cost into a dictionary
## very high inventory cost (10000) has been modelled for the non invent
```

In [183]:

```
df_4 = pd.read_csv('inventory_cost.csv', index_col=0)
inv_dict = makeDict([R,T],df_4.to_numpy())
inv_dict
```

```
  2: 0.015,
  3: 0.015,
  4: 0.015,
  5: 0.015,
  6: 0.015,
  7: 0.015},
 '36x30Flec': {1: 0.005,
  2: 0.005,
  3: 0.005,
  4: 0.005,
  5: 0.005,
  6: 0.005,
  7: 0.005},
 '36x30Pad': {1: 0.005,
  2: 0.005,
  3: 0.005,
  4: 0.005,
  5: 0.005,
  6: 0.005,
  7: 0.005}
```

In [184]:

```
## Reading demand into a dictionary
## sparse metrix has been modelled for demand as 0 for items not mention
```

In [185]:
```
df_5 = pd.read_csv('demand.csv', index_col=0)
demand_dict = makeDict([R,T],df_5.to_numpy())
demand_dict
```

Out[185]:
```
{'PadFab': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'FlecFab': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetRod': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'PVCRod': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetLeg': {1: 4, 2: 0, 3: 8, 4: 6, 5: 0, 6: 0, 7: 0},
 'PVCLeg': {1: 2, 2: 0, 3: 0, 4: 6, 5: 0, 6: 0, 7: 4},
 'Package': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'MetalCut': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'PVCCut': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'FabCut': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'FabSew': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 'KitAssm': {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30M': {1: 0, 2: 0, 3: 2, 4: 0, 5: 0, 6: 1, 7: 0},
 '30x24M': {1: 0, 2: 2, 3: 0, 4: 1, 5: 0, 6: 0, 7: 1},
 '30x24PVC': {1: 1, 2: 3, 3: 4, 4: 2, 5: 1, 6: 6, 7: 8},
 '24x18PVC': {1: 0, 2: 2, 3: 0, 4: 3, 5: 0, 6: 4, 7: 6},
 '36x30Flec': {1: 0, 2: 0, 3: 1, 4: 1, 5: 2, 6: 0, 7: 0},
 '36x30Pad': {1: 0, 2: 1, 3: 0, 4: 1, 5: 0, 6: 0, 7: 1},
 '30x24Flec': {1: 2, 2: 0, 3: 1, 4: 1, 5: 0, 6: 2, 7: 0},
 '30x24Pad': {1: 0, 2: 0, 3: 2, 4: 0, 5: 0, 6: 1, 7: 0},
 '24x18Flec': {1: 1, 2: 1, 3: 0, 4: 0, 5: 2, 6: 0, 7: 0},
 '24x18Pad': {1: 0, 2: 1, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0},
 '36x30MPad': {1: 2, 2: 3, 3: 2, 4: 5, 5: 3, 6: 8, 7: 11},
 '36x30MFlec': {1: 0, 2: 2, 3: 3, 4: 6, 5: 0, 6: 8, 7: 12},
 '30x24MPad': {1: 1, 2: 0, 3: 2, 4: 2, 5: 3, 6: 24, 7: 14},
 '30x24MFlec': {1: 1, 2: 0, 3: 2, 4: 3, 5: 2, 6: 12, 7: 11},
 '30x24PVCPad': {1: 0, 2: 2, 3: 0, 4: 3, 5: 2, 6: 9, 7: 12},
 '30x24PVCFlec': {1: 0, 2: 0, 3: 2, 4: 2, 5: 0, 6: 3, 7: 5},
 '24x18PVCPad': {1: 2, 2: 3, 3: 2, 4: 1, 5: 0, 6: 24, 7: 23},
 '24x18PVCFlec': {1: 2, 2: 0, 3: 2, 4: 3, 5: 0, 6: 13, 7: 24}}
```

In [186]:
```
## Reading revenue into a dictionary
## sparse metrix has been modelled for revenue as 0 for items not mentio
```

In [187]:
```
df_6 = pd.read_csv('revenue.csv', index_col=0)
revenue_dict = makeDict([R,T],df_6.to_numpy())
revenue_dict
```

Out[187]:
```
{'PadFab': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'FlecFab': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'MetRod': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'PVCRod': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'MetLeg': {1: 2.96, 2: 2.96, 3: 2.96, 4: 2.96, 5: 2.96, 6: 2.96, 7: 2
.96},
 'PVCLeg': {1: 1.87, 2: 1.87, 3: 1.87, 4: 1.87, 5: 1.87, 6: 1.87, 7: 1
```

```
.87},
 'Package': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'MetalCut': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'PVCCut': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'FabCut': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'FabSew': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 'KitAssm': {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
 '36x30M': {1: 12.49,
  2: 12.49,
  3: 12.49,
  4: 12.49,
  5: 12.49,
  6: 12.49,
  7: 12.49},
 '30x24M': {1: 10.49,
  2: 10.49,
  3: 10.49,
  4: 10.49,
  5: 10.49,
  6: 10.49,
  7: 10.49},
 '30x24PVC': {1: 9.89, 2: 9.89, 3: 9.89, 4: 9.89, 5: 9.89, 6: 9.89, 7:
9.89},
 '24x18PVC': {1: 8.49, 2: 8.49, 3: 8.49, 4: 8.49, 5: 8.49, 6: 8.49, 7:
8.49},
 '36x30Flec': {1: 7.28, 2: 7.28, 3: 7.28, 4: 7.28, 5: 7.28, 6: 7.28, 7
: 7.28},
 '36x30Pad': {1: 7.68, 2: 7.68, 3: 7.68, 4: 7.68, 5: 7.68, 6: 7.68, 7:
7.68},
 '30x24Flec': {1: 5.99, 2: 5.99, 3: 5.99, 4: 5.99, 5: 5.99, 6: 5.99, 7
: 5.99},
 '30x24Pad': {1: 6.29, 2: 6.29, 3: 6.29, 4: 6.29, 5: 6.29, 6: 6.29, 7:
6.29},
 '24x18Flec': {1: 4.49, 2: 4.49, 3: 4.49, 4: 4.49, 5: 4.49, 6: 4.49, 7
: 4.49},
 '24x18Pad': {1: 5.22, 2: 5.22, 3: 5.22, 4: 5.22, 5: 5.22, 6: 5.22, 7:
5.22},
 '36x30MPad': {1: 38.27,
  2: 38.27,
  3: 38.27,
  4: 38.27,
  5: 38.27,
  6: 33.99,
  7: 33.99},
 '36x30MFlec': {1: 34.98,
  2: 34.98,
  3: 34.98,
  4: 34.98,
  5: 34.98,
  6: 28.99,
  7: 28.99},
 '30x24MPad': {1: 29.49,
  2: 29.49,
  3: 29.49,
  4: 29.49,
  5: 29.49,
  6: 24.99,
  7: 24.99},
 '30x24MFlec': {1: 26.24,
  2: 26.24,
  3: 26.24,
  4: 26.24,
  5: 26.24,
  6: 19.99,
  7: 19.99},
 '30x24PVCPad': {1: 24.98,
  2: 24.98,
  3: 24.98,
  4: 24.98,
  5: 24.98,
  6: 19.99,
  7: 19.99},
 '30x24PVCFlec': {1: 22.24,
  2: 22.24,
  3: 22.24,
  4: 22.24,
  5: 22.24,
  6: 18.92,
  7: 18.92},
 '24x18PVCPad': {1: 15.24,
  2: 15.24,
  3: 15.24,
  4: 15.24,
  5: 15.24,
  6: 11.24,
  7: 11.24},
 '24x18PVCFlec': {1: 14.98,
  2: 14.98,
  3: 14.98,
  4: 14.98,
  5: 14.98,
  6: 9.99,
  7: 9.99}}
```

In [188]:

```
penalty = 0.95
revenue_3D = np.zeros((30, 7, 7))
revenue = df_6.to_numpy()
for i in range(0,30):
    for j in range (0,7):
        for k in range (0,7):
            if(k<j):
                revenue_3D[i][j][k] = 0
            elif (k==j):
                revenue_3D[i][j][k] = revenue[i][j]
            elif (k>j):
                revenue_3D[i][j][k] = revenue[i][j]*pow(penalty, k-j)
```

In [189]:

```
revenue_3D_dict = makeDict([R,T,T],revenue_3D)
revenue_3D_dict
```

```
  7: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0}},
 'MetRod': {1: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0
},
  2: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  3: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  4: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  5: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  6: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  7: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0}},
 'PVCRod': {1: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0
},
  2: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  3: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  4: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  5: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  6: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0},
  7: {1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0, 6: 0.0, 7: 0.0}},
 'MetLeg': {1: {1: 2.96,
   2: 2.812,
   3: 2.6713999999999998,
```

In [190]:

```
## setting up LP Problem
```

In [191]:

```
prob = LpProblem("Multi period production scheduling", LpMaximize)
```

In [192]:

```
## defining LP Variables
```

In [193]:

```
sold = LpVariable.dicts("Product_sold", [(product,t,tau) for product in
produced = LpVariable.dicts("Product_produced", [(product,t) for product
scapped = LpVariable.dicts("Items_scrapped", [(product,t) for product in
inventory = LpVariable.dicts("Inventory", [(product,t) for product in R
```

In [194]:

```
## setting up objective function
```

In [197]:

```
prob += lpSum([sold[(product,t,tau)]*revenue_3D_dict[(product,t,tau)] fo
prob -= lpSum([scapped[(product,t)]*scrcost_dict[(product,t)] for produc
prob -= lpSum([inventory[(product,t)]*inv_dict[(product,t)] for product
```

```
---------------------------------------------------------------------------
-----
KeyError                                  Traceback (most recent call
last)
<ipython-input-197-337a68a1a47b> in <module>
----> 1 prob += lpSum([sold[(product,t,tau)]*revenue_3D_dict[(product,
t,tau)] for product in R for t in T for tau in T])
      2 prob -= lpSum([scapped[(product,t)]*scrcost_dict[(product,t)]
for product in R for t in T])
      3 prob -= lpSum([inventory[(product,t)]*inv_dict[(product,t)] fo
r product in R for t in T])

<ipython-input-197-337a68a1a47b> in <listcomp>(.0)
----> 1 prob += lpSum([sold[(product,t,tau)]*revenue_3D_dict[(product,
t,tau)] for product in R for t in T for tau in T])
      2 prob -= lpSum([scapped[(product,t)]*scrcost_dict[(product,t)]
for product in R for t in T])
      3 prob -= lpSum([inventory[(product,t)]*inv_dict[(product,t)] fo
r product in R for t in T])

KeyError: ('PadFab', 1, 1)
```

In [198]:

```
## setting up constraints
```

In [ ]:

```
for product in M:
    for t in T:
        prob += lpSum([produced[(product1,t)]*resources_bill_dict[(produ
            + scapped[(product,t) + inventory[(product,t)
            + lpSum[sold[(product,t,tau)] for product in R for t in T fo
            == supply_dict[(product,t)] + produced[(product1,t)] + inven
```

In [ ]:

```
for product in C:
    for t in T:
        prob += lpSum([produced[(product1,t)]*resources_bill_dict[(produ
                + scapped[(product,t) == supply_dict[(product,t)]
```

In [ ]:
```
for product in W:
    for t in T:
        prob += lpSum([produced[(product1,t)]*resources_bill_dict[(produ
            + scapped[(product,t) + inventory[(product,t)
            + lpSum[sold[(product,t,tau)] for product in R for t in T fo
            == supply_dict[(product,t)] + inventory[(product,t-1)
```

In [ ]:
```
for product in M:
    for t in T:

        prob += lpSum([sold[(product,t,tau)]*revenue_3D_dict[(product,t,
                <=lpSum[demand[(product,t,tau)] for tau in range(0,t)]
```

In [ ]:
```
prob.writeLP("Multi period production scheduling.lp")
```

In [ ]:
```
prob.solve()
```

In [100]:
```
# for i in M:
#     for t in T:
#         print(supply_dict[i][t])

```

In [ ]:

In [ ]: