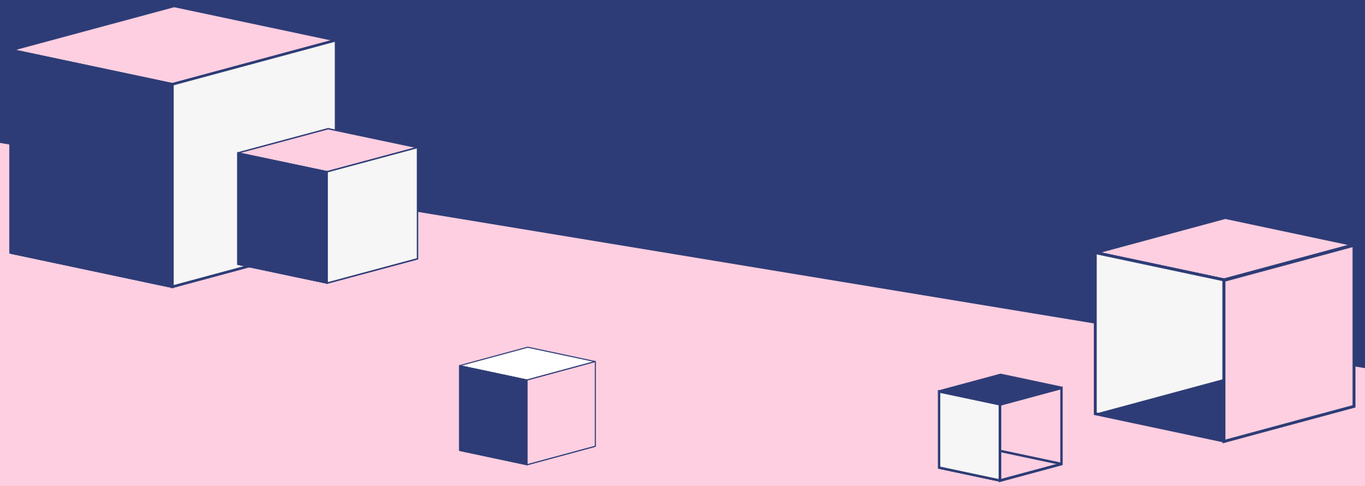
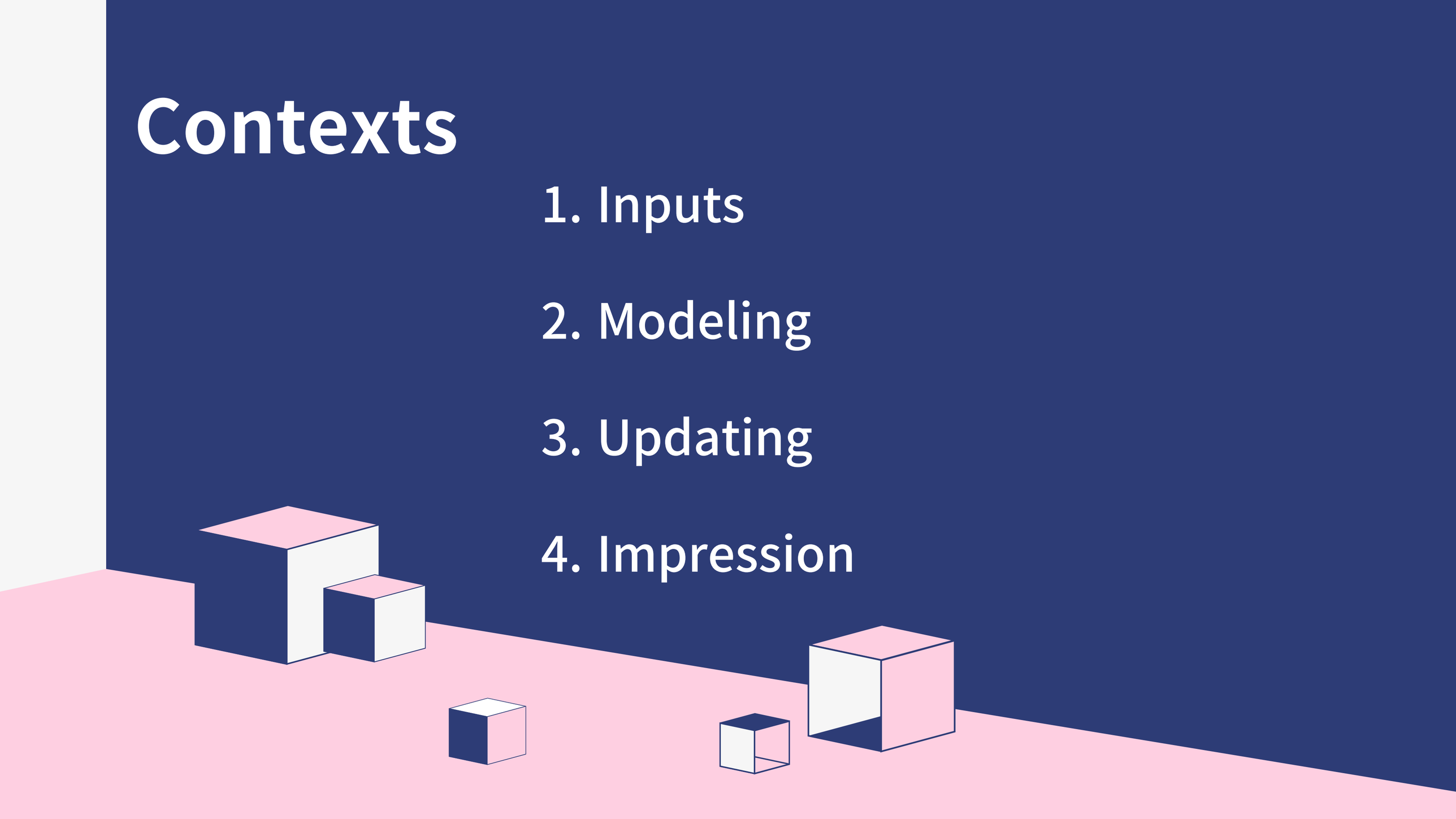
생산계획 기말 발표



산업경영공학부
2017170813 정종혁
2017170820 민재원
2017170827 이병주
2018170807 남이량

Contexts

1. Inputs
2. Modeling
3. Updating
4. Impression



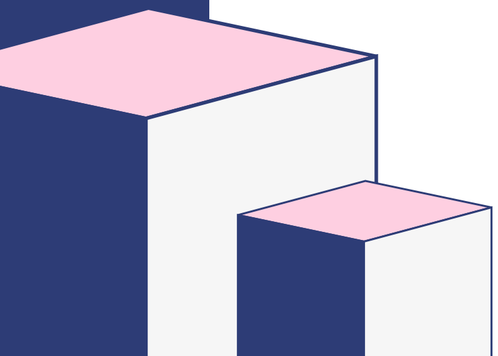
INPUTS

Problem Situation

1. MTO Process

2. MPS - Batch or Chasing

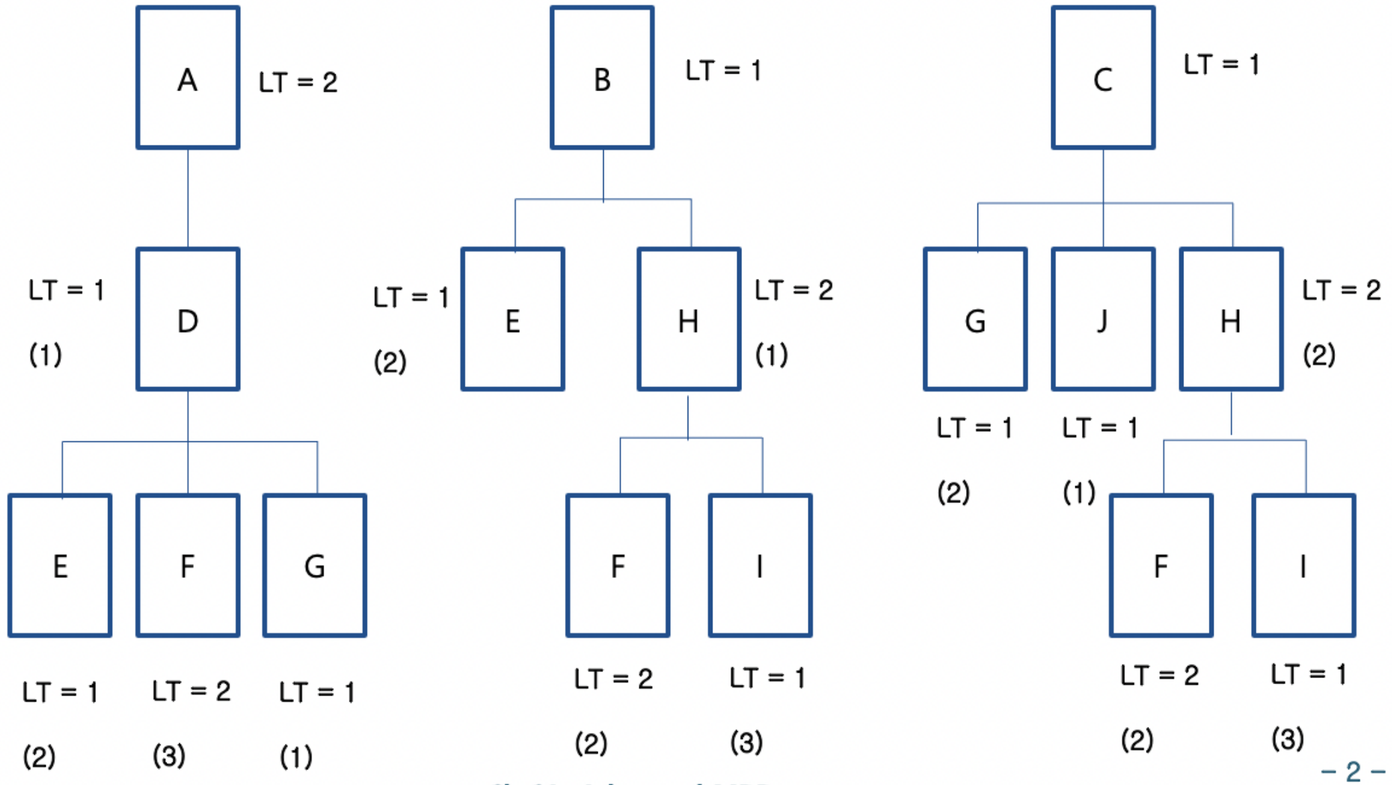
3. MRP - Batch or Lot for Lot



INPUTS

Problem Situation

- LT : Lead time / (#) : 개수



INPUTS

Problem Situation

예)		On-hand	Month 1	2	3	4	5	6	7	8	9	10	11	12
A	Forecasts	160	100	100	100	120	150	150	150	200	200	200	200	200
	Orders		120	80	50	30	20	10	0	0	0	0	0	0
B	Forecasts	200	150	150	180	180	180	180	230	230	230	230	230	230
	Orders		180	120	80	50	30	20	0	0	0	0	0	0
C	Forecasts	240	200	200	200	240	240	240	280	280	280	280	280	280
	Orders		200	180	120	80	40	10	0	0	0	0	0	0

각 item별 PIB

A : 160

B : 200

C : 240

D : 220

E : 300

F : 360

G : 280

H : 400

I : 350

J : 300

- MRP record for PIB Input값

예)	On-hand	Month 1	2	3	4	5	6	7	8	9	10	11	12
Gross requirement													
Scheduled receipts													
PIB	뒷장												
Net requirements													
Planned receipts													
Plan. order release													

INPUTS

Data form conversion

Data -> CSV or Excel
-> DataFrame -> Modeling

예)		Month									
		On-hand	1	2	3	4	5	6	7	8	9
A	Forecasts	160	100	100	100	120	150	150	150	200	200
	Orders		120	80	50	30	20	10	0	0	0
B	Forecasts	200	150	150	180	180	180	180	230	230	230
	Orders		180	120	80	50	30	20	0	0	0
C	Forecasts	240	200	200	200	240	240	240	280	280	280
	Orders		200	180	120	80	40	10	0	0	0

week	Forecast	Orders	Available(ending)	ATP	MPS
0			160		
1	100	100			
2	80	40			
3	70	20			
4	70	10			
5	120	0			
6	100	0			
7	120	0			
8	60	0			
9	70	0			
10	80	0			
11	110	0			
12	100	0			

```
df_a= pd.read_excel(path + '생계_input/a_input.xlsx')
df_b= pd.read_excel(path + '생계_input/b_input.xlsx')
df_c= pd.read_excel(path + '생계_input/c_input.xlsx')
MPS_a, MPS_b, MPS_c = MPS(df_a, 10, 120), MPS(df_b, 10, 120), MPS(df_c, 10, 120)
```

INPUTS

Data form conversion

Select Options(SS, Batch_size, Type of MPS, PIB)

예)		Month												
		On-hand	1	2	3	4	5	6	7	8	9	10	11	12
A	Forecasts	160	100	100	100	120	150	150	150	200	200	200	200	200
	Orders		120	80	50	30	20	10	0	0	0	0	0	0
B	Forecasts	200	150	150	180	180	180	180	230	230	230	230	230	230
	Orders		180	120	80	50	30	20	0	0	0	0	0	0
C	Forecasts	240	200	200	200	240	240	240	280	280	280	280	280	280
	Orders		200	180	120	80	40	10	0	0	0	0	0	0

각 item 별 PIB
A : 160, B : 200, C : 240, D : 220, E : 300,
F : 360, G : 280, H : 400, I : 250, J : 300

		Safety Stock	Batch	Chasing	Initial Inventory
2	A	10	120	N	160
3	B	10	120	N	200
4	C	10	120	N	240
5	D	0	0	Y	220
6	E	0	0	Y	300
7	F	0	0	Y	360
8	G	0	0	Y	280
9	H	0	0	Y	400
10	I	0	0	Y	350
11	J	0	0	Y	300

Modeling by Python

MPS Modeling - Chasing

```python
def chasing_MPS(data):

    mps_df = pd.DataFrame(columns = ['week','Forecast','Order','Available(ending)','ATP','MPS'])
    mps_df['week'] = data['week']
    mps_df['Forecast']=data['Forecast']
    mps_df['Order'] = data['Orders']
    mps_df['Available(ending)'] = data['Available(ending)']

    for i in range(1,13):

        mps_df['Available(ending)'][i] = mps_df['Available(ending)'][i-1] - max(mps_df['Forecast'][i],
                                                                                mps_df['Order'][i])

        if mps_df['Available(ending)'][i]<0:
            mps_df['Available(ending)'][i] = 0
        else:
            pass

        mps_df['MPS'][i] = mps_df['Forecast'][i] - mps_df['Available(ending)'][i-1]

        if mps_df['MPS'][i] < 0:
            mps_df['MPS'][i] = 0
        else:
            pass

        mps_df['ATP'][i] = mps_df['MPS'][i] - mps_df['Order'][i]

        if mps_df['ATP'][i] < 0:
            mps_df['ATP'][i] = 0
        else:
            pass

    return mps_df
```

Modeling by Python

MPS Modeling - Batch

```
def MPS(data, safety_stock, lot_size=0):
    data = data.fillna(0)
    max_lot = max (max (data["Forecast"]), max(data["Orders"]) )
    if lot_size == 0 :
        lot_size = max_lot
    for i in range(0,len(data)-1):
        data["Available(ending)"][i+1]=data["Available(ending)"][i]-max(data["Orders"][i+1],data["Forecast"][i+1])
        if data["Available(ending)"][i+1] <  safety_stock:
            data["MPS"][i+1]= lot_size
        else:
            data["MPS"][i+1]=0
        data["Available(ending)"][i+1]=data["Available(ending)"][i]-max(data["Orders"][i+1],
                                                                       data["Forecast"][i+1])+data["MPS"][i+1]


    for i in range(1,len(data)):
        if i==1:
            if data["MPS"][i]==0:

                data["ATP"][i]=data["Available(ending)"][0]
            else:
                a=data["Orders"][i]
                for j in range(i+1,len(data)-1):
                    if data["MPS"][j] ==0:
                        a += data["Orders"][j]
                    else:
                        break

                data["ATP"][i]=data["Available(ending)"][0] + data["MPS"][i] -a
        else:
            if data["MPS"][i]==0:
                data["ATP"][i]=0
            else:
                a=data["Orders"][i]
                for j in range(i+1,len(data)):
                    if data["MPS"][j] ==0:
                        a += data["Orders"][j]
                    else:
                        break

                data["ATP"][i]= data["MPS"][i] -a

    return data
```

Modeling by Python

MRP Modeling - Level 0 product A,B,C

Choose the type of MPS method
-> Running MPS -> MPS = MRP's Gross requirement

MRP of A, B, C

```
def mrp_abc(MPS, lead_time) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    #lead time 설정
    mrp["Gross Requirement"] = MPS["MPS"]
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if i != 0 :
            if idx - lead_time <= 0 :
                mrp["Scheduled Receipts"][idx] = i
            else :
                mrp["Planned Receipts"][idx] = i
                mrp["Planned Order Release"][idx - lead_time] = i
    return mrp
```

Modeling by Python

MRP Modeling - Level 1 product without duplication D, J

D , J's MRP's Gross requirement = Level 0's Gross requirement * Required number of parts

```
def mrp_dj(MRP_a, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_a["Planned Order Release"]

    mrp["PIB"][0] = inventory

    lead_time = 1
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

MRP Modeling - E

```
def mrp_e(MRP_a, MRP_b, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_a["Planned Order Release"] * 2 + MRP_b["Planned Order Release"] * 2
    mrp["PIB"][0] = inventory
    lead_time = 1
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

MRP Modeling - F

MRP of F

```
def mrp_f(MRP_d, MRP_h, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_d["Planned Order Release"] * 3 + MRP_h["Planned Order Release"] * 2
    mrp["PIB"][0] = inventory
    lead_time = 2
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

MRP Modeling - G

MRP of G

```
def mrp_g(MRP_d, MRP_c, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_d["Planned Order Release"] + MRP_c["Planned Order Release"] * 2
    mrp["PIB"][0] = inventory
    lead_time = 1
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

MRP Modeling - H

MRP of H

```
def mrp_h(MRP_b, MRP_c, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_b["Planned Order Release"] + MRP_c["Planned Order Release"] * 2
    mrp["PIB"][0] = inventory
    lead_time = 2
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

MRP Modeling - I

MRP of I

```
def mrp_i(MRP_h, inventory,  safety_stock = 0, batch = 0) :
    mrp = pd.read_excel(input_file)
    mrp = mrp.fillna(0)
    mrp["Gross Requirement"] = MRP_h["Planned Order Release"] * 3
    mrp["PIB"][0] = inventory
    lead_time = 1
    for idx, i in enumerate(mrp["Gross Requirement"]) :
        if idx == 0 :
            continue
        if i == 0 :
            mrp["PIB"][idx] = mrp["PIB"][idx-1]
        else :
            if mrp["PIB"][idx-1] - i >= safety_stock :
                mrp["PIB"][idx] = mrp["PIB"][idx-1] - i
            else :
                if idx - lead_time <= 0 :
                    mrp["Scheduled Receipts"][idx] = i - mrp["PIB"][idx]
                if batch != 0 :
                    mrp["PIB"][idx] = mrp["PIB"][idx-1] + batch - i
                    mrp["Planned Receipts"][idx] = batch
                else :
                    mrp["PIB"][idx] = 0
                mrp["Net Requirements"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Receipts"][idx] = i - mrp["PIB"][idx-1]
                mrp["Planned Order Release"][idx - lead_time] = mrp["Planned Receipts"][idx]
    return mrp
```

Modeling by Python

Printing - Put in the path of dataset

```
def print_all(path) :
    info_data = pd.read_excel(path + '/data_input.xlsx')
    info = info_data.rename(index={0: 'A', 1: 'B', 2: 'C', 3 : 'D', 4 : 'E', 5: 'F', 6: 'G', 7: 'H', 8:'I' ,9:'J'} )

    df_a= pd.read_excel(path + '/a_input.xlsx')
    df_b= pd.read_excel(path + '/b_input.xlsx')
    df_c= pd.read_excel(path + '/c_input.xlsx')
    input_file = path + "/d_input.xlsx"
    if info["Chasing"]["A"] == "N" :
        MPS_a = MPS(df_a, info["Safety Stock"]["A"], lot_size = info["Batch"]["A"])
        MPS_b = MPS(df_b, info["Safety Stock"]["B"], lot_size = info["Batch"]["B"])
        MPS_c = MPS(df_c, info["Safety Stock"]["C"], lot_size = info["Batch"]["C"])
    else :
        MPS_a = MPS(df_a)
        MPS_b = MPS(df_b)
        MPS_c = MPS(df_c)

    print(color.BOLD + "MPS of A" + color.END)
    print(tabulate(MPS_a, headers='keys', tablefmt='pretty', showindex=False))

    print(color.BOLD + "MPS of B" + color.END)
    print(tabulate(MPS_b, headers='keys', tablefmt='pretty', showindex=False))

    print(color.BOLD + "MPS of C" + color.END)
    print(tabulate(MPS_c, headers='keys', tablefmt='pretty', showindex=False))

    input_file = path + "/d_input.xlsx"
```

Modeling by Python

check the available order

주어진 MPS 상황에서 최대로 받을 수 있는 Order 수 산출

In [66]:
```
def order_okay(MPS, order, week) :
    cum_ATP = 0
    i = 0
    while i <= week :
        cum_ATP += MPS["ATP"][i]
        i+=1
    if cum_ATP >= order :
        print(color.BOLD + "Order can be accepted." +color.END)
        print("Availabe Order is " + str(cum_ATP) + " 입니다.")
    else :
        print(color.BOLD + "Order can not be accepted." +color.END)
        print("Availabe Order is " + str(cum_ATP) + " 입니다.")
```

In [67]:
```
order_okay(MPS_a, 300, 4)
```

Order can not be accepted.
Availabe Order is 230.0 입니다.

Modeling by Python

Result

Modeling process

Updating

Updating

Order Changes or Batch Size Changes
-> Need MPS, MRP changes

		Safety Stock	Batch	Chasing	Initial Inventory
2	A	10	120	N	160
3	B	10	120	N	200
4	C	10	120	N	240
5	D	0	0	Y	220
6	E	0	0	Y	300
7	F	0	0	Y	360
8	G	0	0	Y	280
9	H	0	0	Y	400
10	I	0	0	Y	350
11	J	0	0	Y	300

→

	A	B	C	D	E
1		Safety Stock	Batch	Chasing	Initial Inventory
2	A	10	200	N	160
3	B	10	0	Y	200
4	C	10	280	N	240
5	D	0	250	N	220
6	E	0	0	Y	300
7	F	0	0	Y	360
8	G	0	**650**	N	280
9	H	0	0	Y	400
10	I	0	0	Y	350
11	J	0	0	Y	300
12					
13					

Updating

Updating

EXCEL's Result
=
PYTHON's Result

Impression

IE Challenge

1. Connectivity

2. BOM Complexity

3. Importance of Forecast

THANK YOU.