Basic Settings

In [1]:
```
# !pip install matplotlib
# !pip install forallpeople
```

In [2]:
```
import csv
from dataclasses import dataclass, field, asdict, astuple
from typing import List
from collections import namedtuple
from functools import reduce
from functools import partial
# import forallpeople as si ## 단위 변환 관련 패키지(아직 미사용)
import math
import matplotlib.pyplot as plt
```

In [3]:
```
##################################Module For Functional Programing####################
curry = lambda f: lambda a,*args: f(a, *args) if (len(args)) else lambda *args: f(

filter = curry(filter)
map = curry(map)

def _take(length, iter):
    res = []
    for a in iter:
        res.append(a)
        if len(res) == length:
            return res

take = curry(_take)
reduce = curry(reduce)

go = lambda *args: reduce(lambda a,f: f(a), args) ## 함수도 축약 가능 ##
#######################################################################################
```

In [4]:
```
## 함수형 프로그래밍 코드 사용 예제

add = curry(lambda a,b: a + b)

example = lambda _list: goW
(
    _list,
    filter(lambda a: not a % 2),
    filter(lambda a: not a % 4),
    map(lambda a: a * a + 5),
    reduce(add)
)
```

In [5]:
```
example([1,2,3,4,5,6])
```

Out[5]: 21

In [6]:
```
## 결과 출력용 함수 ##
def showResult(res):
    if res[0] > res[1]: ineqSign = '>'
```

```
    else: ineqSign = '<'

    return f"nominal: '{res[0]:.2f}' kN·m {ineqSign} required: '{res[1]:.2f}' kN·m
```

Definition Code Part

In [7]:
```
## 사용 예시

######## 사용자 입력부 ########
input_path = 'Section Profile.csv'

input_dsgnMode = "LRFD"

input_fy = 344.738 ## 50
input_E = 199900 ## 29000

input_DL = 6.567
input_LL = 10.945
input_length = 10670

input_cb_mode = "Cb고려"
input_table_mode = "continuous"
input_brace_idx = 2

##############################
```

0. 단면 정보 import (선언부)

In [8]:
```
def importCSV(_path):
    with open(_path,'r') as f:
        reader  = csv.reader(f) # csv의 행별로 읽어옴
        res = [x for x in reader]

    return res
```

In [9]:
```
def exportCSV(_filename, _datas):
    f = open(_filename, 'w', newline='') # 자동줄바꿈 방지 header 이후 첫행 빈행 생성
    csv_writer = csv.writer(f)

    for x in _datas:
        csv_writer.writerow(x)
    f.close()
```

In [10]:
```
dfSectionProfile = importCSV(input_path)
dfHeader = dfSectionProfile.pop(0) #pop(0)의 의미 첫행을 값을 반환하고 나머지 리스트는
```

1. 단면 자료형 Setter (선언부)

In [11]:
```
def makeSectionForm(args):

    Form = namedtuple('SectionForm', ['ID','shape','h','bf','tw','tf','k'])
```

```
    form = W
        Form(ID=args[0], shape=args[1],
            h=float(args[2]), bf=float(args[3]),
            tw=float(args[4]), tf=float(args[5]), k=float(args[6]))

    return form


### 예시 ###
sForm = makeSectionForm(dfSectionProfile[294])
```

2. 단면 속성 Setter (선언부)

In [12]:

```
def setSectionProp(sForm):
    def defineC():
        if _shape == "W" or _shape == "H": return 1
        elif _shape == "C": return (_h0/2) * (_Iy/_Cw)**0.5
        else: return 1 ## 임시 대처

    Prop = namedtuple(
        'SectionProperty',
        ['ID', 'h', 'bf', 'tw', 'tf', 'k', 'shape', 'Area', 'Weight',
         'Ix', 'Sx', 'Zx', 'rx', 'Iy', 'Sy', 'Zy', 'ry',
         'Cw', 'J', 'rts', 'h0', 'C'])

    (_ID,_h,_bf,_tw,_tf,_k,_shape) = (sForm.ID,sForm.h,sForm.bf,sForm.tw,sForm.tf,sFo
    _Area = 2*_tf*_bf+(_h-2*_tf)*_tw
    _Weight = _Area*77.22/10**6
    _Ix = (2*(_bf*_tf**3/12+_bf*_tf*((_h-2*_tf)/2+_tf/2)**2)+_tw*(_h-2*_tf)**3/12)
    _Sx = _Ix/(_h/2)
    _Zx = _bf*_tf*(_h-_tf)+0.25*(_h-2*_tf)**2*_tw
    _rx = (_Ix/_Area)**0.5
    _Iy = 2*(_tf*(_bf)**3/12)+(_h-2*_tf)*(_tw)**3/12
    _Sy = _Iy/(_bf/2)
    _Zy = 0.5*(_bf)**2*_tf+0.25*(_h-2*_tf)*(_tw)**2
    _ry = (_Iy/_Area)**0.5
    _Cw = (_h-_tf)**2*_bf**3*_tf/24
    (_J,_rts,_h0,_C) = ((2*_bf*_tf**3+(_h-_tf)*_tw**3)/3, ((_Iy*_Cw)**0.5/_Sx)**0.5

    prop = Prop(
        ID= _ID, h= _h, bf= _bf, tw= _tw, tf= _tf, k= _k, shape= _shape, Area= _Area,
        Ix= _Ix, Sx= _Sx, Zx= _Zx, rx= _rx, Iy= _Iy, Sy= _Sy, Zy= _Zy, ry= _ry,
        Cw= _Cw, J= _J, rts= _rts, h0= _h0, C= _C)

    return prop

### 예시 ###
aa = setSectionProp(sForm)
aa
```

Out[12]:

```
SectionProperty(ID='H-248×124x5x8', h=248.0, bf=124.0, tw=5.0, tf=8.0, k=12.0, shape
='H', Area=3144.0, Weight=0.24277968, Ix=33783168.0, Sx=272444.9032258064, Zx=305360.
0, rx=103.65945418893585, Iy=2544582.0, Sy=41041.645161290326, Zy=62954.0, ry=28.44899
681618542, Cw=36607180800.0, J=52325.333333333336, rts=33.470048360084256, h0=240.0, C
=1)
```

3. 재료 속성 Setter (선언부)

In [13]:
```
def setMaterialProp(_fy, _E):
    Prop = namedtuple('MaterialProperty', ['fy', 'E'])
    prop = Prop(fy=_fy, E=_E)

    return prop

### 예시 ###
mProp = setMaterialProp(344.738, 199900)
```

4. 디자인 베이스 Setter (선언부)

In [14]:
```
def setDesignBase(_DL, _LL, _length):
    Base = namedtuple('DesignBase', ['DL', 'LL', 'length'])
    base = Base(DL=_DL, LL=_LL, length=_length)

    return base

### 예시 ###
dBase = setDesignBase(input_DL, input_LL, input_length)
dBase
```

Out[14]: DesignBase(DL=6.567, LL=10.945, length=10670)

5. 서브 디자인 베이스 Setter (선언부)

In [15]:
```
def setFlexureBase(cb_mode, table_mode, _brace_idx):
    Base = namedtuple('SubBase_flx', ['brace_idx', 'Cb'])

    def findCb():
        _none = {
            "1p": [[1.32]],
            "2p": [[1.14]],
            "3p": [[1.14]],
            "continuous": [[1.14]] }

        _atLoad = {
            "1p": [1.67, 1.67],
            "2p": [1.67, 1.00, 1.67],
            "3p": [1.67, 1.11, 1.11, 1.67],
            "continuous":
                [[1.30, 1.30],
                [1.45, 1.01, 1.45],
                [1.52, 1.06, 1.06, 1.52],
                [1.56, 1.12, 1.00, 1.12, 1.56]] }
        if cb_mode == "Cb고려":
            if _brace_idx == 0:
                result = _none[table_mode][0]#[_brace_idx]
            else:
                if table_mode == "continuous":
                    result = _atLoad[table_mode][_brace_idx-1]
                else:
```

```
                    result = _atLoad[table_mode]
        elif cb_mode == "Cb미고려":
            result = [ 1.00 ]

        return result

    base = Base(brace_idx= _brace_idx, Cb=findCb())
    return base

### 예시 ###
fBase = setFlexureBase("Cb고려", "continuous", 2)
fBase
```

Out[15]: SubBase_flx(brace_idx=2, Cb=[1.45, 1.01, 1.45])

In [16]:
```
def setCompressureBase(cb_mode, table_mode, _brace_idx):
    pass
```

In [17]:
```
def setTensileBase(cb_mode, table_mode, _brace_idx):
    pass
```

6. 디자인 결과 Checker (선언부)

In [18]:
```
def checkDesignResult(_dsgnMode, _sProp, _mProp, _dBase, _subBaseColl, _subCheckColl):

    def calcRequired(_dsgnMode):
        def _calcStr():
            if _dsgnMode == "LRFD":
                result = 1.2*_dBase.DL + 1.6*_dBase.LL
            elif _dsgnMode == "ASD":
                result = _dBase.DL + _dBase.LL
            else:
                result = "check the DesignMode"
            return result

        def _calcMoment():
            result = ((_calcStr() * _dBase.length**2) / 8) / 1000**2
            return result

        return (_calcStr(), _calcMoment())

    (fCheck, fBase) = (_subCheckColl.fCheck, _subBaseColl.fBase)
#     (cCheck, cBase) = (_subCheckColl.cCheck, _subBaseColl.cBase)
#     (tCheck, tBase) = (_subCheckColl.tCheck, _subBaseColl.tBase)

    def chkDesignResult(_dsgnMode, _sProp, _mProp, _dBase, _subBase, _subCheck):
        NomStr = _subCheck(_dsgnMode,_sProp, _mProp, _dBase, _subBase)

        if NomStr > calcRequired(_dsgnMode)[1]:
            return (NomStr, calcRequired(_dsgnMode)[1], "O.K.")
        else:
            return (NomStr, calcRequired(_dsgnMode)[1], "N.G.")

    result_flx = chkDesignResult(_dsgnMode, _sProp, _mProp, _dBase, fBase,fCheck)
#     result_comp = chkDesignResult(_dsgnMode, _sProp, _mProp, _dBase, cBase,cCheck)
#     result_tension = chkDesignResult(_dsgnMode, _sProp, _mProp, _dBase, tBase,tCheck

    return result_flx
```

6.a 서브 디자인 결과 Checker_휨 (선언부)

In [19]:

```
def checkFlexure(_dsgnMode,_sProp, _mProp, _dBase, _subBase):

    def findLp():
        return 1.76 * _sProp.ry * (_mProp.E/_mProp.fy)**0.5

    def findLr():
        return 1.95*_sProp.rts*_mProp.E/(0.7*_mProp.fy)*(_sProp.J/(_sProp.Sx*(_sPro

    def findMp():
        ### for Strong Axis ###
        Mp_x = _mProp.fy * _sProp.Zx
        ### for Weak Axis ###
        Mp_y = min([(_mProp.fy * _sProp.Zy), (1.6*_mProp.fy*_sProp.Sy)])

        return {"Mp_x": Mp_x, "Mp_y": Mp_y}

    def findMn(): ### for Strong Axis ###
        (Mp,Lp,Lr,Lb) = (findMp()["Mp_x"], findLp(), findLr(), _dBase.length/(_subBas

        if Lb <= Lp:
            Mn = Mp
        elif Lp < Lb <= Lr:
            Mn = min(map(lambda x: x * (Mp-(Mp-0.7*_mProp.fy*_sProp.Sx)*((Lb-Lp) /
        elif Lb > Lr:
            Fcr = min(map(lambda x: (x * (math.pi**2 * _mProp.E)/((Lb/_sProp.rts)**
            Mn = min((Fcr)*_sProp.Sx, Mp)

        return Mn / 1000**2


    def calcNominal():
        if _dsgnMode == "LRFD":
            result = 0.90 * findMn()
        elif _dsgnMode == "ASD":
            result = findMn() / 1.67
        else:
            result = "check the DesignMode"

        return result

    return calcNominal()
```

6.b 서브 디자인 결과 Checker_압축 (선언부)

In [20]:

```
def checkCompressure(_dsgnMode,_sProp, _mProp, _dBase, _subBase):
    pass
```

6.c 서브 디자인 결과 Checker_인장 (선언부)

In [21]:

```
def checkTensile(_dsgnMode,_sProp, _mProp, _dBase, _subBase):
    pass
```

6.d 서브 디자인 결과 Checker_복합력 (선언부)

In [22]:
```
def checkCombined(_dsgnMode,_sProp, _mProp, _dBase, _subBase):
    pass
```

Client Code Part

[[단일 부재 검토 Mode]]

Given:

Select a W-shape beam for span and uniform dead and live loads as shown in Figure F.1-1A. Limit the member to a maximum nominal depth of 18 in. Limit the live load deflection to *L*/360. The beam is simply supported and continuously braced. The beam is ASTM A992 material.

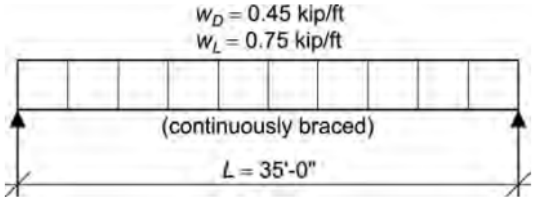


Fig. F.1-1A. Beam loading and bracing diagram.

Solution:

From AISC *Manual* Table 2-4, the material properties are as follows:

ASTM A992
F_y = 50 ksi
F_u = 65 ksi

------- 사용자 입력부 -------

In [23]:
```
## 사용 예시

######## 사용자 입력부 ########
input_path = 'Section Profile.csv'

input_dsgnMode = "LRFD"

input_fy = 344.738 ## 50
input_E = 199900 ## 29000

input_DL = 6.567
input_LL = 10.945
input_length = 10670

input_cb_mode = "Cb고려"
input_table_mode = "continuous"
input_brace_idx = 2

##############################
```

---

0. 단면 정보 import (사용부)

In [24]:
```
dfSectionProfile = importCSV(input_path)
dfSectionProfile.remove(dfSectionProfile[0])
# dfSectionProfile
```

1. 단면 자료형 Setter (사용부)

In [25]:
```
targetSection = makeSectionForm(dfSectionProfile[160])
```

2. 단면 속성 Setter (사용부)

In [26]:
```
sProp = setSectionProp(targetSection)
```

3. 재료 속성 Setter (사용부)

In [27]:
```
mProp = setMaterialProp(input_fy, input_E)
```

4.디자인 베이스 Setter (사용부)

In [28]:
```
dBase = setDesignBase(input_DL, input_LL, input_length)
```

5. 서브 디자인 베이스 Setter (사용부)

In [29]:
```
mkSubBaseColl = namedtuple('subBaseColl', 'fBase, cBase, tBase')

subBase_flx = setFlexureBase(input_cb_mode, input_table_mode, input_brace_idx)
# subBase_comp = setCompressureBase(input_cb_mode, input_table_mode, input_brace_idx)
# subBase_tens = setTensileBase(input_cb_mode, input_table_mode, input_brace_idx)
subBaseColl = mkSubBaseColl(fBase=subBase_flx, cBase=0, tBase=0)
```

6.a, b, c. 서브 디자인 결과 Checker (사용부)

In [30]:
```
mkSubCheckColl = namedtuple('subCheckColl', 'fCheck, cCheck, tCheck')

subCheck_flx = checkFlexure
# subCheck_comp = checkCompressure
# subCheck_tens = checkTensile
subCheckColl = mkSubCheckColl(fCheck=subCheck_flx, cCheck=0, tCheck=0)
```

6. 디자인 결과 Checker (사용부)

In [31]:
```
res = checkDesignResult(input_dsgnMode, sProp, mProp, dBase, subBaseColl, subCheckCol
showResult(res)
```

Out[31]:
```
"nominal: '411.69' kN · m > required: '361.36' kN · m -> O.K."
```

[[다중 부재 검토 Mode]]

------- 사용자 입력부 -------

In [40]:
```
## 사용 예시

####### 사용자 입력부 #######
input_path = 'Section Profile.csv'

input_dsgnMode = "LRFD"

input_fy = 344.738 ## 50
input_E = 199900 ## 29000

input_DL = 6.567
input_LL = 10.945
input_length = 10670

input_cb_mode = "Cb고려"
input_table_mode = "continuous"
input_brace_idx = 2

##############################
```

---

0 ~ 1. 다중 단면 자료형 세팅

In [33]:
```
## 여기서 makeSectionForms는 데이터를 저장하지 않고 함수 객체를 저장한다. C#, JAVA에서
makeSectionForms = lambda _list: goW
(
    _list,
    map(lambda x: makeSectionForm(x)),
    list
)
```

In [34]:
```
#### 0. 단면 정보 import
dfSectionProfile = importCSV(input_path)
dfSectionProfile.remove(dfSectionProfile[0])

#### 1. 단면 자료형 setter
targetSections = makeSectionForms(dfSectionProfile)
```

2 ~ 6. 각종 Setter들 및 Checker들 고정 조건 세팅

In [35]:
```
### 고정 조건 세팅

    #### 3. 재료속성
mProp = setMaterialProp(input_fy, input_E)

    #### 4. 디자인 베이스 setter
dBase = setDesignBase(input_DL, input_LL, input_length)

    #### 5. 서브 디자인 베이스 setter
mkSubBaseColl = namedtuple('subBaseColl', 'fBase, cBase, tBase')

subBase_flx = setFlexureBase(input_cb_mode, input_table_mode, input_brace_idx)
# subBase_comp = setCompressureBase(input_cb_mode, input_table_mode, input_brace_idx)
# subBase_tens = setTensileBase(input_cb_mode, input_table_mode, input_brace_idx)
subBaseColl = mkSubBaseColl(fBase=subBase_flx, cBase=0, tBase=0)

    #### 6.a~d 서브디자인 checker
```

```
mkSubCheckColl = namedtuple('subCheckColl', 'fCheck, cCheck, tCheck')
subCheck_flx = checkFlexure
# subCheck_comp = checkCompressure
# subCheck_tens = checkTensile
subCheckColl = mkSubCheckColl(fCheck=subCheck_flx, cCheck=0, tCheck=0)
```

디자인 결과 Checker 용 합성함수 제조

In [36]:
```
### 가변 조건 검토를 위한 함수 합성

multiChecker1 = lambda _list: goW
(
    _list,
    map(lambda x: makeSectionForm(x)),
    map(lambda x: setSectionProp(x)),
    map(lambda x: checkDesignResult(input_dsgnMode, x, mProp, dBase, subBaseColl, sub
    filter(lambda x: x[2] == "O.K." ),
    map(lambda x: showResult(x)),
    list
)
```

In [37]:
```
multiChecker1(targetSections)
```

Out[37]:
```
["nominal: '8219.64' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '7147.49' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6409.00' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '5559.92' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '15562.29' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '13899.82' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '11690.61' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '9844.68' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '9052.46' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '8426.50' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '8197.37' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '7312.72' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6617.07' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6253.83' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '5604.58' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '4807.17' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '4338.14' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '8465.59' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6987.44' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6865.53' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6121.66' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '5745.82' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '5408.93' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '4834.44' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '4291.96' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '3625.50' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '3202.63' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '2693.97' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '20820.23' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '19743.77' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '18493.91' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '16528.54' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '14715.63' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '11737.87' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '10722.04' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '9692.81' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '8598.32' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '7783.39' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '7105.77' kN·m > required: '361.36' kN·m -> O.K.",
 "nominal: '6461.65' kN·m > required: '361.36' kN·m -> O.K.",
```

"nominal: '6010.11' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5547.56' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5176.66' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4838.16' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5074.55' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4537.17' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4022.06' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3671.81' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3436.65' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3177.88' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2956.43' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2757.11' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2383.30' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '7902.28' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '7190.01' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '6419.29' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5822.63' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5237.26' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4760.62' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4318.25' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3886.81' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3000.62' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2665.37' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2410.51' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2171.64' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1919.83' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '7332.75' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '6662.12' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '6034.63' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5383.74' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4766.07' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4281.40' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3787.16' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3399.74' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3057.83' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2344.72' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2009.53' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1872.66' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1724.97' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1558.25' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1397.86' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1245.73' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '9606.37' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '6305.42' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5729.12' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5168.58' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4737.91' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4315.67' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3902.72' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3570.46' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3202.27' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2885.26' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2599.73' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2351.57' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1818.53' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1574.05' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1381.82' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1243.61' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '1087.21' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5826.20' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '5198.86' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4693.18' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '4252.15' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3778.91' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3422.62' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '3068.78' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2845.38' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2574.90' kN・m > required: '361.36' kN・m -> O.K.",
"nominal: '2365.82' kN・m > required: '361.36' kN・m -> O.K.",

"nominal: '2094.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1858.70' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1631.29' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1442.66' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1254.73' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1124.12' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '983.20' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '866.28' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '752.55' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '552.42' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '469.39' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3770.32' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3363.53' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3004.83' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2679.58' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2407.23' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2176.38' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1875.14' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1663.49' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1533.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1375.76' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1256.17' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '968.95' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '847.88' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '735.65' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '674.91' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '601.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '515.88' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '423.96' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '467.18' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '380.22' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3825.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3442.31' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3114.07' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2797.50' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2492.96' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2242.06' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2000.39' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1777.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1597.68' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1448.61' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1299.09' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1128.94' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1028.79' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '906.83' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '784.76' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '623.87' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '565.69' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '512.05' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '459.13' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '411.69' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '966.54' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '851.42' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '721.67' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '615.95' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '437.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '376.00' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '10319.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '9297.83' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '8450.62' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '7535.90' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '6698.96' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '5948.68' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '5316.86' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '4742.60' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '4421.07' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '4079.50' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3739.26' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3411.06' kN · m > required: '361.36' kN · m -> O.K.",

"nominal: '3039.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2737.61' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2470.91' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2198.71' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1968.25' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1799.75' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1620.93' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1452.33' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1317.33' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1182.28' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1071.04' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '961.43' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '876.19' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '783.61' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '668.94' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '608.22' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '544.99' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '485.91' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '386.36' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3059.01' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2713.84' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2445.15' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2166.18' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1971.72' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1776.32' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1601.09' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1400.82' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1238.14' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1083.86' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '943.34' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '826.99' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '746.77' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '661.68' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '598.57' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '538.41' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '481.82' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '414.90' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '371.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '744.04' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '647.55' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '561.13' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '483.44' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '419.76' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '365.95' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '378.22' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '416.36' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '451.19' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '483.79' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '515.51' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '632.04' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '668.24' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '396.68' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '558.27' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '642.14' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '697.22' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '773.55' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '840.06' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '394.07' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '561.90' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '646.32' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '846.96' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '945.10' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1029.39' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1116.99' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1216.28' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1305.39' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1537.04' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1935.77' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2937.59' kN · m > required: '361.36' kN · m -> O.K.",

```
"nominal: '4463.02' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '417.04' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '692.35' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '830.84' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '463.30' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '536.84' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '644.32' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '799.40' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '939.12' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '620.89' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '711.44' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '842.26' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '978.01' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1125.93' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1294.84' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1513.51' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1603.13' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1871.72' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2211.46' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2069.64' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2377.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2776.28' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2508.84' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2994.15' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3633.72' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3960.26' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '4639.54' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '10285.50' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2500.42' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '7727.13' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '4697.91' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '5936.56' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '1855.48' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '9062.11' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '5531.36' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '3551.98' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '2160.21' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '11632.88' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '6980.97' kN · m > required: '361.36' kN · m -> O.K.",
"nominal: '8875.29' kN · m > required: '361.36' kN · m -> O.K."]
```

In [38]:

```
### 가변 조건 검토를 위한 함수 합성

multiChecker2 = lambda _list: goW
(
    _list,
    map(lambda x: makeSectionForm(x)),
    map(lambda x: setSectionProp(x)),
    map(lambda x: checkDesignResult(input_dsgnMode, x, mProp, dBase, subBaseColl, sub
    filter(lambda x: x[2] == "N.G." ),
    map(lambda x: showResult(x)),
    list
)
```

In [39]:

```
multiChecker2(targetSections)
```

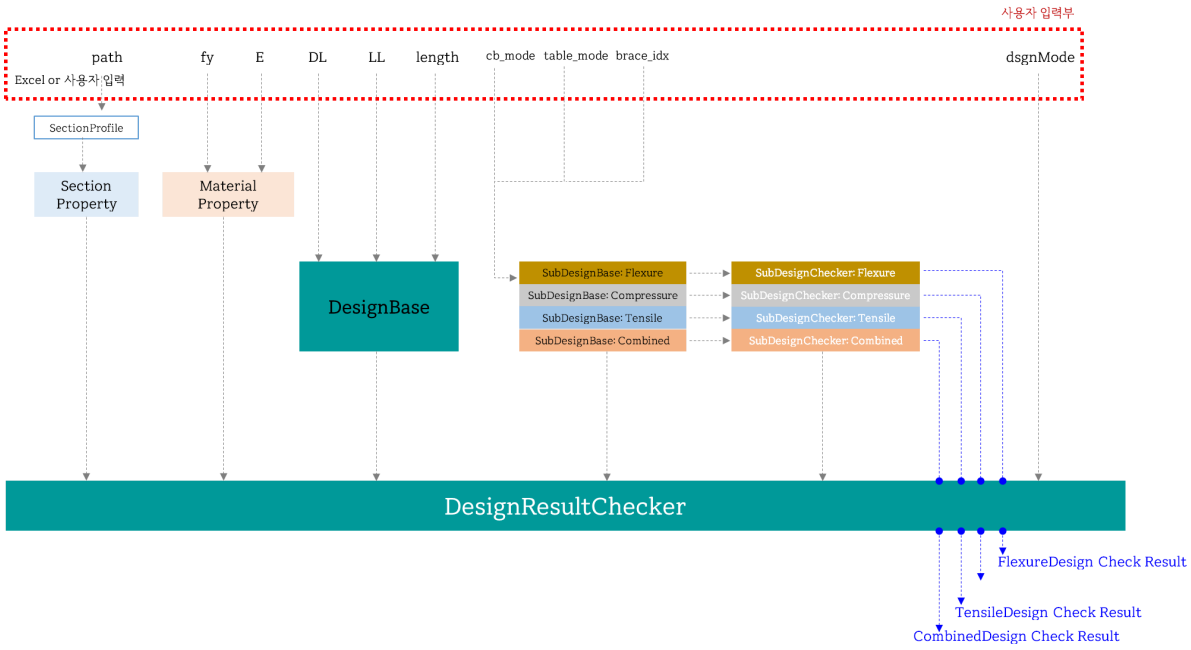
Out[39]:

```
["nominal: '317.25' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '321.62' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '268.13' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '215.08' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '329.72' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '286.34' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '246.09' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '169.33' kN · m < required: '361.36' kN · m -> N.G.",
 "nominal: '126.99' kN · m < required: '361.36' kN · m -> N.G.",
```

"nominal: '343.20' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '301.36' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '243.87' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '212.43' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '178.03' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '115.26' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '83.98' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '325.98' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '286.90' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '250.04' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '207.18' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '169.05' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '142.41' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '61.99' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '47.01' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '32.16' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '26.00' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '323.81' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '292.23' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '253.46' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '212.77' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '170.30' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '145.03' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '118.93' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '94.04' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '50.45' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '39.37' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '30.26' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '20.76' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '342.85' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '290.30' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '234.53' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '187.17' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '161.25' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '138.30' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '117.00' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '97.09' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '76.22' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '59.74' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '33.28' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '24.79' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '16.44' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '83.76' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '63.18' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '42.74' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '41.22' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '24.40' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '14.83' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '12.95' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '50.01' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '39.71' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '25.17' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '20.34' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '38.04' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '11.74' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '31.57' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '63.90' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '23.55' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '30.37' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '72.56' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '146.30' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '158.34' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '204.21' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '56.50' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '68.85' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '139.40' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '229.78' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '255.73' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '279.73' kN · m < required: '361.36' kN · m -> N.G.",

```
"nominal: '302.75' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '97.00' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '113.27' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '222.37' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '264.82' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '163.42' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '204.37' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '247.62' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '336.34' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '279.51' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '335.67' kN · m < required: '361.36' kN · m -> N.G.",
"nominal: '352.63' kN · m < required: '361.36' kN · m -> N.G."]
```

Code Structure Diagram

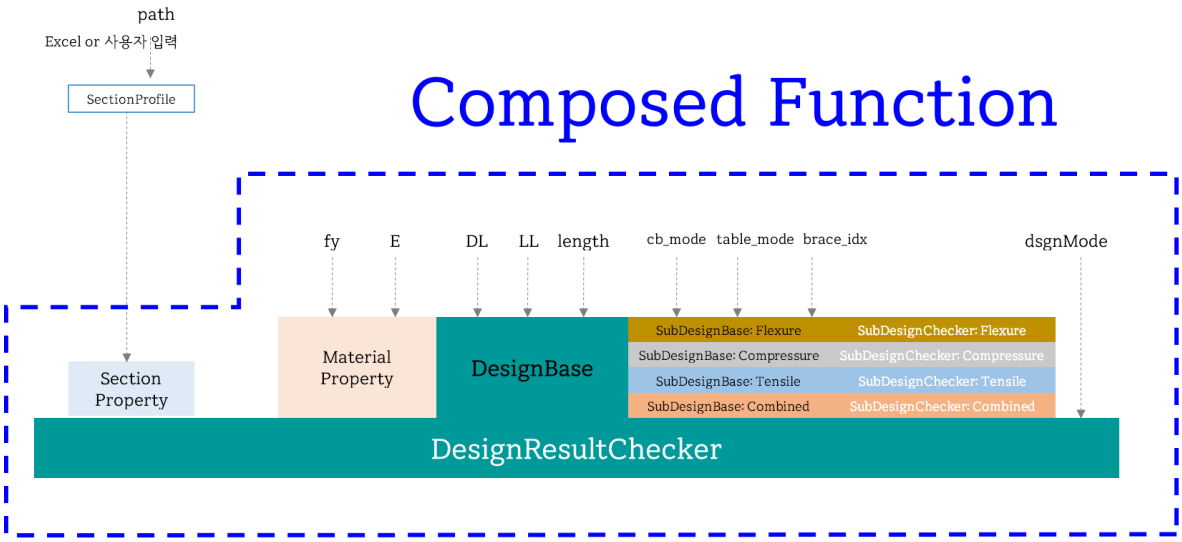


$$(g \circ f)(x) = g(f(x))$$

Data와 모델링을 추상화 하는 것 보다,

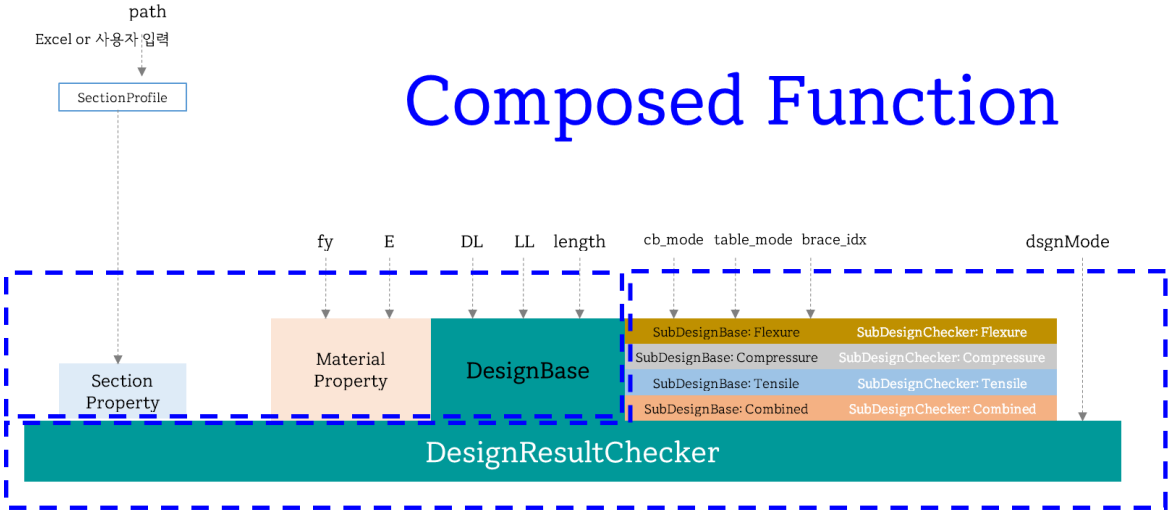
연산을 추상화하는 것이

복잡성을 다루는 데 더 좋다



하나의 연산으로 추상화된 12가지의 연산(함수)

※추상화 범위는 얼마든지 변경이 가능하다.



※추상화 범위는 얼마든지 변경이 가능하다.

In [ ]: