[2-2. Index 값 설정]

1. 기간 인덱스 작성

1-1. 적용 기간 중 월 데이터 리스트 작성 함수

- 모델시작일 2023-12-01, 모델종료일 2027-01-31 일 때 [2023-12-31, 2024-01-31, ... , 2026-12-31, 2027-01-31]과 같이 해당 기간 중 월 마지막날 기준으로 월별 데이터의 리스트를 구하고자 함.

In [ ]:
```python
from datetime import datetime, timedelta

def end_of_month(dt):
    # 다음 달의 첫날
    next_month = dt.replace(day=28) + timedelta(days=4)
    # 이번 달의 마지막 날
    return next_month - timedelta(days=next_month.day)

def list_end_of_months(start_date, end_date):
    end_of_months = []

    current_date = end_of_month(start_date)
    while current_date <= end_date:
        end_of_months.append(current_date)
        next_month = current_date.replace(day=28) + timedelta(days=4)
        current_date = end_of_month(next_month)

    return end_of_months
```

In [ ]:
```python
list_end_of_months(datetime(2023, 12, 1), datetime(2027, 1, 31))
```

```
Out[ ]: [datetime.datetime(2023, 12, 31, 0, 0),
         datetime.datetime(2024, 1, 31, 0, 0),
         datetime.datetime(2024, 2, 29, 0, 0),
         datetime.datetime(2024, 3, 31, 0, 0),
         datetime.datetime(2024, 4, 30, 0, 0),
         datetime.datetime(2024, 5, 31, 0, 0),
         datetime.datetime(2024, 6, 30, 0, 0),
         datetime.datetime(2024, 7, 31, 0, 0),
         datetime.datetime(2024, 8, 31, 0, 0),
         datetime.datetime(2024, 9, 30, 0, 0),
         datetime.datetime(2024, 10, 31, 0, 0),
         datetime.datetime(2024, 11, 30, 0, 0),
         datetime.datetime(2024, 12, 31, 0, 0),
         datetime.datetime(2025, 1, 31, 0, 0),
         datetime.datetime(2025, 2, 28, 0, 0),
         datetime.datetime(2025, 3, 31, 0, 0),
         datetime.datetime(2025, 4, 30, 0, 0),
         datetime.datetime(2025, 5, 31, 0, 0),
         datetime.datetime(2025, 6, 30, 0, 0),
         datetime.datetime(2025, 7, 31, 0, 0),
         datetime.datetime(2025, 8, 31, 0, 0),
         datetime.datetime(2025, 9, 30, 0, 0),
         datetime.datetime(2025, 10, 31, 0, 0),
         datetime.datetime(2025, 11, 30, 0, 0),
         datetime.datetime(2025, 12, 31, 0, 0),
         datetime.datetime(2026, 1, 31, 0, 0),
         datetime.datetime(2026, 2, 28, 0, 0),
         datetime.datetime(2026, 3, 31, 0, 0),
         datetime.datetime(2026, 4, 30, 0, 0),
         datetime.datetime(2026, 5, 31, 0, 0),
         datetime.datetime(2026, 6, 30, 0, 0),
         datetime.datetime(2026, 7, 31, 0, 0),
         datetime.datetime(2026, 8, 31, 0, 0),
         datetime.datetime(2026, 9, 30, 0, 0),
         datetime.datetime(2026, 10, 31, 0, 0),
```

```
 datetime.datetime(2026, 11, 30, 0, 0),
 datetime.datetime(2026, 12, 31, 0, 0),
 datetime.datetime(2027, 1, 31, 0, 0)]
```

1-2. 기간 인덱스 작성 코드

In [ ]:

```python
from m01_assumption import assumption

index = {}

index["model"] = list_end_of_months(
    assumption['period_assumptions']['기본기간가정']['모델시작일'],
    assumption['period_assumptions']['기본기간가정']['모델종료일']
)

index["operating"] = list_end_of_months(
    assumption['period_assumptions']['기본기간가정']['운영시작일'],
    assumption['period_assumptions']['기본기간가정']['운영종료일']
)

index["이자지급"] = list_end_of_months(
    assumption['period_assumptions']['자금조달일정']['이자지급시작일'],
    assumption['period_assumptions']['자금조달일정']['이자지급종료일']
)

index["원금상환"] = list_end_of_months(
    assumption['period_assumptions']['자금조달일정']['원금상환시작일'],
    assumption['period_assumptions']['자금조달일정']['원금상환종료일']
)

index["수선TypeA"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeA'],
    assumption['facility_cost']['수선종료일']['TypeA']
)
```

```
index["수선TypeB"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeB'],
    assumption['facility_cost']['수선종료일']['TypeB']
)

index["수선TypeC"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeC'],
    assumption['facility_cost']['수선종료일']['TypeC']
)
```

In [ ]:
```
index
```

Out[ ]:
```
{'model': [datetime.datetime(2023, 12, 31, 0, 0),
  datetime.datetime(2024, 1, 31, 0, 0),
  datetime.datetime(2024, 2, 29, 0, 0),
  datetime.datetime(2024, 3, 31, 0, 0),
  datetime.datetime(2024, 4, 30, 0, 0),
  datetime.datetime(2024, 5, 31, 0, 0),
  datetime.datetime(2024, 6, 30, 0, 0),
  datetime.datetime(2024, 7, 31, 0, 0),
  datetime.datetime(2024, 8, 31, 0, 0),
  datetime.datetime(2024, 9, 30, 0, 0),
  datetime.datetime(2024, 10, 31, 0, 0),
  datetime.datetime(2024, 11, 30, 0, 0),
  datetime.datetime(2024, 12, 31, 0, 0),
  datetime.datetime(2025, 1, 31, 0, 0),
  datetime.datetime(2025, 2, 28, 0, 0),
  datetime.datetime(2025, 3, 31, 0, 0),
  datetime.datetime(2025, 4, 30, 0, 0),
  datetime.datetime(2025, 5, 31, 0, 0),
  datetime.datetime(2025, 6, 30, 0, 0),
  datetime.datetime(2025, 7, 31, 0, 0),
  datetime.datetime(2025, 8, 31, 0, 0),
  datetime.datetime(2025, 9, 30, 0, 0),
```

```
 datetime.datetime(2025, 10, 31, 0, 0),
 datetime.datetime(2025, 11, 30, 0, 0),
 datetime.datetime(2025, 12, 31, 0, 0),
 datetime.datetime(2026, 1, 31, 0, 0),
 datetime.datetime(2026, 2, 28, 0, 0),
 datetime.datetime(2026, 3, 31, 0, 0),
 datetime.datetime(2026, 4, 30, 0, 0),
 datetime.datetime(2026, 5, 31, 0, 0),
 datetime.datetime(2026, 6, 30, 0, 0),
 datetime.datetime(2026, 7, 31, 0, 0),
 datetime.datetime(2026, 8, 31, 0, 0),
 datetime.datetime(2026, 9, 30, 0, 0),
 datetime.datetime(2026, 10, 31, 0, 0),
 datetime.datetime(2026, 11, 30, 0, 0),
 datetime.datetime(2026, 12, 31, 0, 0),
 datetime.datetime(2027, 1, 31, 0, 0)],
'operating': [datetime.datetime(2024, 1, 31, 0, 0),
 datetime.datetime(2024, 2, 29, 0, 0),
 datetime.datetime(2024, 3, 31, 0, 0),
 datetime.datetime(2024, 4, 30, 0, 0),
 datetime.datetime(2024, 5, 31, 0, 0),
 datetime.datetime(2024, 6, 30, 0, 0),
 datetime.datetime(2024, 7, 31, 0, 0),
 datetime.datetime(2024, 8, 31, 0, 0),
 datetime.datetime(2024, 9, 30, 0, 0),
 datetime.datetime(2024, 10, 31, 0, 0),
 datetime.datetime(2024, 11, 30, 0, 0),
 datetime.datetime(2024, 12, 31, 0, 0),
 datetime.datetime(2025, 1, 31, 0, 0),
 datetime.datetime(2025, 2, 28, 0, 0),
 datetime.datetime(2025, 3, 31, 0, 0),
 datetime.datetime(2025, 4, 30, 0, 0),
 datetime.datetime(2025, 5, 31, 0, 0),
 datetime.datetime(2025, 6, 30, 0, 0),
 datetime.datetime(2025, 7, 31, 0, 0),
```

```
 datetime.datetime(2025, 8, 31, 0, 0),
 datetime.datetime(2025, 9, 30, 0, 0),
 datetime.datetime(2025, 10, 31, 0, 0),
 datetime.datetime(2025, 11, 30, 0, 0),
 datetime.datetime(2025, 12, 31, 0, 0),
 datetime.datetime(2026, 1, 31, 0, 0),
 datetime.datetime(2026, 2, 28, 0, 0),
 datetime.datetime(2026, 3, 31, 0, 0),
 datetime.datetime(2026, 4, 30, 0, 0),
 datetime.datetime(2026, 5, 31, 0, 0),
 datetime.datetime(2026, 6, 30, 0, 0),
 datetime.datetime(2026, 7, 31, 0, 0),
 datetime.datetime(2026, 8, 31, 0, 0),
 datetime.datetime(2026, 9, 30, 0, 0),
 datetime.datetime(2026, 10, 31, 0, 0),
 datetime.datetime(2026, 11, 30, 0, 0),
 datetime.datetime(2026, 12, 31, 0, 0)],
'이자지급': [datetime.datetime(2024, 1, 31, 0, 0),
 datetime.datetime(2024, 2, 29, 0, 0),
 datetime.datetime(2024, 3, 31, 0, 0),
 datetime.datetime(2024, 4, 30, 0, 0),
 datetime.datetime(2024, 5, 31, 0, 0),
 datetime.datetime(2024, 6, 30, 0, 0),
 datetime.datetime(2024, 7, 31, 0, 0),
 datetime.datetime(2024, 8, 31, 0, 0),
 datetime.datetime(2024, 9, 30, 0, 0),
 datetime.datetime(2024, 10, 31, 0, 0),
 datetime.datetime(2024, 11, 30, 0, 0),
 datetime.datetime(2024, 12, 31, 0, 0),
 datetime.datetime(2025, 1, 31, 0, 0),
 datetime.datetime(2025, 2, 28, 0, 0),
 datetime.datetime(2025, 3, 31, 0, 0),
 datetime.datetime(2025, 4, 30, 0, 0),
 datetime.datetime(2025, 5, 31, 0, 0),
 datetime.datetime(2025, 6, 30, 0, 0),
```

```
 datetime.datetime(2025, 7, 31, 0, 0),
 datetime.datetime(2025, 8, 31, 0, 0),
 datetime.datetime(2025, 9, 30, 0, 0),
 datetime.datetime(2025, 10, 31, 0, 0),
 datetime.datetime(2025, 11, 30, 0, 0),
 datetime.datetime(2025, 12, 31, 0, 0),
 datetime.datetime(2026, 1, 31, 0, 0),
 datetime.datetime(2026, 2, 28, 0, 0),
 datetime.datetime(2026, 3, 31, 0, 0),
 datetime.datetime(2026, 4, 30, 0, 0),
 datetime.datetime(2026, 5, 31, 0, 0),
 datetime.datetime(2026, 6, 30, 0, 0),
 datetime.datetime(2026, 7, 31, 0, 0),
 datetime.datetime(2026, 8, 31, 0, 0),
 datetime.datetime(2026, 9, 30, 0, 0),
 datetime.datetime(2026, 10, 31, 0, 0),
 datetime.datetime(2026, 11, 30, 0, 0),
 datetime.datetime(2026, 12, 31, 0, 0)],
'원금상환': [datetime.datetime(2024, 1, 31, 0, 0),
 datetime.datetime(2024, 2, 29, 0, 0),
 datetime.datetime(2024, 3, 31, 0, 0),
 datetime.datetime(2024, 4, 30, 0, 0),
 datetime.datetime(2024, 5, 31, 0, 0),
 datetime.datetime(2024, 6, 30, 0, 0),
 datetime.datetime(2024, 7, 31, 0, 0),
 datetime.datetime(2024, 8, 31, 0, 0),
 datetime.datetime(2024, 9, 30, 0, 0),
 datetime.datetime(2024, 10, 31, 0, 0),
 datetime.datetime(2024, 11, 30, 0, 0),
 datetime.datetime(2024, 12, 31, 0, 0),
 datetime.datetime(2025, 1, 31, 0, 0),
 datetime.datetime(2025, 2, 28, 0, 0),
 datetime.datetime(2025, 3, 31, 0, 0),
 datetime.datetime(2025, 4, 30, 0, 0),
 datetime.datetime(2025, 5, 31, 0, 0),
```

```
 datetime.datetime(2025, 6, 30, 0, 0),
 datetime.datetime(2025, 7, 31, 0, 0),
 datetime.datetime(2025, 8, 31, 0, 0),
 datetime.datetime(2025, 9, 30, 0, 0),
 datetime.datetime(2025, 10, 31, 0, 0),
 datetime.datetime(2025, 11, 30, 0, 0),
 datetime.datetime(2025, 12, 31, 0, 0),
 datetime.datetime(2026, 1, 31, 0, 0),
 datetime.datetime(2026, 2, 28, 0, 0),
 datetime.datetime(2026, 3, 31, 0, 0),
 datetime.datetime(2026, 4, 30, 0, 0),
 datetime.datetime(2026, 5, 31, 0, 0),
 datetime.datetime(2026, 6, 30, 0, 0),
 datetime.datetime(2026, 7, 31, 0, 0),
 datetime.datetime(2026, 8, 31, 0, 0),
 datetime.datetime(2026, 9, 30, 0, 0),
 datetime.datetime(2026, 10, 31, 0, 0),
 datetime.datetime(2026, 11, 30, 0, 0),
 datetime.datetime(2026, 12, 31, 0, 0)],
'수선TypeA': [datetime.datetime(2025, 3, 31, 0, 0)],
'수선TypeB': [datetime.datetime(2025, 4, 30, 0, 0)],
'수선TypeC': [datetime.datetime(2025, 5, 31, 0, 0)]}
```

2. 연간인상률 인덱스 작성

2-1. 연간 인상률 계산 함수 작성

- 기준연도 대비 적용 연도에 반영되는 인상률을 연 인상률에 해당 기간 연도 수만큼 제곱하여 계산

In [ ]:
```python
def calculate_inflation_rate(year, base_year, rate):
    return ((1 + rate) ** max(year - base_year, 0))
```

In [ ]:
```
print(f'2024년: {calculate_inflation_rate(2024, 2024, 0.05) * 100}')
print(f'2025년: {calculate_inflation_rate(2025, 2024, 0.05) * 100}')
print(f'2026년: {calculate_inflation_rate(2026, 2024, 0.05) * 100}')
```

```
2024년: 100.0
2025년: 105.0
2026년: 110.25
```

2-2. 연간 인상률 계산

In [ ]:
```
import pandas as pd

pd.set_option('display.max_rows', 30)
pd.set_option('display.max_columns', 100)
pd.set_option('display.max_colwidth', 20)
pd.set_option('display.width', 300)

# DataFrame의 출력을 확장하여 한 줄로 계속 출력되도록 설정
pd.set_option('display.expand_frame_repr', True)
```

In [ ]:
```
base_year = assumption['period_assumptions']['기본기간가정']['운영시작일'].year

data = []
for dt in index['model']:
    dct = {}
    dct['판매단가'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['판매단가']
    )
    dct['운영비'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['운영비']
    )
    dct['인건비'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['인건비']
    )
    data.append(dct)
index["연간인상률"] = pd.DataFrame(data, index=index['model'])
```

In [ ]:
```
index['연간인상률']
```

Out[ ]:

	판매단가	운영비	인건비
2023-12-31	1.000000	1.000000	1.000000
2024-01-31	1.000000	1.000000	1.000000
2024-02-29	1.000000	1.000000	1.000000
2024-03-31	1.000000	1.000000	1.000000
2024-04-30	1.000000	1.000000	1.000000
...	...	...	...
2026-09-30	1.102500	1.060900	1.102500
2026-10-31	1.102500	1.060900	1.102500
2026-11-30	1.102500	1.060900	1.102500
2026-12-31	1.102500	1.060900	1.102500
2027-01-31	1.157625	1.092727	1.157625

38 rows × 3 columns

3. 기간별 계산일수 계산

In [ ]:

```
data = []
for dt in index['model']:
    dct = {}
    dct['월간일수'] = (dt - dt.replace(day=1) + timedelta(days=1)).days
    dct['연간일수'] = 366 if dt.year % 4 == 0 else 365
    data.append(dct)
index["days"] = pd.DataFrame(data, index=index['model'])
```

In [ ]:
```
index['days']
```

Out[ ]:

	월간일수	연간일수
2023-12-31	31	365
2024-01-31	31	366
2024-02-29	29	366
2024-03-31	31	366
2024-04-30	30	366
...	...	...
2026-09-30	30	365
2026-10-31	31	365
2026-11-30	30	365
2026-12-31	31	365
2027-01-31	31	365

38 rows × 2 columns

4. general_function.py 파일 작성

- 작성된 코드 중 공통으로 사용 가능한 함수들은 별도로 모아서 general_function.py 파일에 작성

In [ ]:
```python
# m00_general_function.py

from datetime import datetime, timedelta

def end_of_month(dt):
    # 다음 달의 첫날
    next_month = dt.replace(day=28) + timedelta(days=4)
    # 이번 달의 마지막 날
    return next_month - timedelta(days=next_month.day)

def list_end_of_months(start_date, end_date):
    end_of_months = []

    current_date = end_of_month(start_date)
    while current_date <= end_date:
        end_of_months.append(current_date)
        next_month = current_date.replace(day=28) + timedelta(days=4)
        current_date = end_of_month(next_month)

    return end_of_months

def calculate_inflation_rate(year, base_year, rate):
    return ((1 + rate) ** max(year - base_year, 0))
```

5. index.py 파일 작성

- index 설정 코드들을 index.py 파일에 작성
- 필요한 assumption 값 및 general_function은 해당 파일에서 import하여 사용

In [ ]:
```python
# m02_index.py

```

```python
import pandas as pd
from datetime import timedelta

from m00_general_function import (
    list_end_of_months,
    calculate_inflation_rate,
)
from m01_assumption import assumption


#### 1. 기간인덱스
index = {}

index["model"] = list_end_of_months(
    assumption['period_assumptions']['기본기간가정']['모델시작일'],
    assumption['period_assumptions']['기본기간가정']['모델종료일']
)

index["operating"] = list_end_of_months(
    assumption['period_assumptions']['기본기간가정']['운영시작일'],
    assumption['period_assumptions']['기본기간가정']['운영종료일']
)

index["이자지급"] = list_end_of_months(
    assumption['period_assumptions']['자금조달일정']['이자지급시작일'],
    assumption['period_assumptions']['자금조달일정']['이자지급종료일']
)

index["원금상환"] = list_end_of_months(
    assumption['period_assumptions']['자금조달일정']['원금상환시작일'],
    assumption['period_assumptions']['자금조달일정']['원금상환종료일']
)

index["수선TypeA"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeA'],
```

```
    assumption['facility_cost']['수선종료일']['TypeA']
)

index["수선TypeB"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeB'],
    assumption['facility_cost']['수선종료일']['TypeB']
)

index["수선TypeC"] = list_end_of_months(
    assumption['facility_cost']['수선시작일']['TypeC'],
    assumption['facility_cost']['수선종료일']['TypeC']
)


#### 2. 연간인상률 인덱스
base_year = assumption['period_assumptions']['기본기간가정']['운영시작일'].year

data = []
for dt in index['model']:
    dct = {}
    dct['판매단가'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['판매단가']
    )
    dct['운영비'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['운영비']
    )
    dct['인건비'] = calculate_inflation_rate(
        dt.year,
        base_year,
        assumption['period_assumptions']['연간인상률']['인건비']
    )
```

```
    data.append(dct)
index["연간인상률"] = pd.DataFrame(data, index=index['model'])


#### 3. 기간별 계산일수
data = []
for dt in index['model']:
    dct = {}
    dct['월간일수'] = (dt - dt.replace(day=1) + timedelta(days=1)).days
    dct['연간일수'] = 366 if dt.year % 4 == 0 else 365
    data.append(dct)
index["days"] = pd.DataFrame(data, index=index['model'])
```