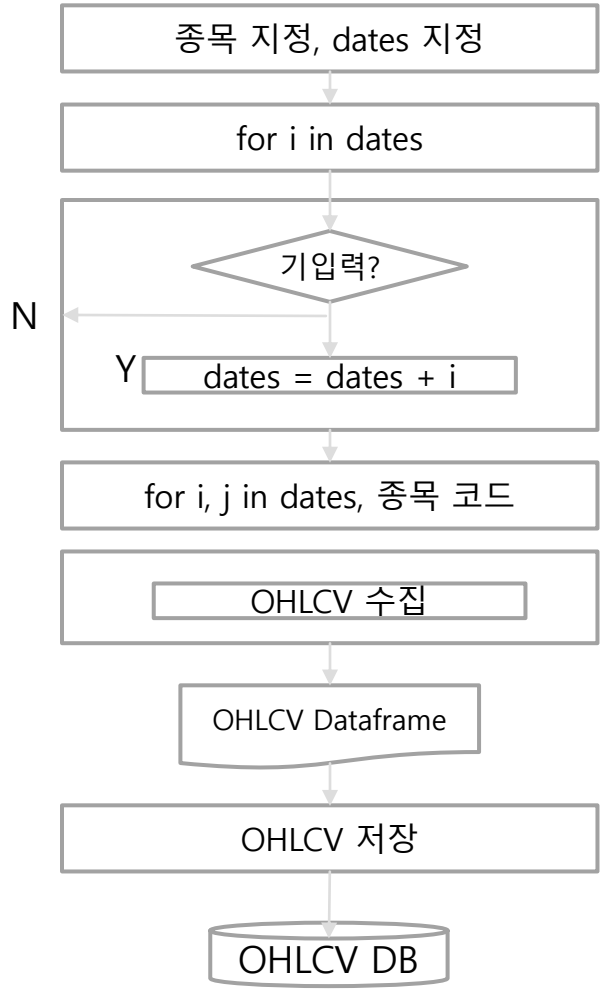
소프트웨어특강1
텀 프로젝트

1. 주가 데이터 수집 모듈

1. 주가 데이터 수집 모듈

2) 원시주가 데이터 수집



- KRX 사이트를 이용하여 원시주가 데이터를 가져옵니다.
- 원시주가 테이블을 조회하여 저장된 가장 마지막 일자를 불러온다.
- 마지막일자로부터 현재일자까지의 데이터를 조회한다.

1. 주가 데이터 수집 모듈

1) 종목 수집

	Symbol	Market	Name	Sector	Industry	ListingDate	SettleMonth	Re...	HomePage	Region
1	060310	KOSDAQ	3S	전자부품 제조업	반도체 웨이퍼 캐리어	2002-04-23	03월	김세완	http://www.3sref.com	서울특별시
2	095570	KOSPI	AJ네트웍스	산업용 기계 및 ?	렌탈(파렛트, OA장비, 건설?	2015-08-21	12월	박대현, 손삼	http://www.ajnet.co.kr	서울특별시
3	006840	KOSPI	AK홀딩스	기타 금융업	지주사업	1999-08-11	12월	채형석, 이석	http://www.aekyunggroup.co.kr	서울특별시
4	054620	KOSDAQ	APS홀딩스	기타 금융업	인터넷 트래픽 솔루션	2001-12-04	12월	정기로	http://www.apsholdings.co.kr	경기도
5	265520	KOSDAQ	AP시스템	특수 목적용 기계	디스플레이 제조 장비	2017-04-07	12월	김영주	http://www.apsystems.co.kr	경기도
6	211270	KOSDAQ	AP위성	통신 및 방송 장?	위성통신 단말기	2016-03-04	12월	류장수	http://www.apsi.co.kr	서울특별시
7	027410	KOSPI	BGF	기타 금융업	지주회사	2014-05-19	12월	홍정국	http://www.bgf.co.kr	서울특별시
8	282330	KOSPI	BGF리테일	종합 소매업	체인화 편의점	2017-12-08	12월	이건준	http://www.bgfretail.com	서울특별시
9	032790	KOSDAQ	BNGT	기계장비 및 관련	Bio 이종장기 사업, ICT 프	1997-06-26	12월	조상환	http://www.mgenplus.com	서울특별시
10	138930	KOSPI	BNK금융지주	기타 금융업	금융지주회사	2011-03-30	12월	김지완	http://www.bnkfg.com	부산광역시
11	001460	KOSPI	BYC	봉제의복 제조업	메리야스,란제리 제조,도매/?	1975-06-02	12월	김 대 환	http://home.byc.co.kr	서울특별시
12	013720	KOSDAQ	CBI	자동차 신품 부품	엔진부품	1993-12-29	12월	오경원	http://www.cheongbo.co.kr	인천광역시
13	001040	KOSPI	CJ	기타 금융업	지주회사	1973-06-29	12월	손경식, 김홍	http://www.cj.net	서울특별시
14	079160	KOSPI	CJ CGV	영화, 비디오물,	영화상영,영화관 운영	2004-12-24	12월	허민회	http://www.cgv.co.kr	서울특별시
15	035760	KOSDAQ	CJ ENM	텔레비전 방송업	의류, 생활주방, 가전제품,	1999-11-23	12월	Ho Sung K…	http://www.cjmall.com	서울특별시
16	311690	KOSDAQ	CJ 바이오사이	자연과학 및 공학	생명정보 플랫폼, 마이크로바	2019-12-26	12월	천종식	http://www.cjbioscience.com	서울특별시
17	000120	KOSPI	CJ대한통운	도로 화물 운송업	Contract Logistics, 포…	1956-07-02	12월	강신호,민영희	http://www.cjlogistics.com	서울특별시
18	011150	KOSPI	CJ씨푸드	기타 식품 제조업	수산물(어묵,맛살)가공품 도매	1988-11-26	12월	이인덕, 김정	http://www.cjseafood.net	경기도
19	097950	KOSPI	CJ제일제당	기타 식품 제조업	설탕,소맥분,조미식품,육가공	2007-09-28	12월	손경식, 최은	http://www.cj.co.kr	서울특별시
20	051500	KOSDAQ	CJ프레시웨이	음·식료품 및 담?	식자재유통, 단체급식	2001-07-26	12월	정성필	http://www.cjfreshway.com	경기도

- FinanceDataReader 을 이용하여 상장폐지 되지 않은 종목의 정보를 받아온다.
- 종목 데이터는 사용을 위해 DB에 저장한다.

1. 주가 데이터 수집 모듈

2) 원시주가 데이터 수집

	종목코드	종목명	시장구분	날짜	등락률	시가	고가	저가	종가	거래량	거래대금	시가총액	상장주식수
1	000020	동화약품	KOSPI	2016-01-04	0	8130	8150	7920	8140	281440	2265829150	227362165800	27931470
2	000030	우리은행	KOSPI	2016-01-04	-0.0249	8820	8820	8580	8600	1714060	14844939720	5813600000000	676000000
3	000040	KR모터스	KOSPI	2016-01-04	0.0276	1295	1325	1255	1305	6053163	7839874495	228776632020	175307764
4	000050	경방	KOSPI	2016-01-04	-0.0649	187500	191000	180000	180000	3307	602364500	493474860000	2741527
5	000060	메리츠화재	KOSPI	2016-01-04	-0.0155	15900	15950	15750	15900	114871	1822816950	1684811700000	105963000
6	000070	삼양홀딩스	KOSPI	2016-01-04	-0.0472	156500	158000	151000	151500	18848	2902358000	1297487056500	8564271
7	000075	삼양홀딩스우	KOSPI	2016-01-04	-0.0029	68700	69200	66200	68300	911	61125400	20767161400	304058
8	000080	하이트진로	KOSPI	2016-01-04	-0.0256	23300	23400	22800	22800	168023	3859920550	1599046330800	70133611
9	000087	하이트진로2우B	KOSPI	2016-01-04	0.0027	18300	18350	18050	18350	2415	44103550	20738032300	1130138
10	000100	유한양행	KOSPI	2016-01-04	-0.0018	276000	282500	271500	272000	59355	16332957500	3033492512000	11152546
11	000105	유한양행우	KOSPI	2016-01-04	0	181000	181000	175500	180000	17	3032500	42513840000	236188
12	000120	CJ대한통운	KOSPI	2016-01-04	-0.0314	189000	190500	185000	185000	31606	5883434000	4220283640000	22812344
13	000140	하이트진로홀딩스	KOSPI	2016-01-04	-0.0515	13450	13600	12900	12900	16138	211134000	299367268500	23206765
14	000145	하이트진로홀딩스	KOSPI	2016-01-04	0.0046	10950	11200	10800	10850	986	10808050	5108288500	470810
15	000150	두산	KOSPI	2016-01-04	-0.0667	87600	87900	82600	82600	269584	22779210300	1756975348800	21270888

파라미터:
원시주가를 저장할 테이블명 : table_name
조회 시작: start_date = ''
조회 종료: end_date = ''
저장 여부: save = False (True: 데이터 DB 저장)

원시주가 데이터는 KRX사이트를 통하여 불러온다.

1. 주가 데이터 수집 모듈

2) 원시주가 데이터 수집 – 시작일 지정

```
def getStartDate(table_name):
    en = EngineDB()
    engine = en.getEngine()
    # 테이블 존재 확인
    sql = "SELECT count(*) FROM Information_schema.tables WHERE table_schema = 'krx_stock_data' " \
          "AND table_name = '{}'".format(table_name)

    exist = pd.read_sql_query(sql, engine)

    if exist.values == 0:
        return '2016-01-01'

    # 저장된 마지막 날짜 확인
    sql = 'select 날짜 from {} ORDER BY 날짜 DESC LIMIT 1'.format(table_name)
    df = pd.read_sql_query(sql, engine)

    start_date = df + timedelta(days=1)
    start_date = (start_date['날짜'].iloc[0]).strftime('%Y-%m-%d')

    return start_date  # 주가를 조회할 첫 날짜
```

원시주가가 저장된 테이블이 이미 존재하는지 확인한다.

테이블이 존재하며
시작일이 지정되지 않은 경우
DB에서 마지막으로 저장된
날짜를 불러온다.

```
now = datetime.now()
end_date = now.strftime('%Y-%m-%d')
```

종료일이 입력되지 않은 경우 현재 일자를 종료일로 한다.

1. 주가 데이터 수집 모듈

2) 원시주가 데이터 수집 – 시작,종료일 지정

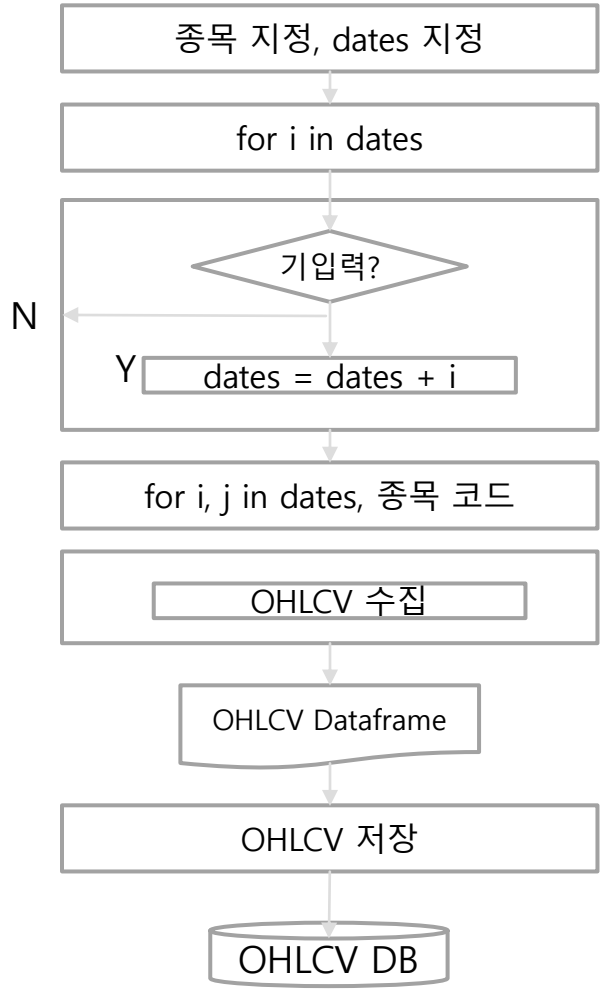
```
def getStartDate(table_name):
    en = EngineDB()
    engine = en.getEngine()
    # 테이블 존재 확인
    sql = "SELECT count(*) FROM Information_schema.tables WHERE table_schema = 'krx_stock_data' " \
          "AND table_name = '{}'".format(table_name)

    exist = pd.read_sql_query(sql, engine)

    if exist.values == 0:
        return '2016-01-01'

    # 저장된 마지막 날짜 확인
    sql = 'select 날짜 from {} ORDER BY 날짜 DESC LIMIT 1'.format(table_name)
    df = pd.read_sql_query(sql, engine)

    start_date = df + timedelta(days=1)
    start_date = (start_date['날짜'].iloc[0]).strftime('%Y-%m-%d')

    return start_date  # 주가를 조회할 첫 날짜
```

원시주가가 저장된 테이블이 이미 존재하는지 확인한다.

테이블이 존재하며
시작일이 지정되지 않은 경우
DB에서 마지막으로 저장된
날짜를 불러온다.

1. 주가 데이터 수집 모듈

3) 수정 주가 데이터 수집



- FinanceDataReader로 부터 수정 주가 데이터를 가져온다.

1. 주가 데이터 수집 모듈

3) 수정 주가 데이터 수집

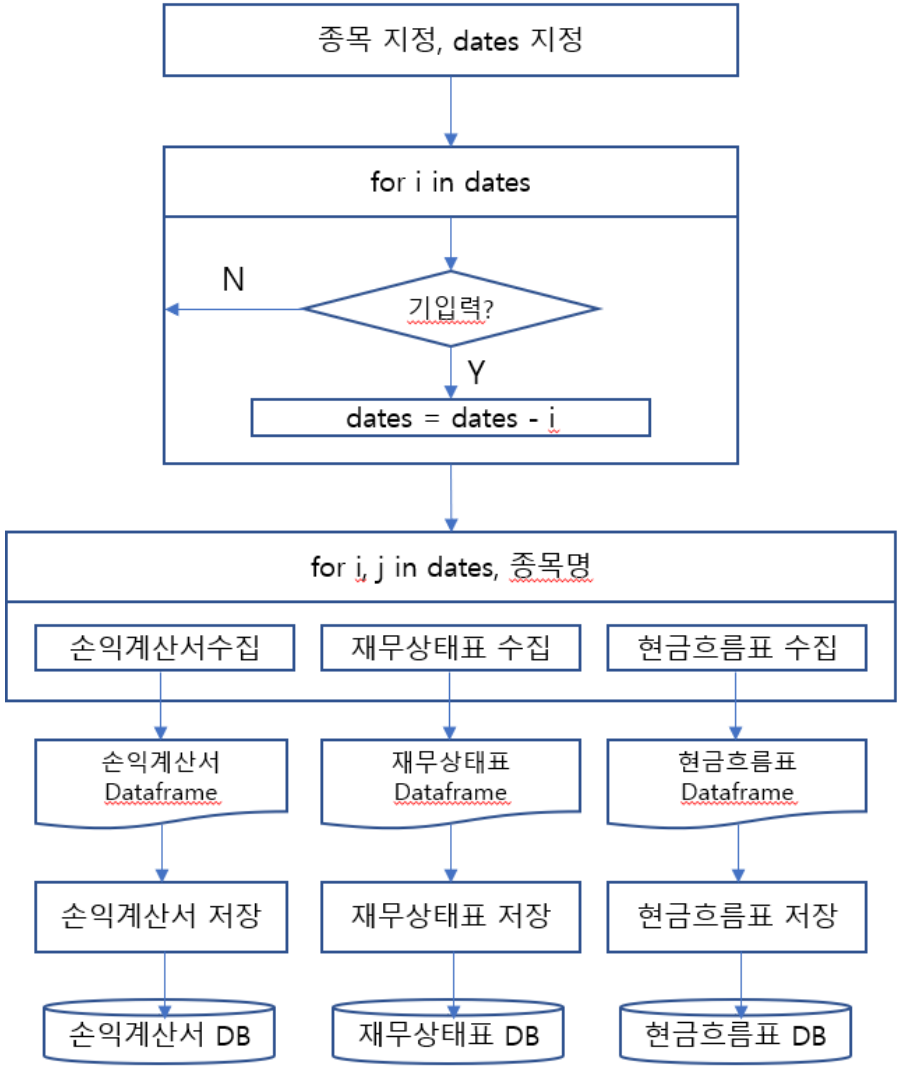
	종목코드	종목명	날짜	시가	고가	저가	종가	거래량	등락률
1	000020	동화약품	2016-01-04	8130	8150	7920	8140	281440	0
2	000040	KR모터스	2016-01-04	7906	8089	7663	7972	6053163	0.0275844
3	000050	경방	2016-01-04	18750	19100	18000	18000	3307	-0.0649351
4	000060	메리츠화재	2016-01-04	15900	15950	15750	15900	114871	-0.0154799
5	000070	삼양홀딩스	2016-01-04	156500	158000	151000	151500	18848	-0.0471698
6	000080	하이트진로	2016-01-04	23300	23400	22800	22800	168023	-0.025641
7	000100	유한양행	2016-01-04	42503	43504	41811	41889	385408	-0.00183482
8	000120	CJ대한통운	2016-01-04	189000	190500	185000	185000	31606	-0.0314136
9	000140	하이트진로홀딩스	2016-01-04	13450	13600	12900	12900	16138	-0.0514706
10	000150	두산	2016-01-04	69005	69241	65066	65067	269584	-0.0666714
11	000180	성창기업지주	2016-01-04	3300	3400	3220	3245	60185	0.0172414
12	000210	DL	2016-01-04	59728	60271	58192	58193	133374	-0.0402421
13	000220	유유제약	2016-01-04	7110	7162	6773	6929	64326	-0.0326679
14	000230	일동홀딩스	2016-01-04	27469	27577	26981	27144	64666	-0.00796725
15	000240	한국앤컴퍼니	2016-01-04	17400	17500	17150	17350	85886	-0.00857143

파라미터:
원시주가를 저장할 테이블명 : table_name
조회 시작: start_date = ''
조회 종료: end_date = ''
저장 여부: save = False (True: DB 저장)

원시주가 데이터는 KRX사이트를 통하여 불러온다.

2. 재무데이터 수집 및 저장

2. 재무데이터 수집 및 저장



1. Fnguid의 손익계산서, 재무상태표, 현금흐름표 수집 연결/별도 – 연간/분기 데이터 수집

2. 기입력 검사: 불러온 데이터의 종목코드, 기간을 이용하여 테이블을 검사한다. 입력되지 않았을 경우 데이터를 추가한다.

2. 재무데이터 수집 및 저장

1) 재무데이터 수집

```
def getISData(stock, rpt_type, freq):
def getBSData(stock, rpt_type, freq):
def getCFData(stock, rpt_type, freq):
```

재무데이터를 직접적으로 수집하는 함수이다.

```
def getIS(rpt_type, freq, table='is_kr', save=False):
def getBS(rpt_type, freq, table='bs_kr', save=False):
def getCF(rpt_type, freq, table='cf_kr', save=False):
```

수집할 재무데이터를 전달받고 위의 함수를 이용하여 전달 받은 재무데이터를 합하여 DB에 저장하는 역할을 맡는다.

2. 재무데이터 수집 및 저장

2) 손익계산서 수집

	종목코드	종목명	시장	기간	컬럼	매출액	매출원가	매출총이익	판매비	관리비	영업이익	금융수익
1	000020	동화약품	KOSPI	2019-12-01	consolidated_a	3071.5	1855.54	1215.96	96.85	141.26	95.51	25.09
2	000040	KR모터스	KOSPI	2019-12-01	consolidated_a	1325.79	1335.67	-9.88	70.69	29.98	-246.5	14.64
3	000050	경방	KOSPI	2019-12-01	consolidated_a	3438.92	2443.96	994.96	14.06	572.09	320.21	32.42
4	000060	메리츠화재	KOSPI	2019-12-01	consolidated_a	174.13	165.69	8.44	1.89	11.15	3527.56	2.54
5	000070	삼양홀딩스	KOSPI	2019-12-01	consolidated_a	24885.6	20251.3	4634.31	879.08	832.13	793.76	242.5
6	000080	하이트진로	KOSPI	2019-12-01	consolidated_a	20350.6	11632.2	8718.43	1239.2	969.05	882.47	31.51
7	000100	유한양행	KOSPI	2019-12-01	consolidated_a	14803.5	10495	4308.51	309.52	677.32	125.36	172.38
8	000120	CJ대한통운	KOSPI	2019-12-01	consolidated_a	104151	94425.2	9725.91	108.84	2605.41	3071.86	560.28
9	000140	하이트진로홀딩	KOSPI	2019-12-01	consolidated_a	20263.2	11486.7	8776.47	1246.63	929.85	1009.44	40.73
10	000150	두산	KOSPI	2019-12-01	consolidated_a	180416	149670	30746.2	1728.33	2858.4	12285.5	5382.71
11	000180	성창기업지주	KOSPI	2019-12-01	consolidated_a	1510.42	1312.08	198.33	61.48	58.91	-29.51	4.43
12	000210	DL	KOSPI	2019-12-01	consolidated_a	16604.3	14006.2	2598.15	266.32	641.93	1359.86	213.37

Fnguid로 부터 손익계산서를 불러온다.

2. 재무데이터 수집 및 저장

3) 재무데이터 수집

	종목코드	종목명	시장	기간	컬럼	자산	유동자산	비유동자산	기타금융업자산	부채	유동부채
1	모두 선택	3S	KOSDAQ	2019-03-01	consolidated_a	533.29	180.79	352.5	0	189.58	172.21
2	060310	3S	KOSDAQ	2020-03-01	consolidated_a	496.58	193.13	303.45	0	162.93	135.2
3	060310	3S	KOSDAQ	2021-03-01	consolidated_a	546.15	180.71	365.44	0	180.68	157.67
4	060310	3S	KOSDAQ	2021-12-01	consolidated_a	656.94	284.75	372.19	0	260.9	238.45
5	095570	AJ네트웍스	KOSPI	2019-12-01	consolidated_a	18032.6	5010.55	13022	0	14559.1	7968.37
6	095570	AJ네트웍스	KOSPI	2020-12-01	consolidated_a	15881.7	3546.87	12334.8	0	13001.6	7088.97
7	095570	AJ네트웍스	KOSPI	2021-12-01	consolidated_a	13550.4	2739.13	10811.3	0	9925.34	5251.77
8	095570	AJ네트웍스	KOSPI	2022-03-01	consolidated_a	13672.3	2618.73	11053.6	0	10021.9	6138.04
9	006840	AK홀딩스	KOSPI	2019-12-01	consolidated_a	43279.6	13277.9	30001.8	0	28980.3	15654.8
10	006840	AK홀딩스	KOSPI	2020-12-01	consolidated_a	41520.9	11852.5	29668.4	0	28894.3	16116.7
11	006840	AK홀딩스	KOSPI	2021-12-01	consolidated_a	45487.4	13527.5	31959.9	0	33463.8	18739.4
12	006840	AK홀딩스	KOSPI	2022-03-01	consolidated_a	45365	13653.8	31711.2	0	33975.3	19822.4
13	054620	APS홀딩스	KOSDAQ	2019-12-01	consolidated_a	2977.08	564.61	2412.47	0	829.23	435.57
14	054620	APS홀딩스	KOSDAQ	2020-12-01	consolidated_a	2713.48	496.57	2216.91	0	1077.55	302.23
15	054620	APS홀딩스	KOSDAQ	2021-12-01	consolidated_a	3370.93	1004.84	2366.09	0	1498.23	689.73
16	054620	APS홀딩스	KOSDAQ	2022-03-01	consolidated_a	3363.22	971.42	2391.8	0	1533.15	735.4
17	265520	AP시스템	KOSDAQ	2019-12-01	consolidated_a	4209.66	2524.29	1685.37	0	3068.43	2868.08
18	265520	AP시스템	KOSDAQ	2020-12-01	consolidated_a	3713.15	2222.55	1490.6	0	2331.55	1532.56
19	265520	AP시스템	KOSDAQ	2021-12-01	consolidated_a	4287.2	2912.54	1374.66	0	2371.78	1355.94
20	265520	AP시스템	KOSDAQ	2022-03-01	consolidated_a	4668.89	3228.15	1440.74	0	2659.4	1649.34
21	211270	AP위성	KOSDAQ	2019-12-01	consolidated_a	1026.43	869.37	157.06	0	203.83	195.26
22	211270	AP위성	KOSDAQ	2020-12-01	consolidated_a	1127.47	1013.6	113.88	0	328.47	316.44
23	211270	AP위성	KOSDAQ	2021-12-01	consolidated_a	1111.88	1007.74	104.14	0	300.83	296.28
24	211270	AP위성	KOSDAQ	2022-03-01	consolidated_a	0	0	0	0	0	0

Fnguid로 부터 재무데이터를 불러온다.

2. 재무데이터 수집 및 저장

4) 현금흐름표 수집

	종목코드	종목명	시장	기간	컬럼	영업활동으로인한현금흐름	당기순손익	법인세비용...	현금유출이없는비용등가산	감가상각비	현금유입이없는수익등차감	`영업활동으로인
1	000020	동화약품	KOSPI	2021-06-01	consolidated_q	62.97	70.4	0	80.3	25.57	24.28	
2	000040	KR모터스	KOSPI	2021-06-01	consolidated_q	42.43	-13.93	0	36.8	8.14	-0.38	
3	000050	경방	KOSPI	2021-06-01	consolidated_q	-159.43	103.81	0	223.75	101.92	61.21	
4	000060	메리츠화재	KOSPI	2021-06-01	consolidated_q	10070.3	0	0	7854.55	0	0	
5	000070	삼양홀딩스	KOSPI	2021-06-01	consolidated_q	982.76	858.49	0	0	0	-106.24	
6	000080	하이트진로	KOSPI	2021-06-01	consolidated_q	-2968.41	236.89	0	704.7	350.48	37.86	
7	000100	유한양행	KOSPI	2021-06-01	consolidated_q	-723.12	332.21	0	366.24	117.16	188.07	
8	000120	CJ대한통운	KOSPI	2021-06-01	consolidated_q	92.82	493.63	0	1683.93	0	0	
9	000140	하이트진로홀딩	KOSPI	2021-06-01	consolidated_q	-2990.89	233.84	0	728.25	352.87	38.26	
10	000150	두산	KOSPI	2021-06-01	consolidated_q	-202.24	2001.04	0	4568.47	1118.92	-1026.34	
11	000180	성창기업지주	KOSPI	2021-06-01	consolidated_q	39.3	51.8	0	0	0	2.3	
12	000210	DL	KOSPI	2021-06-01	consolidated_q	742.57	4501.61	0	910.63	305.01	4537.67	
13	000220	유유제약	KOSPI	2021-06-01	consolidated_q	47.96	3.34	0	20.21	10.22	1.1	
14	000230	일동홀딩스	KOSPI	2021-06-01	consolidated_q	45.08	50.93	0	210.66	54.47	237.26	
15	000240	한국앤컴퍼니	KOSPI	2021-06-01	consolidated_q	576.87	518.18	0	0	0	249.42	
16	000250	삼천당제약	KOSDAQ	2021-06-01	consolidated_q	-8.65	-13.93	0	69.54	15.91	41.41	
17	000270	기아	KOSPI	2021-06-01	consolidated_q	27131	13429.1	0	15856.8	4257.05	3273.16	
18	000300	대유플러스	KOSPI	2021-06-01	consolidated_q	70.3	-7.25	0	157.55	33.75	76.15	
19	000320	노루홀딩스	KOSPI	2021-06-01	consolidated_q	188.1	42.12	0	144.88	53.78	22.53	

Fnguid로 부터 재무데이터를 불러온다.

2. 재무데이터 수집 및 저장

5) 재무데이터 업데이트

```
def dataTimeCheck(code, date, column, table):
    en = EngineDB()
    engine = en.getEngine()
    sql = "SELECT count(*) FROM Information_schema.tables WHERE table_schema = 'krx_stock_data' " \
          "AND table_name = '{}'".format(table)
    exist = pd.read_sql_query(sql, engine)
    if exist.values == 0:
        return True

    sql = "select 기간 FROM {} where 종목코드 = '{}' and 컬럼 = '{}' order by 기간 desc limit 1".format(table, code, column)
    fdate = pd.read_sql_query(sql, engine)
    engine.dispose()
    if fdate.empty:
        return True

    fdate = fdate.iloc[0].values
    if fdate < date:
        return True
    else:
        return False
```

재무데이터를 업데이트 시에는 dateTimeCheck로 저장할 일자가 저장된 일자보다 이후의 데이터인지 확인한다.

2. 재무데이터 수집 및 저장

6) 데이타베이스 저장

```
class ConnectDB:

    def __init__(self):
        self.conn = pymysql.connect(host='127.0.0.1', port=3306, db='krx_stock_data',
                                    user='root', passwd='abcd123456', autocommit=True)
        self.cursor = self.conn.cursor()

    def executeQuery(self, query):
        self.cursor.execute(query)
        df = pd.DataFrame(self.cursor.fetchall())
        df.columns = [col[0] for col in self.cursor.description]
        return df

    def closeConnection(self):
        self.cursor.close()
        self.conn.close()


class EngineDB:

    def __init__(self):
        self.server = '127.0.0.1'
        self.user = 'root'
        self.passwd = 'abcd123456'
        self.db = 'krx_stock_data'

    def getEngine(self):
        # sqlalchemy의 create_engine을 이용하여 DB 연결
        engine = create_engine('mysql+pymysql://{}:{}@{}/{}?charset=utf8'.format(
            self.user, self.passwd, self.server, self.db))
        return engine
```

저장하고자하는 데이터는 ConnectDB 클래스의 pymysql과 engine을 이용하여 sql문을 전달받으면 실행이 가능하도록 하였다.

3. 데이터 처리

3. 데이터 처리

1) 데이터 불러오기

수정주가 불러오기

```
sql = "SELECT srp.*,  sop.시장구분, sop.거래대금, sop.시가총액, sop.상장주식수 FROM stock_revise_price srp " \
      "LEFT JOIN stock_origin_price sop " \
      "ON (srp.종목코드, srp.날짜) = (sop.종목코드, sop.날짜) " \
      "WHERE srp.날짜 BETWEEN '{}' AND '{}'".format(start_date, end_date)
```

재무데이터 불러오기

```
sql = "SELECT * FROM {} WHERE 종목코드='{}' AND 기간 LIKE '{}' AND 컬럼='{}'".format(
    table, stock_code, period, column)
```

앞으로 팩터를 만들때 필요한 것이 수정주가 데이터와 트레일링 데이터이다.
연결-분기를 이용할 경우 4개의 분기 데이터를 묶어 사용하기 위한 트레일링 데이터와 분기별 수정주가가 필요하다.

3. 데이터 처리

2) 저장할 종목 조회

```
def get_stockcode():
    cd = ConnectDB()
    sql = "SELECT Symbol as '종목코드' FROM stock_list"
    df = cd.executeQuery(sql)
    cd.closeConnection()

    df = df.sort_values(by=['종목코드'])
    df = df.values.tolist()
    return df
```

저장하고자 하는 종목의 종목코드를 DB에서 불러온다.

3. 데이터 처리

3) 데이터 저장 확인

```
# 데이터가 존재하는지 확인
def CheckDataExists(table, stock_code, period, column):
    cd = ConnectDB()
    sql = "SELECT EXISTS (SELECT 종목코드, 컬럼, 기간 " \
          "FROM {} WHERE 종목코드='{}' AND 컬럼='{}' AND 기간 LIKE '{}')" \
          " as 'IsExists'".format(table[0], stock_code, column, period)
    exist = cd.executeQuery(sql)
    if exist.values == 0:  # is_kr 데이터가 존재하지 않음
        return False

    sql = "SELECT EXISTS (SELECT 종목코드, 컬럼, 기간 " \
          "FROM {} WHERE 종목코드='{}' AND 컬럼='{}' AND 기간 LIKE '{}')" \
          " as 'IsExists'".format(table[1], stock_code, column, period)
    exist = cd.executeQuery(sql)
    if exist.values == 0:  # bs_kr 데이터가 존재하지 않음
        return False

    sql = "SELECT EXISTS (SELECT 종목코드, 컬럼, 기간 " \
          "FROM {} WHERE 종목코드='{}' AND 컬럼='{}' AND 기간 LIKE '{}')" \
          " as 'IsExists'".format(table[2], stock_code, column, period)
    exist = cd.executeQuery(sql)
    if exist.values == 0:  # cf_kr 데이터가 존재하지 않음
        return False

    return True  # 모든 데이터 존재
```

이용하고자 하는 데이터가 DB에 저장되어있는지 확인할 필요가 있다.

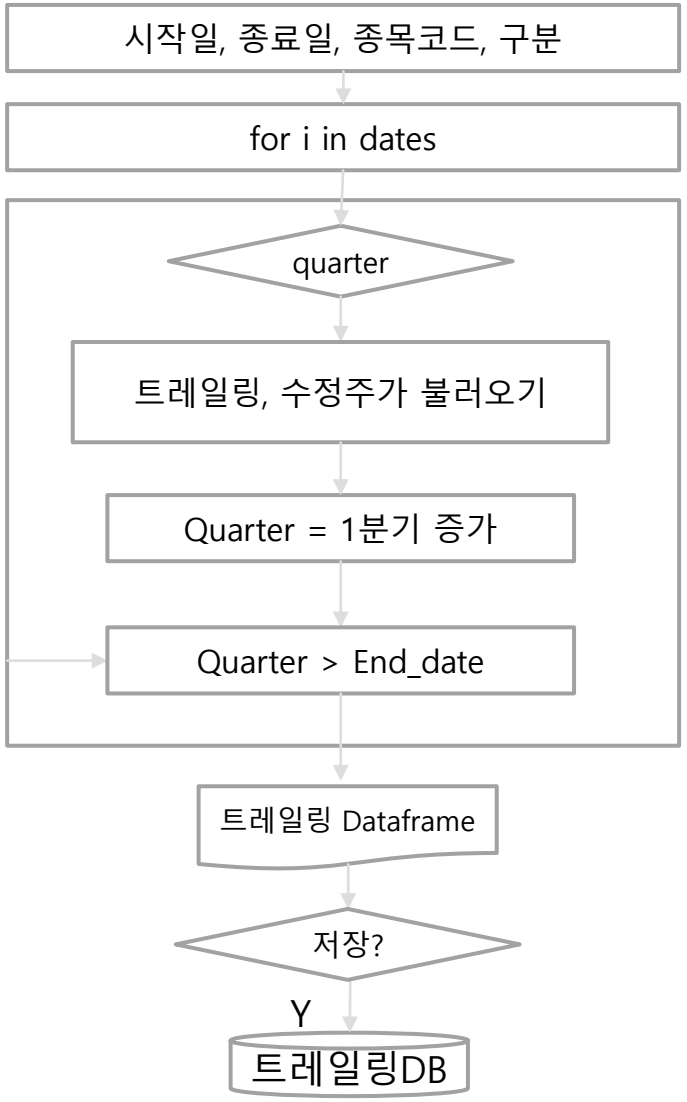
4. 종목 추출 모듈

4. 종목 추출 모듈

- 이익대비 저평가 종목: PER
- 장부가치대비 저평가 종목: PBR
- 매출대비 저평가 종목: PSR
- 현금흐름대비 저평가 종목: PCR
- 가치지표 결합하기
- 가치투자 4대장 콤보: (PER+PBR+PSR+PCR)
- 실적대비 기업가치: EV/EBITDA & EV Sales
- 그레이엄의 청산가치: NCAV
- PEG 기반 종목 추출

4. 종목 추출 모듈 - 가치주

1) getFator 클래스



- getFactor 클래스

팩터를 계산하는 클래스이다.

클래스가 생성될때 트레일링 데이터와 수정주가 데이터가 만들어진다.

트레일링 데이터는 생성후 DB 저장이 가능하다.

저장된 트레일링 데이터는 팩터 계산에 사용될 수 있다.

팩터클래스에서는 각종 팩터를 계산하기 위한 팩터가 있다.

4. 종목 추출 모듈 - 가치주

1) getFator 클래스

```
def __init__(self, start_quarter, end_quarter=None, stock_code=None, by='consolidated', freq='q'):
    self.stock_code = stock_code
    self.by = by
    self.freq = freq

    self.start_quarter = start_quarter
    if end_quarter is None:
        end_quarter = start_quarter
    self.end_quarter = end_quarter
    self.by = by
    self.freq = freq

    # 수정주가 데이터 불러오기
    print('수정주가 데이터 로딩')
    self.priceData = GetDBData.get_price(start_quarter, stock_code=stock_code)
    if start_quarter != end_quarter:
        quarter = self.getNextQuarter(start_quarter)
        while quarter <= end_quarter:
            temp = GetDBData.get_price(quarter, stock_code=stock_code)
            self.priceData = pd.concat([self.priceData, temp])
            quarter = self.getNextQuarter(quarter)
    print('수정주가 데이터 로딩 성공')
    self.priceData.sort_values(['종목코드', '기간'])
```

- getFactor 클래스

팩터를 계산하는 클래스이다. 클래스가 선언되면 필요한 주가 데이터와 트레일링 데이터를 수집한다.

```
if stock_code.upper() != 'LATER':
    if stock_code is not None:
        # 손익, 재무, 현금흐름 데이터 불러오기
        self.trailingData = GetDBData.get_trailing(stock_code, start_quarter, by, freq)

        quarter = self.getNextQuarter(start_quarter)
        while quarter <= end_quarter:
            temp = GetDBData.get_trailing(stock_code, quarter, by, freq)
            self.trailingData = pd.concat([self.trailingData, temp])
            quarter = self.getNextQuarter(quarter)

    elif stock_code.upper() == 'ALL':  # 모든 종목의 트레일링 데이터 조회
        # 주식의 종목코드 목록
        print('모든 종목의 트레일링 데이터 로딩')
        self.stockList = GetDBData.get_stockcode()
        self.trailingData = pd.DataFrame()
        # 손익, 재무, 현금흐름
        for s in self.stockList:
            temp = GetDBData.get_trailing(s[0], start_quarter, by, freq)
            self.trailingData = pd.concat([self.trailingData, temp])

        quarter = self.getNextQuarter(start_quarter)
        while quarter <= end_quarter:
            for s in self.stockList:
                temp = GetDBData.get_trailing(s[0], quarter, by, freq)
                self.trailingData = pd.concat([self.trailingData, temp])
            quarter = self.getNextQuarter(quarter)
        print('트레일링 데이터 로딩 성공')

    self.trailingData.sort_values(['종목코드', '기간'])
    self.Factor = pd.DataFrame(self.trailingData, columns=['종목코드', '종목명', '시장', '기간', '컬럼'])
```

4. 종목 추출 모듈 - 가치주

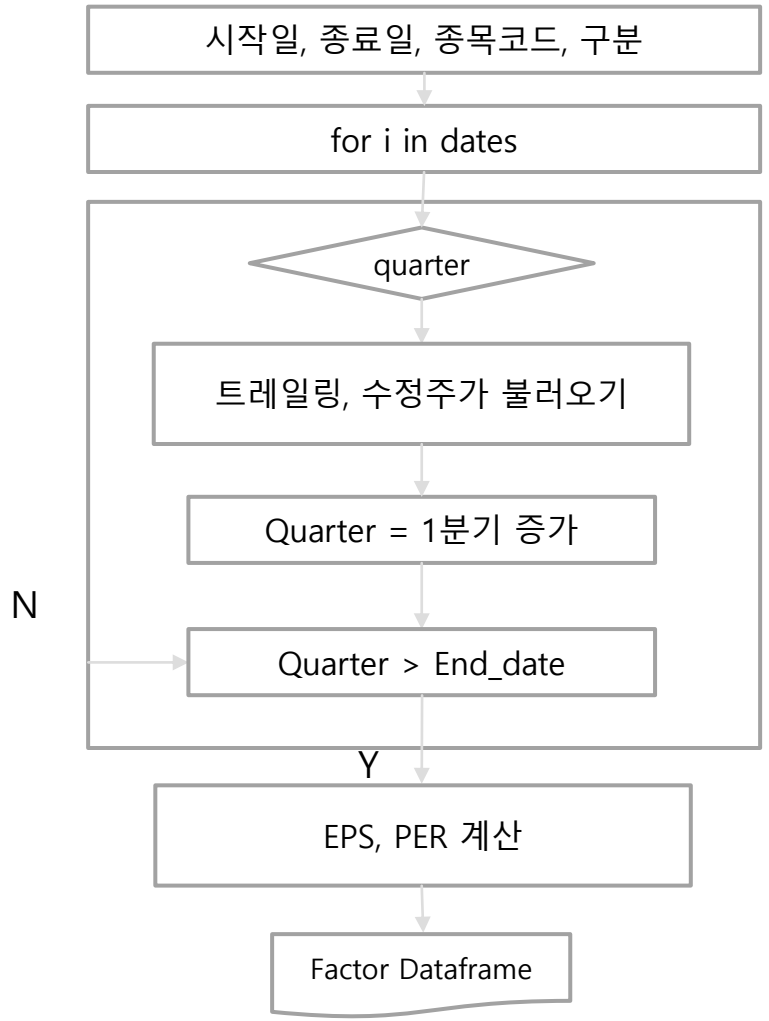
2) getFilteredRow 함수

```
def getFilteredRow(self, idx):
    valid = True
    tf = self.trailingData[(self.trailingData['종목코드'] == self.Factor['종목코드'].iloc[idx]) & (
            self.trailingData['기간'] == self.Factor['기간'].iloc[idx])]
    pf = self.priceData[(self.priceData['종목코드'] == self.Factor['종목코드'].iloc[idx]) & (
            self.priceData['기간'] == self.Factor['기간'].iloc[idx])]
    # 데이터 유효성 확인
    if tf.empty or pf.empty:
        valid = False
    return tf, pf, valid
```

- 팩터를 계산하기 위해서는 필요한 데이터만을 불러와야 한다.
- 트레일링 데이터 및 수정 주가 데이터에서 원하는 종목과 기간의 데이터만을 반환해주는 함수이다.
- 만약, 데이터가 존재하지 않다면 valid가 False를 가리키며 다음 종목을 검색한다.

4. 종목 추출 모듈 - 가치주

3) PER 계산 함수



- 이익대비 저평가 종목: PER => getPER
  - EPS = 당기순이익 / 상장주식수
  - PER = 주가 / EPS

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 EPS PER 반환

4. 종목 추출 모듈 - PER

3) PER 계산 함수

```
# 주가수익비율 PER
def getPER(self):
    self.Factor['PER'] = np.NaN
    self.Factor['EPS'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['EPS'].iloc[i] = (tf['당기순이익'].iloc[0] * getFactor.won) / pf['상장주식수'].iloc[0]
        self.Factor['PER'].iloc[i] = pf['종가'].iloc[0] / self.Factor['EPS'].iloc[i]

    self.Factor['PER'] = self.Factor['PER'].replace([np.inf, -np.inf], 0)
    self.Factor = self.Factor.dropna()
    self.Factor = self.Factor.reset_index(drop=True)
    return self.Factor.set_index('종목코드')
```

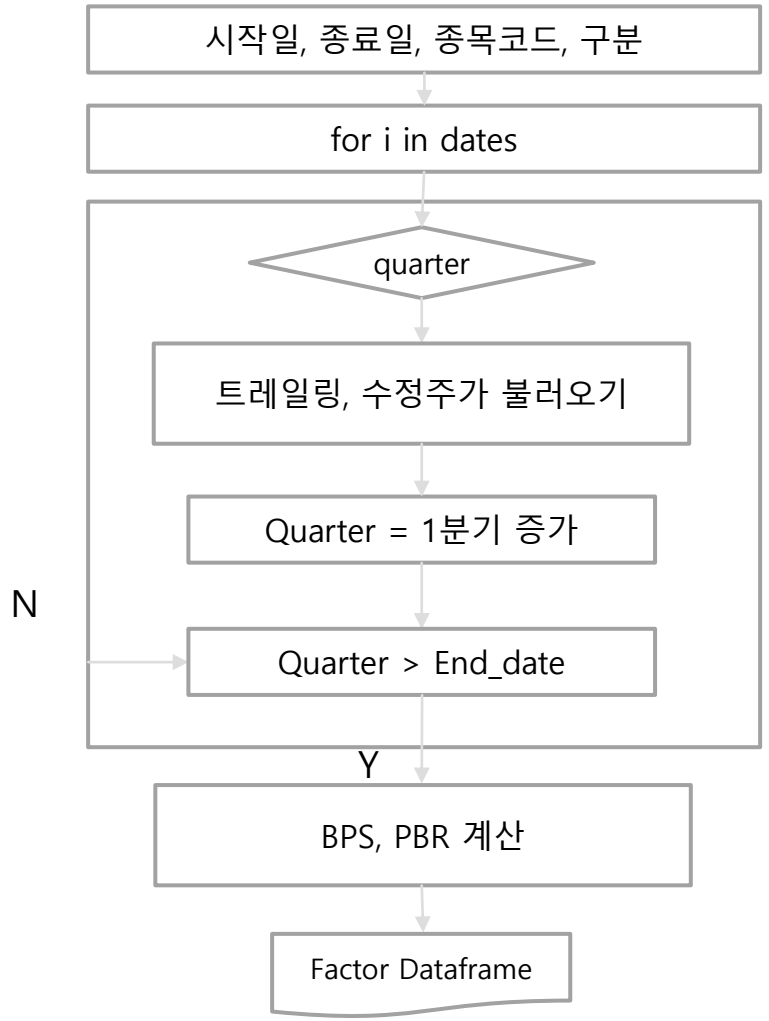
4. 종목 추출 모듈 - PER

3) PER

	종목코드	종목명	시장	기간	컬럼	EPS	PER
0	000020	동화약품	KOSPI	2022Q1	consolid...	711.491...	19.18505
1	000040	KR모터스	KOSPI	2022Q1	consolid...	-134.02...	-6.63299
2	000050	경방	KOSPI	2022Q1	consolid...	1115.07...	14.30401
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	6168.19...	7.69269
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	27841.7...	3.09248
5	000080	하이트진로	KOSPI	2022Q1	consolid...	1136.57...	32.94992
6	000100	유한양행	KOSPI	2022Q1	consolid...	1298.36...	44.97974
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	7503.52...	17.25855
8	000140	하이트진...	KOSPI	2022Q1	consolid...	3266.50...	4.17878
9	000150	두산	KOSPI	2022Q1	consolid...	16295.2...	6.38223
10	000180	성창기업...	KOSPI	2022Q1	consolid...	-14.76669	-177.42...
11	000210	DL	KOSPI	2022Q1	consolid...	31432.3...	1.94704
12	000220	유유제약	KOSPI	2022Q1	consolid...	-89.13467	-90.76154
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	-20915....	-1.04228
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	1876.53...	7.80696
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	-180.99...	-198.34...
16	000270	기아	KOSPI	2022Q1	consolid...	11737.2...	6.30470
17	000300	대유플러스	KOSPI	2022Q1	consolid...	24.13248	54.07651
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	571.432...	21.69984
19	000370	한화손해...	KOSPI	2022Q1	consolid...	1003.97...	5.01008
20							
21							
22							

4. 종목 추출 모듈 – PBR

4) PBR 계산 함수



- getPBR
  - BPS = 자본 / 상장주식수
  - PBR = 주가 / BPS

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

	종목명	시장	기간	컬럼	PBR	BPS
005930	삼성전자	KOSPI	2022Q1	consolid...	0.34645	200891....

4. 종목 추출 모듈 - PBR

4) PBR 계산 함수

```python
# 주가순자산비율 PBR
def getPBR(self):
    self.Factor['PBR'] = np.NaN
    self.Factor['BPS'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['BPS'].iloc[i] = (tf['자본'].iloc[0] * getFactor.won) / pf['상장주식수'].iloc[0]
        self.Factor['PBR'].iloc[i] = pf['종가'].iloc[0] / self.Factor['BPS'].iloc[i]
    self.setMissingValue(by='PBR')
    return self.Factor.set_index('종목코드')
```

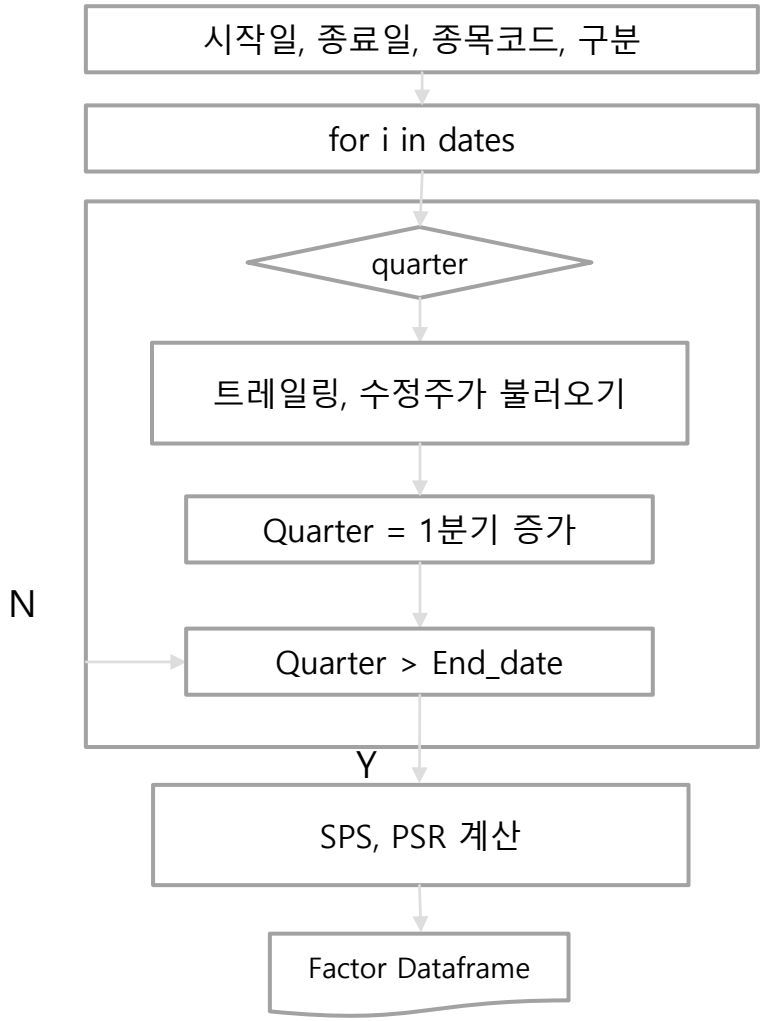
4. 종목 추출 모듈

4) PBR

	종목코드	종목명	시장	기간	컬럼	BPS	PBR
0	000020	동화약품	KOSPI	2022Q1	consolid...	51003.0...	0.26763
1	000040	KR모터스	KOSPI	2022Q1	consolid...	2094.25...	0.42449
2	000050	경방	KOSPI	2022Q1	consolid...	110169....	0.14478
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	73541.8...	0.64521
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	1129448...	0.07623
5	000080	하이트진로	KOSPI	2022Q1	consolid...	61759.8...	0.60638
6	000100	유한양행	KOSPI	2022Q1	consolid...	104422....	0.55927
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	696767....	0.18586
8	000140	하이트진...	KOSPI	2022Q1	consolid...	186874....	0.07304
9	000150	두산	KOSPI	2022Q1	consolid...	2124397...	0.04896
10	000180	성창기업...	KOSPI	2022Q1	consolid...	31142.0...	0.08413
11	000210	DL	KOSPI	2022Q1	consolid...	848530....	0.07212
12	000220	유유제약	KOSPI	2022Q1	consolid...	29989.8...	0.26976
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	84445.5...	0.25815
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	156321....	0.09372
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	44121.0...	0.81367
16	000270	기아	KOSPI	2022Q1	consolid...	335146....	0.22080
17	000300	대유플러스	KOSPI	2022Q1	consolid...	4681.27...	0.27877
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	175511....	0.07065
19	000370	한화손해...	KOSPI	2022Q1	consolid...	46154.1...	0.10898

4. 종목 추출 모듈 - PSR

5) PSR 계산 함수



- getPSR
  - SPS = 매출액 / 상장주식수
  - PSR = 주가 / SPS

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

	종목명	시장	기간	컬럼	SPS	PSR
005930	삼성전자	KOSPI	2022Q1	consolid...	48912.6...	1.42295

4. 종목 추출 모듈 - PSR

5) PSR 계산 함수

```python
# 주가매출비율 PSR
def getPSR(self):
    self.Factor['SPS'] = np.NaN
    self.Factor['PSR'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['SPS'].iloc[i] = (tf['매출액'].iloc[0] * getFactor.won) / pf['상장주식수'].iloc[0]
        self.Factor['PSR'].iloc[i] = pf['종가'].iloc[0] / self.Factor['SPS'].iloc[i]
    self.setMissingValue(by='PSR')
    return self.Factor.set_index('종목코드')
```

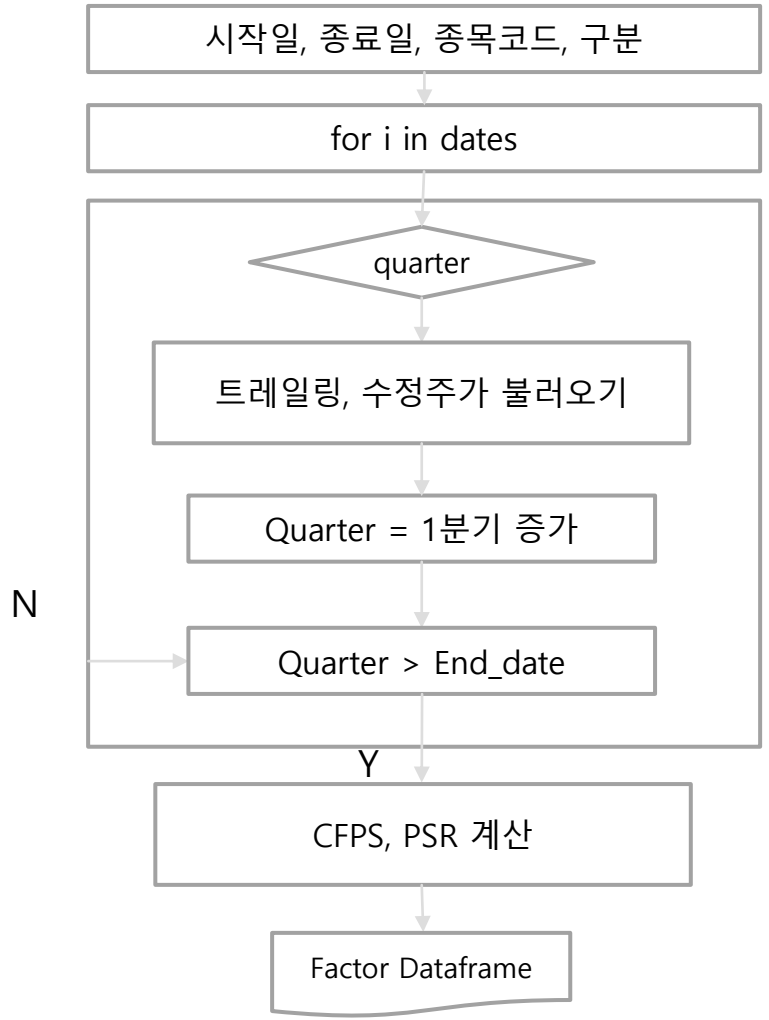
4. 종목 추출 모듈

5) PSR

	종목코드	종목명	시장	기간	컬럼	SPS	PSR
0	000020	동화약품	KOSPI	2022Q1	consolid...	10972.7...	1.24400
1	000040	KR모터스	KOSPI	2022Q1	consolid...	1455.85...	0.61064
2	000050	경방	KOSPI	2022Q1	consolid...	14501.8...	1.09986
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	0.00000	0.00000
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	377025....	0.22837
5	000080	하이트진로	KOSPI	2022Q1	consolid...	32103.1...	1.16655
6	000100	유한양행	KOSPI	2022Q1	consolid...	23482.1...	2.48700
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	504463....	0.25671
8	000140	하이트진...	KOSPI	2022Q1	consolid...	96539.0...	0.14139
9	000150	두산	KOSPI	2022Q1	consolid...	828736....	0.12549
10	000180	성창기업...	KOSPI	2022Q1	consolid...	2950.95...	0.88785
11	000210	DL	KOSPI	2022Q1	consolid...	131639....	0.46490
12	000220	유유제약	KOSPI	2022Q1	consolid...	7235.57...	1.11809
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	53594.8...	0.40676
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	10497.0...	1.39563
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	7445.96...	4.82140
16	000270	기아	KOSPI	2022Q1	consolid...	176725....	0.41873
17	000300	대유플러스	KOSPI	2022Q1	consolid...	4197.75...	0.31088
18							
19							
20							

4. 종목 추출 모듈 - PCR

6) PCR 계산 함수



- getPCR
  - CFPS = 영업활동현금흐름 / 상장주식수
  - PSR = 주가 / CFPS

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

	종목명	시장	기간	컬럼	CFPS	PCR
005930	삼성전자	KOSPI	2022Q1	consolid...	10344.6...	6.72808

4. 종목 추출 모듈 - PCR

6) PCR 계산 함수

```python
# 주가현금흐름비율 PCR
def getPCR(self):
    self.Factor['CFPS'] = np.NaN
    self.Factor['PCR'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['CFPS'].iloc[i] = (tf['영업활동으로인한현금흐름'].iloc[0] * getFactor.won) / pf['상장주식수'].iloc[0]
        self.Factor['PCR'].iloc[i] = pf['종가'].iloc[0] / self.Factor['CFPS'].iloc[i]

    self.Factor['PCR'] = self.Factor['PCR'].replace([np.inf, -np.inf], 0)
    self.Factor = self.Factor.dropna()
    self.Factor = self.Factor.reset_index(drop=True)
    return self.Factor.set_index('종목코드')
```

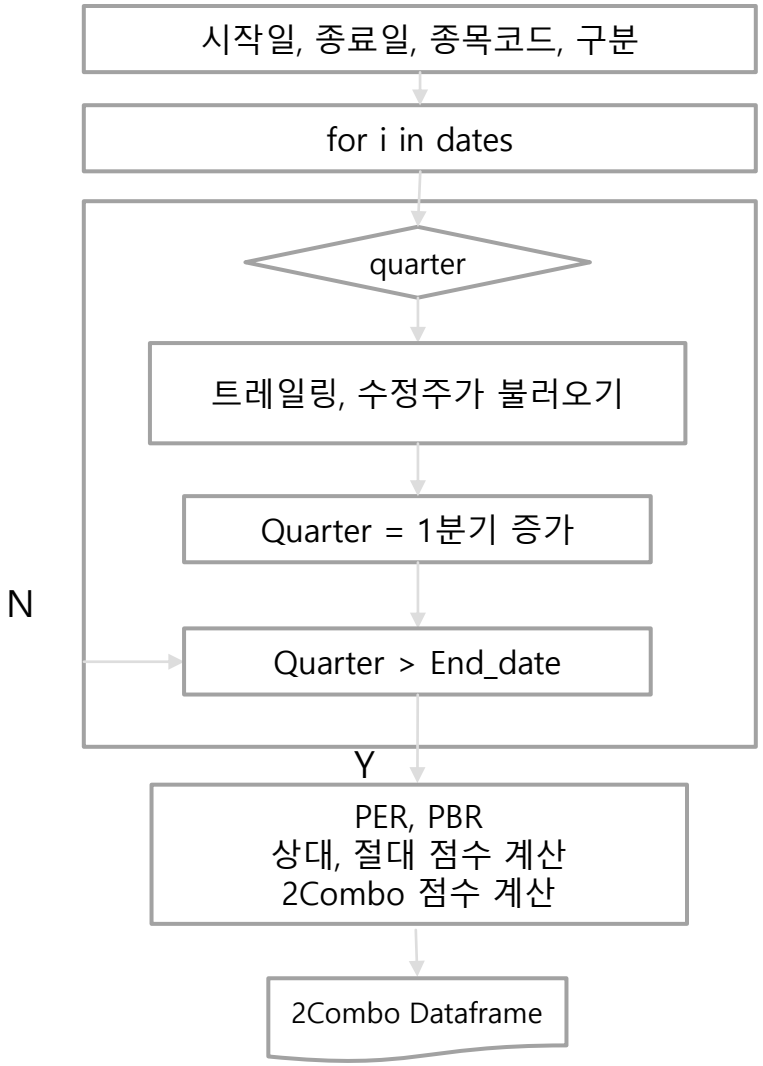
4. 종목 추출 모듈

6) PCR

	종목코드	종목명	시장	기간	컬럼	CFPS	PCR
0	000020	동화약품	KOSPI	2022Q1	consolid...	1564.97...	8.72219
1	000040	KR모터스	KOSPI	2022Q1	consolid...	-89.21609	-9.96457
2	000050	경방	KOSPI	2022Q1	consolid...	1306.31...	12.20991
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	9186.15...	5.16538
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	23586.0...	3.65047
5	000080	하이트진로	KOSPI	2022Q1	consolid...	4385.84...	8.53884
6	000100	유한양행	KOSPI	2022Q1	consolid...	27.07852	2156.69...
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	10379.0...	12.47708
8	000140	하이트진...	KOSPI	2022Q1	consolid...	12766.0...	1.06924
9	000150	두산	KOSPI	2022Q1	consolid...	1610.82...	64.56321
10	000180	성창기업...	KOSPI	2022Q1	consolid...	137.272...	19.08608
11	000210	DL	KOSPI	2022Q1	consolid...	3658.97...	16.72601
12	000220	유유제약	KOSPI	2022Q1	consolid...	333.036...	24.29164
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	-2274.8...	-9.58293
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	367.176...	39.89906
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	-395.94...	-90.66807
16	000270	기아	KOSPI	2022Q1	consolid...	15898.3...	4.65457
17	000300	대유플러스	KOSPI	2022Q1	consolid...	-249.44...	-5.23153
18							
19							
20							

4. 종목 추출 모듈 – 2Combo

7) 가치지표 결합하기(PER, PBR)



- get2Combo

- PER, PBR
- 2Combo: PER+PBR의 절대, 상대 점수 합

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터, 상대/절대 점수, 2Combo 합계 점수

4. 종목 추출 모듈 – 2Combo

7) 가치지표 결합하기

```
def getFiltered(self, by, floor=None, cap=None, asc=True):
    # floor, cap으로 필터링 해준다.
    if floor is not None and cap is not None:
        filtered = self.Factor.loc[(self.Factor[by] >= floor) & (self.Factor[by] < cap)]
    elif floor is not None and cap is None:
        filtered = self.Factor.loc[(self.Factor[by] >= floor)]
    elif floor is None and cap is not None:
        filtered = self.Factor.loc[(self.Factor[by] < cap)]
    else:
        filtered = self.Factor

    filtered = filtered.sort_values(by=by, ascending=asc)
    filtered = filtered.dropna()  # 결측치 제거
    filtered = filtered.reset_index(drop=True)  # 인덱스 리셋
```

```
def getScore(self, mod, by, floor=None, cap=None, asc=True):
    calc = self.getFiltered(by, floor, cap, asc)
    calc[by + '_score'] = np.NaN  # 팩터의 상대 점수를 저장할 컬럼
    # 상대 점수 계산 함수
    if mod.upper() == 'RELATIVE':
        # 전체 종목에 대하여 순위를 매기고 이에 대하여 상대 점수를 매긴다.
        for i in range(len(calc)):
            calc[by + '_score'].iloc[i] = 100 - (((calc.index[i] + 1) / len(calc)) * 100)

    # 절대점수 계산 floor, cap 존재 필요
    if mod.upper() == 'ABSOLUTE':
        div = cap - floor
        for i in range(len(calc)):
            temp = int((calc[by].iloc[i] - floor) / div * 1000)
            if temp < 0 or temp == np.inf:
                temp = 0
            calc[by + '_score'].iloc[i] = (1000 - temp) / 10
    return calc[['종목코드', '종목명', '시장', '기간', '컬럼', by + '_score']]
```

- getFiltered

필요한 팩터를 검색해주는 함수 이다.

필요한 지표에 대하여 floor과 cap 사이의 데이터만 추려서 반환해준다.

- getScore

Relative: 상대점수 계산 지표에 대하여 순서대로 점수를 매긴다.

Absolute: 절대점수 계산 0.1당 0.1점씩 차이나게 점수를 부여한다.

4. 종목 추출 모듈 – 2Combo

7) 가치지표 결합하기

```python
# 가치지표 결합하기(2개 팩터 PER, PBR) 상대, 절대점수 산정
def get2Combo(self, mod, n=None):
    self.getPER()
    self.getPBR()
    s3 = pd.DataFrame()
    # 상대점수 계산
    if mod.upper() == 'RELATIVE':
        s1 = self.getScore(mod=mod, by='PER', floor=1, cap=10, asc=True)
        s2 = self.getScore(mod=mod, by='PBR', floor=0.1, cap=1, asc=True)
        s3 = pd.merge(s1, s2, how='inner')
        s3['2Combo'] = s3['PER_score'] + s3['PBR_score']
    # 절대점수 계산
    elif mod.upper() == 'ABSOLUTE':
        s1 = self.getScore(mod=mod, by='PER', floor=1, cap=10, asc=True)
        s2 = self.getScore(mod=mod, by='PBR', floor=0.1, cap=1, asc=True)
        s3 = pd.merge(s1, s2, how='inner')
        s3['2Combo'] = s3['PER_score'] + s3['PBR_score']

    if n is None:
        return s3
    else:
        return s3.head(n)
```

- get2Combo

1. PER, PBR의 상대/절대 점수 계산
2. 두 지표의 교집합 데이터를 생성
3. PER, PBR 점수의 합을 계산
4. 2Combo DataFrame 반환

4. 종목 추출 모듈

7) 2Combo 절대평가

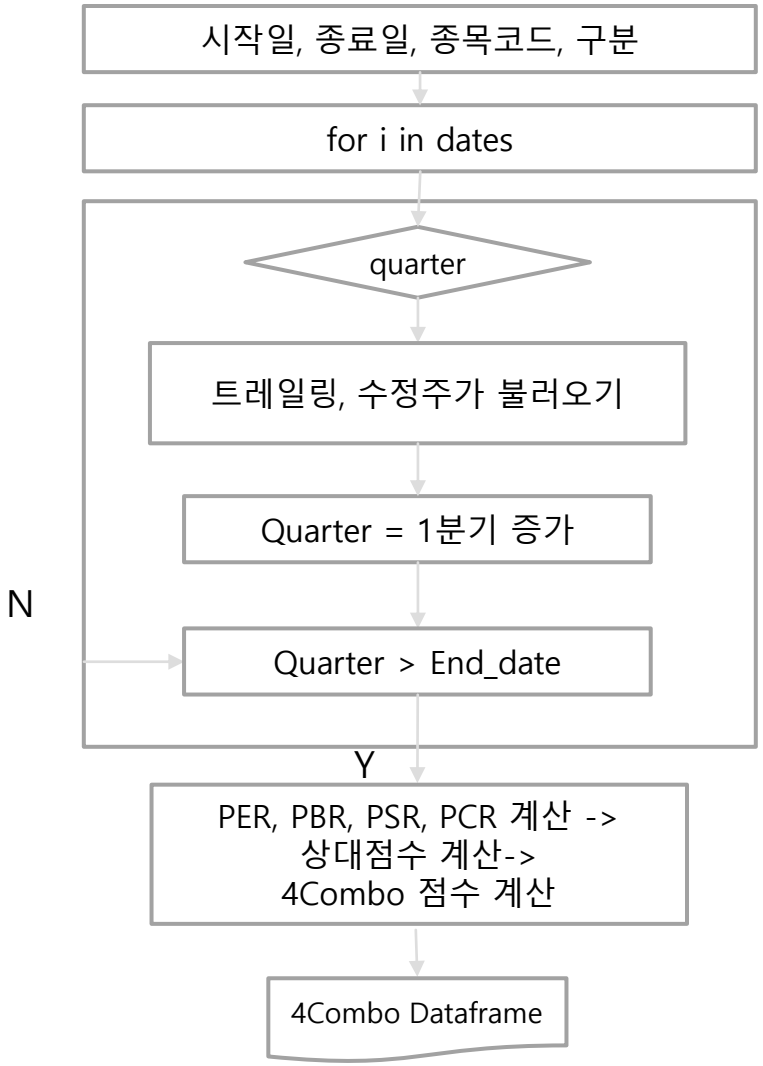
	종목코드	종목명	시장	기간	컬럼	PER_score	PBR_score	2Combo
0	124560	태웅로직스	KOSDAQ	2022Q1	consolid...	98.80000	88.50000	187.30000
1	084650	랩지노믹스	KOSDAQ	2022Q1	consolid...	97.40000	86.50000	183.90000
2	016250	SGC이테...	KOSDAQ	2022Q1	consolid...	95.60000	94.00000	189.60000
3	041190	우리기술...	KOSDAQ	2022Q1	consolid...	95.10000	68.80000	163.90000
4	036710	심텍홀딩스	KOSDAQ	2022Q1	consolid...	94.70000	96.30000	191.00000
5	095270	웨이브일...	KOSDAQ	2022Q1	consolid...	93.20000	66.10000	159.30000
6	104480	티케이케...	KOSDAQ	2022Q1	consolid...	93.00000	77.50000	170.50000
7	011200	HMM	KOSPI	2022Q1	consolid...	92.10000	68.00000	160.10000
8	140520	대창스틸	KOSDAQ	2022Q1	consolid...	92.10000	90.90000	183.00000
9	950130	엑세스바...	KOSDAQ	2022Q1	consolid...	90.60000	54.50000	145.10000
10	001230	동국제강	KOSPI	2022Q1	consolid...	89.10000	94.70000	183.80000
11	001120	LX인터내...	KOSPI	2022Q1	consolid...	88.90000	94.90000	183.80000
12	014790	한라	KOSPI	2022Q1	consolid...	88.30000	96.00000	184.30000
13	205470	휴마시스	KOSDAQ	2022Q1	consolid...	88.30000	19.20000	107.50000
14	011300	성안	KOSPI	2022Q1	consolid...	88.00000	81.50000	169.50000
15	298020	효성티앤씨	KOSPI	2022Q1	consolid...	87.20000	74.80000	162.00000
16	004720	팜젠사이...	KOSPI	2022Q1	consolid...	87.20000	80.20000	167.40000
17	039560	다산네트...	KOSDAQ	2022Q1	consolid...	86.10000	88.90000	175.00000

7) 2Combo

	종목코드	종목명	시장	기간	컬럼	PER_score	PBR_score	2Combo
0	124560	태웅로직스	KOSDAQ	2022Q1	consolid...	99.77679	76.47887	176.25566
1	084650	랩지노믹스	KOSDAQ	2022Q1	consolid...	99.55357	72.46479	172.01836
2	016250	SGC이테...	KOSDAQ	2022Q1	consolid...	98.88393	87.39437	186.27829
3	041190	우리기술...	KOSDAQ	2022Q1	consolid...	98.43750	44.01408	142.45158
4	036710	심텍홀딩스	KOSDAQ	2022Q1	consolid...	98.21429	92.46479	190.67907
5	095270	웨이브일...	KOSDAQ	2022Q1	consolid...	97.76786	41.05634	138.82420
6	104480	티케이케...	KOSDAQ	2022Q1	consolid...	97.32143	55.35211	152.67354
7	011200	HMM	KOSPI	2022Q1	consolid...	95.75893	43.16901	138.92794
8	140520	대창스틸	KOSDAQ	2022Q1	consolid...	95.53571	81.19718	176.73290
9	950130	엑세스바...	KOSDAQ	2022Q1	consolid...	94.41964	28.59155	123.01119
10	001230	동국제강	KOSPI	2022Q1	consolid...	93.52679	88.87324	182.40003
11	001120	LX인터내...	KOSPI	2022Q1	consolid...	93.30357	89.01408	182.31766
12	014790	한라	KOSPI	2022Q1	consolid...	93.08036	91.90141	184.98177
13	205470	휴마시스	KOSDAQ	2022Q1	consolid...	92.85714	6.19718	99.05433
14	011300	성안	KOSPI	2022Q1	consolid...	92.41071	62.81690	155.22762
15	298020	효성티앤씨	KOSPI	2022Q1	consolid...	92.18750	51.69014	143.87764
16	004720	팜젠사이...	KOSPI	2022Q1	consolid...	91.96429	60.21127	152.17555
17	039560	다산네트...	KOSDAQ	2022Q1	consolid...	91.29464	77.04225	168.33690
18	005960	동부건설	KOSPI	2022Q1	consolid...	91.07143	90.42254	181.49396

4. 종목 추출 모듈 – 4Combo

8) 가치투자 4대장 콤보



- get4Combo
  - PER, PBR, PSR, PCR
  - 4Combo: PER+PBR+PSR+PCR 상대점수 합

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – 4Combo

8) 가치투자 4대장 콤보

```
# 필요 지표 계산
self.getPER()
self.getPBR()
self.getPSR()
self.getPCR()
# 가치 지표의 상대 점수 계산
s1 = self.getScore('relative', by='PER', floor=1, asc=True)
s2 = self.getScore('relative', by='PBR', floor=0.1, asc=True)
s3 = self.getScore('relative', by='PSR', floor=0.1, asc=True)
s4 = self.getScore('relative', by='PCR', floor=0.1, asc=True)
df = [s1, s2, s3, s4]
s5 = reduce(lambda left, right: pd.merge(left, right, on=['종목코드', '종목명', '시장', '기간', '컬럼'], how='inner'), df)
s5['4Combo'] = s5['PER_score'] + s5['PBR_score'] + s5['PSR_score'] + s5['PCR_score']
s5.sort_values(by=['4Combo'])
s5.reset_index(drop=True)
if n is not None:
    s5 = s5.head(n)
if not asc:
    s5 = s5.tail(n)
return s5
```

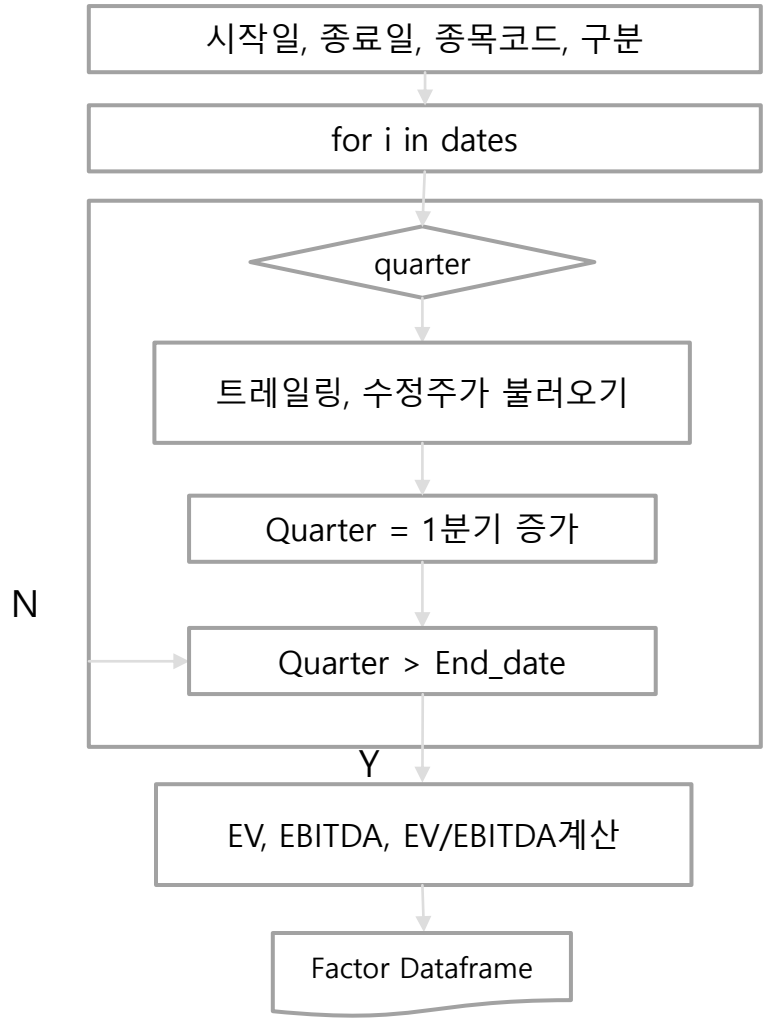
4. 종목 추출 모듈

8) 4Combo

	종목코드	종목명	시장	기간	컬럼	PER_score	PBR_score	PSR_score	PCR_score	4Comb
1	036710	심텍홀딩스	KOSDAQ	2022Q1	consolid...	99.36958	93.60430	97.04377	96.69278	386.71044
6	014790	한라	KOSPI	2022Q1	consolid...	97.55713	93.12612	97.15748	94.51697	382.35770
64	011560	세보엠이씨	KOSDAQ	2022Q1	consolid...	86.76123	96.47340	98.23764	96.43168	377.90395
65	016880	웅진	KOSPI	2022Q1	consolid...	86.68243	89.36043	98.46504	98.43342	372.94131
54	003960	사조대림	KOSPI	2022Q1	consolid...	88.25847	94.26181	97.72598	92.16710	372.41336
20	002990	금호건설	KOSPI	2022Q1	consolid...	94.79905	87.92588	94.48550	94.25587	371.46631
5	001230	동국제강	KOSPI	2022Q1	consolid...	97.71474	90.55589	94.20125	87.20627	369.67814
34	023410	유진기업	KOSDAQ	2022Q1	consolid...	91.33176	97.90795	89.59636	90.60052	369.43659
123	009410	태영건설	KOSPI	2022Q1	consolid...	79.43262	92.16975	98.01023	99.04265	368.65526
26	005490	POSCO홀...	KOSPI	2022Q1	consolid...	93.45942	96.53317	88.06140	85.29156	363.34555
24	278650	노터스	KOSDAQ	2022Q1	consolid...	93.77463	97.72863	81.01194	90.68755	363.20275
126	034220	LG디스플...	KOSPI	2022Q1	consolid...	79.11742	94.79976	91.47243	97.73716	363.12677
30	035890	서희건설	KOSDAQ	2022Q1	consolid...	91.72577	87.08906	87.77715	95.47433	362.06630
38	003070	코오롱글...	KOSPI	2022Q1	consolid...	90.54374	74.29767	99.31779	95.99652	360.15572
94	004170	신세계	KOSPI	2022Q1	consolid...	83.21513	97.78840	84.53667	93.38555	358.92576
47										
98										
40										

4. 종목 추출 모듈 – EV/EBITDA

9) EV/EBITDA 계산 함수



- getEVEBITDA
  - EV = 시가총액 + (단기차입금+장기차입금) - 현금성자산
  - EBITDA = 영업이익 + 감가상각비
  - EVEBITDA = EV / EBITDA

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – EV/EBITDA

9) EV/EBITDA

```
def getEVEBITDA(self):
    self.Factor['EV'] = np.NaN
    self.Factor['EBITDA'] = np.NaN
    self.Factor['EV_EBITDA'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getSelectedData(i)
        if not valid:
            continue
        self.Factor['EV'].iloc[i] = pf['시가총액'].iloc[0] + (tf['단기차입금'].iloc[0] + tf['장기차입금'].iloc[0] - tf['기말현금및현금성자산'].iloc[0]) * getFactor.won
        self.Factor['EBITDA'].iloc[i] = (tf['영업이익'].iloc[0] + tf['감가상각비'].iloc[0]) * getFactor.won
        self.Factor['EV_EBITDA'].iloc[i] = self.Factor['EV'].iloc[i] / self.Factor['EBITDA'].iloc[0]

    self.Factor.replace([np.inf, -np.inf], np.NaN)
    return self.Factor.set_index('종목코드')
```

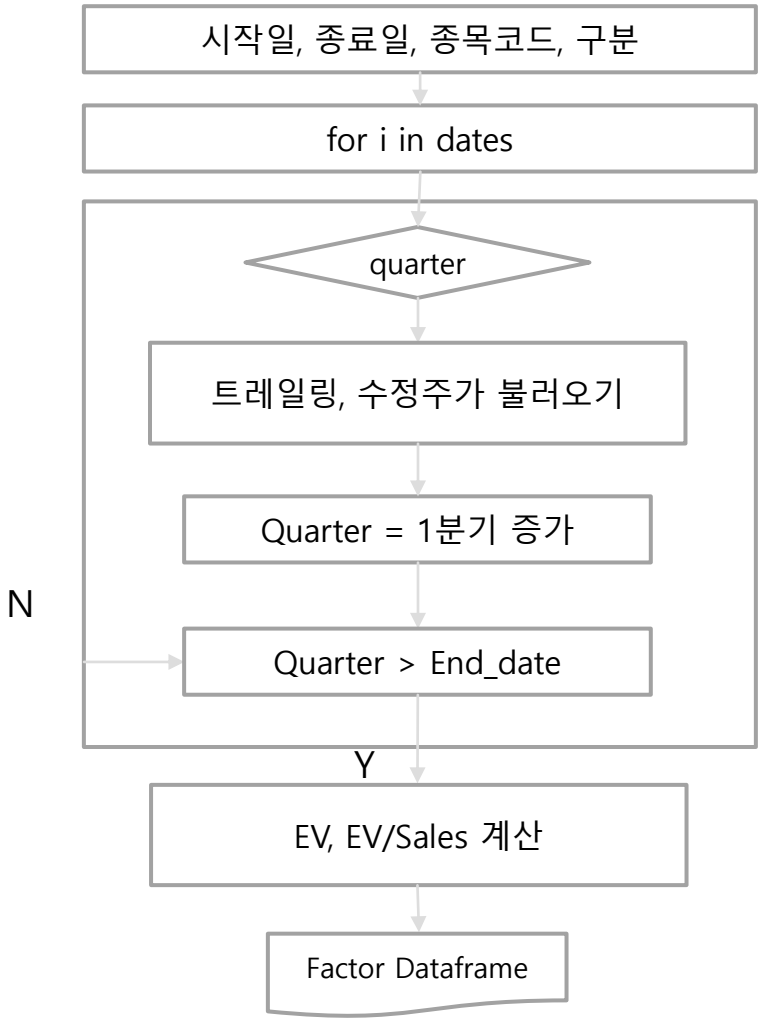
4. 종목 추출 모듈

9) EV/EBITDA

	종목코드	종목명	시장	기간	컬럼	EV	EBITDA	EV/EBITDA
0	000020	동화약품	KOSPI	2022Q1	consolid...	1979065...	37138000...	5.32895
1	000040	KR모터스	KOSPI	2022Q1	consolid...	3039612...	19900000...	0.81846
2	000050	경방	KOSPI	2022Q1	consolid...	1182981...	99615000...	31.85367
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	5723656...	10256900...	154.11859
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	1378534...	40392400...	37.11925
5	000080	하이트진로	KOSPI	2022Q1	consolid...	2678024...	31458100...	72.11010
6	000100	유한양행	KOSPI	2022Q1	consolid...	3854812...	84521000...	103.79698
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	4224698...	79685000...	113.75676
8	000140	하이트진...	KOSPI	2022Q1	consolid...	1483513...	32990100...	39.94597
9	000150	두산	KOSPI	2022Q1	consolid...	1688487...	12658310...	454.65235
10	000180	성창기업...	KOSPI	2022Q1	consolid...	3519081...	11569000...	9.47569
11	000210	DL	KOSPI	2022Q1	consolid...	6091380...	32088800...	164.02014
12	000220	유유제약	KOSPI	2022Q1	consolid...	-219262...	55020000...	-0.59040
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	4345637...	-5474100...	11.70132
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	1559212...	23430099...	41.98428
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	6363576...	-1131000...	17.13495
16	000270	기아	KOSPI	2022Q1	consolid...	3879227...	73175500...	104.45440
17	000300	대유플러스	KOSPI	2022Q1	consolid...	4214214...	29763000...	11.34744
18								
19								
20								

4. 종목 추출 모듈 – EV/Sales

10) EV/Sales 계산 함수



- getEV/Salses
- EV = 시가총액 + (단기차입금+장기차입금) - 현금성자산
- EV_Sales = EV / 매출액

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – EV/Sales

10) EV/Sales

```
def getEVSales(self):
    self.Factor['EV'] = np.NaN
    self.Factor['EV_Sales'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getSelectedData(i)
        if not valid:
            continue
        self.Factor['EV'].iloc[i] = pf['시가총액'].iloc[0] + (tf['단기차입금'].iloc[0] + tf['장기차입금'].iloc[0] - tf['기말현금및현금성자산'].iloc[0]) * getFactor.won
        self.Factor['EV_Sales'].iloc[i] = self.Factor['EV'].iloc[i] / (tf['매출액'].iloc[0] * getFactor.won)

    self.Factor.replace([np.inf, -np.inf], np.NaN)
    return self.Factor.set_index('종목코드')
```

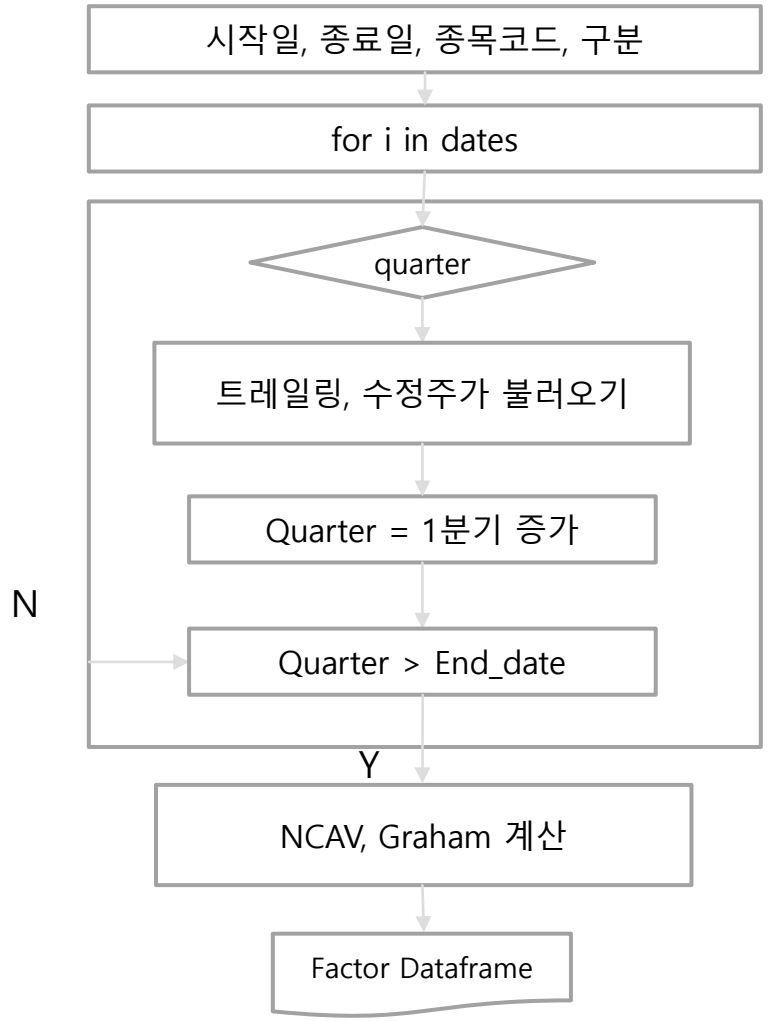
4. 종목 추출 모듈

10) EV/Sales

	종목코드	종목명	시장	기간	컬럼	EV	EV/Sales
0	000020	동화약품	KOSPI	2022Q1	consolid...	1979065...	0.64573
1	000040	KR모터스	KOSPI	2022Q1	consolid...	3039612...	0.21717
2	000050	경방	KOSPI	2022Q1	consolid...	1182981...	2.97552
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	5723656...	inf
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	1378534...	0.42693
5	000080	하이트진로	KOSPI	2022Q1	consolid...	2678024...	1.18943
6	000100	유한양행	KOSPI	2022Q1	consolid...	3854812...	2.24165
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	4224698...	0.36711
8	000140	하이트진...	KOSPI	2022Q1	consolid...	1483513...	0.66218
9	000150	두산	KOSPI	2022Q1	consolid...	1688487...	1.23302
10	000180	성창기업...	KOSPI	2022Q1	consolid...	3519081...	1.70967
11	000210	DL	KOSPI	2022Q1	consolid...	6091380...	2.20812
12	000220	유유제약	KOSPI	2022Q1	consolid...	-219262...	-0.17585
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	4345637...	0.70260
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	1559212...	1.56463
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	6363576...	3.75366
16	000270	기아	KOSPI	2022Q1	consolid...	3879227...	0.05415
17	000300	대유플러스	KOSPI	2022Q1	consolid...	4214214...	0.82970
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	2850252...	0.29091
19	000370	한화손해...	KOSPI	2022Q1	consolid...	8404596...	0.15814
20	000390	삼화페인트	KOSPI	2022Q1	consolid...	6384832...	0.98057
21							
22							
23							

4. 종목 추출 모듈 – 그레이엄(NCAV)

11) 그레이엄 투자법



- getGraham
  - NCAV = 유동자산 – 부채
  - Graham(안전마진) = NCAV – (시가총액 * 1.5)

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – 그레이엄(NCAV)

11) 그레이엄 투자법

```
def getGraham(self):
    self.Factor['Graham'] = np.NaN
    self.Factor['NCAV'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getSelectedData(i)
        if not valid:
            continue
        self.Factor['NCAV'].iloc[i] = (tf['유동자산'].iloc[0] - tf['부채'].iloc[0]) * getFactor.won
        self.Factor['Graham'].iloc[i] = self.Factor['NCAV'].iloc[i] - (pf['시가총액'].iloc[i] * 1.5)

    self.Factor.replace([np.inf, -np.inf], np.NaN)
    return self.Factor.set_index('종목코드')
```

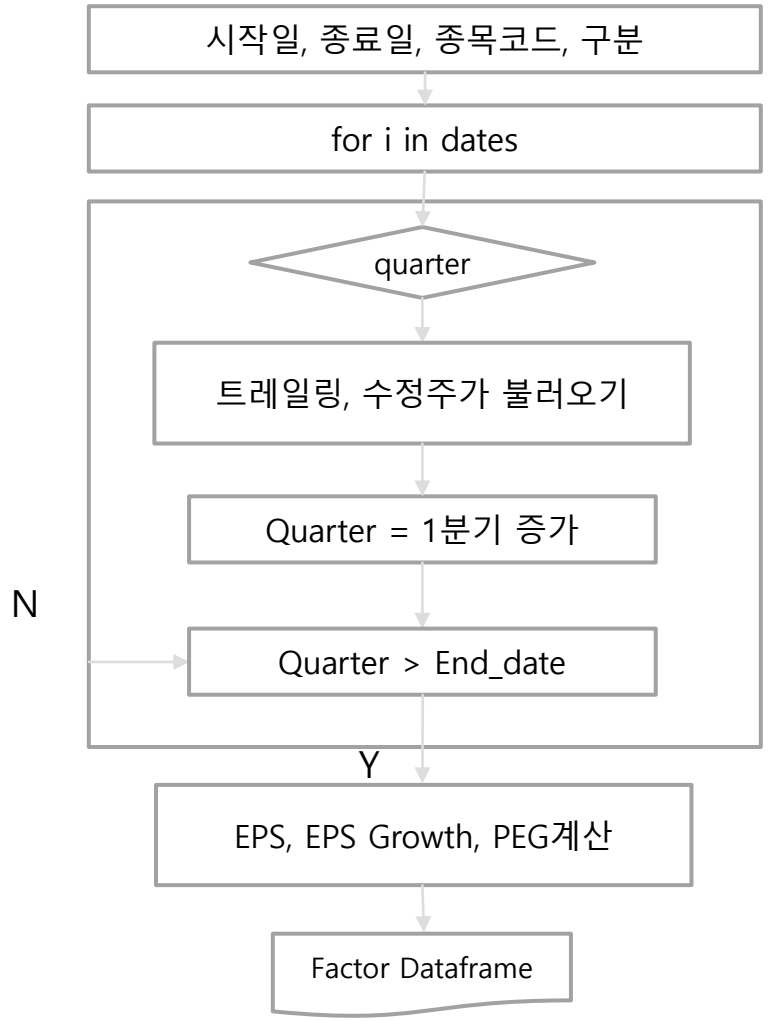
4. 종목 추출 모듈

11) 그레이엄 투자법

	종목코드	종목명	시장	기간	컬럼	NCAV	Graham
0	000020	동화약품	KOSPI	2022Q1	consolid...	5161290...	-55767848...
1	000040	KR모터스	KOSPI	2022Q1	consolid...	-206721...	-33492118...
2	000050	경방	KOSPI	2022Q1	consolid...	-158406...	-22399743...
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	-997191...	-10830463...
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	-117446...	-22805455...
5	000080	하이트진로	KOSPI	2022Q1	consolid...	-506272...	-90024755...
6	000100	유한양행	KOSPI	2022Q1	consolid...	2784650...	-36304272...
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	-105034...	-14934747...
8	000140	하이트진...	KOSPI	2022Q1	consolid...	-742298...	-78981385...
9	000150	두산	KOSPI	2022Q1	consolid...	-289447...	-31522418...
10	000180	성창기업...	KOSPI	2022Q1	consolid...	-355239...	-62936278...
11	000210	DL	KOSPI	2022Q1	consolid...	-839003...	-10313790...
12	000220	유유제약	KOSPI	2022Q1	consolid...	1458960...	-63218579...
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	-120607...	-15834410...
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	1038479...	-10477218...
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	3117789...	-91428000...
16	000270	기아	KOSPI	2022Q1	consolid...	-102210...	-55216331...
17	000300	대유플러스	KOSPI	2022Q1	consolid...	-561859...	-79871510...
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	9542999...	-15178540...
19	000370	한화손해...	KOSPI	2022Q1	consolid...	-535980...	-62405951...
20							
21							
22							

4. 종목 추출 모듈 – PEG 기반 종목 추출

12) PEG 기반 종목 추출



- getPEG
  - EPS = 당기순이익 / 상장주식수
  - EPS Growth = (EPS – 4분기전 EPS) / (4분기전 EPS) * 100
  - PEG = (주가 / EPS) / EPS Growth

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – PEG 기반 종목 추출

12) PEG 기반 종목 추출

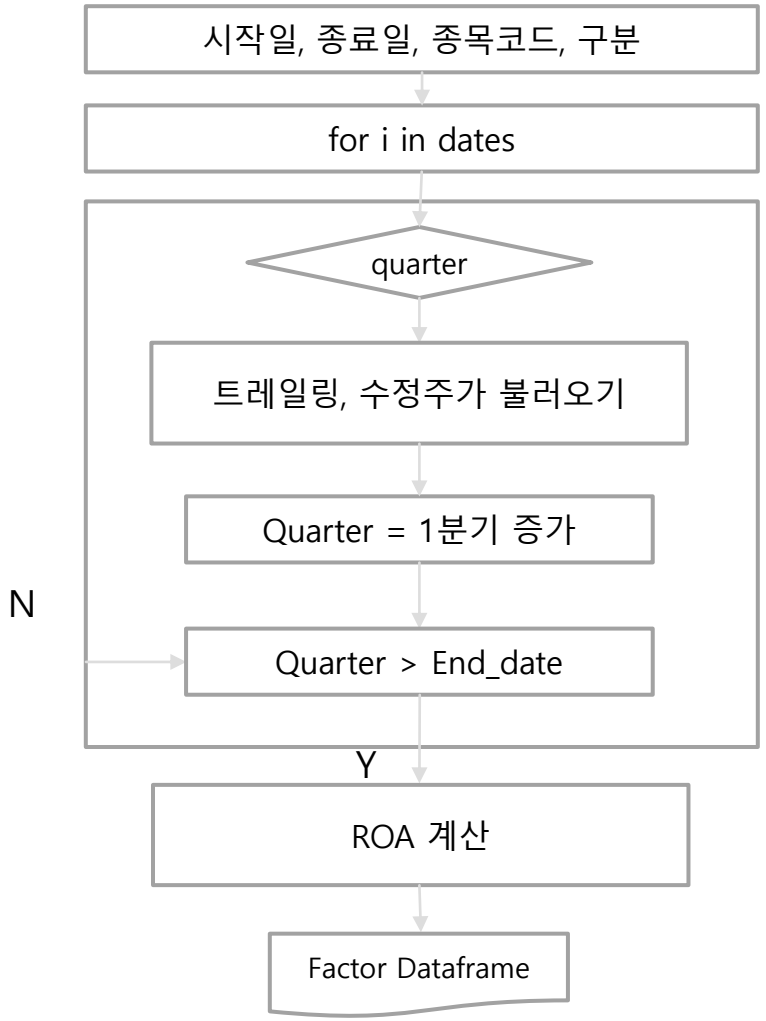
```
# 주가수익성장비율 PEG
def getPEG(self):
    def get4PreQ(period):  # 4분기전 데이터
        return str(int(period[0:5]) - 1) + period[5:]

    self.Factor['EPS Growth'] = np.NaN
    self.Factor['PEG'] = np.NaN
    self.getPER()
    for i in range(0, len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue

        sq4 = get4PreQ(self.Factor['기간'].iloc[i])  # 4분기 이전의 기간
        # 4분기 이전 팩터
        sff = self.Factor[(self.Factor['종목코드'] == self.Factor['종목코드'].iloc[i]) & (
                self.Factor['기간'] == sq4)]
        # 4분기 이전의 값 존재하지 않을 경우
        if sff.empty:
            continue

        self.Factor['EPS Growth'].iloc[i] = (self.Factor['EPS'].iloc[i] - sff['EPS'].iloc[0]) / abs(
            sff['EPS'].iloc[0]) * 100
        self.Factor['PEG'].iloc[i] = (pf['종가'].iloc[0] / self.Factor['EPS'].iloc[i]) / \
                                     self.Factor['EPS Growth'].iloc[i]

    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈 – 우량주

- 투자효율이 좋은 기업:ROA, ROE
- 잔여이익모델: RIM
- 영업효율이 좋은 기업: GP/A
- 존버할 수 있는 기업: 안정성지표
- 쑥쑥 자라는 기업: 성장률 지표
- 부지런한 기업: 회전율 지표
- 해자가 있는 기업: 이익률 지표

4. 종목 추출 모듈 – ROA

1) 투자효율이 좋은 기업:ROA



- getROA

ROA = 당기순이익 / ((기초자산+기말자산)/2)

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – ROA

1) 투자효율이 좋은 기업:ROA

```
# 자산대비이익 ROA
def getROA(self):
    self.Factor['ROA'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['ROA'].iloc[i] = tf['당기순이익'].iloc[0] / (
                (tf['기초현금및현금성자산'].iloc[0] + tf['기말현금및현금성자산'].iloc[0]) / 2)

    return self.Factor.set_index('종목코드')
```

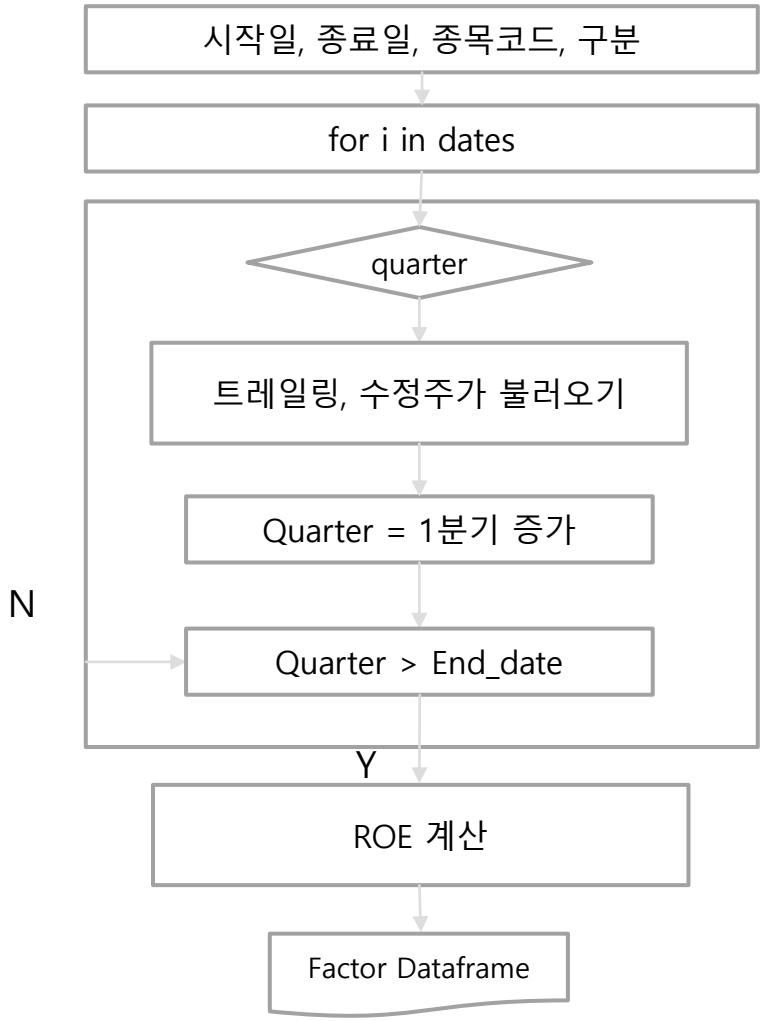
4. 종목 추출 모듈

1) 투자효율이 좋은 기업:ROA

	종목코드	종목명	시장	기간	컬럼	ROA
0	000020	동화약품	KOSPI	2022Q1	consolid...	0.09594
1	000040	KR모터스	KOSPI	2022Q1	consolid...	-0.13813
2	000050	경방	KOSPI	2022Q1	consolid...	0.39044
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	inf
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	0.28542
5	000080	하이트진로	KOSPI	2022Q1	consolid...	0.04066
6	000100	유한양행	KOSPI	2022Q1	consolid...	0.09737
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	0.11273
8	000140	하이트진...	KOSPI	2022Q1	consolid...	0.03820
9	000150	두산	KOSPI	2022Q1	consolid...	0.02621
10	000180	성창기업...	KOSPI	2022Q1	consolid...	-0.01356
11	000210	DL	KOSPI	2022Q1	consolid...	0.11661
12	000220	유유제약	KOSPI	2022Q1	consolid...	-0.00791
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	-0.34540
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	0.54462
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	-0.01720
16	000270	기아	KOSPI	2022Q1	consolid...	0.11015
17	000300	대유플러스	KOSPI	2022Q1	consolid...	0.01705
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	0.01677
19	000370	한화손해...	KOSPI	2022Q1	consolid...	0.00471
20	000390	삼화페인트	KOSPI	2022Q1	consolid...	-0.01498
21						
22						
23						

4. 종목 추출 모듈 – ROE

2) 자본대비이익: ROE



- getROE

ROE = 당기순이익 / ((자본+4분기 전 자본)/2)

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

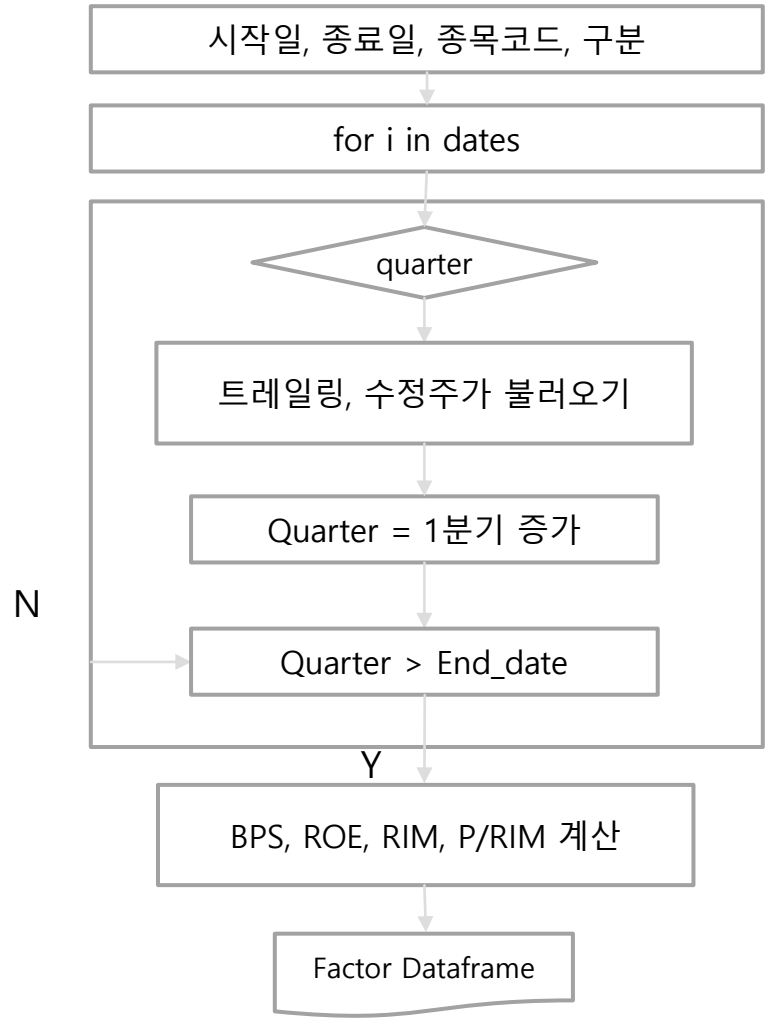
4. 종목 추출 모듈 – ROE

1) 투자효율이 좋은 기업:ROE

```
# 자본대비이익 ROE
def getROE(self):
    def get4PreQ(period):
        return str(int(period[0:5]) - 1) + period[5:]

    self.Factor['ROE'] = np.NaN
    for i in range(0, len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        ''' 4분기 전의 값 존재시
        sq4 = get4PreQ(self.Factor['기간'].iloc[i]) # 4분기 이전의 기간
        # 4분기 이전 트레일링 데이터
        stf = self.trailingData[(self.trailingData['종목코드'] == self.Factor['종목코드'].iloc[i]) & (
                self.trailingData['기간'] == sq4)]
        # 4분기 이전의 값 존재하지 않을 경우
        if stf.empty:
            continue
        self.Factor['ROE'].iloc[i] = tf['당기순이익'].iloc[0] / ((tf['자본'].iloc[0] + stf['자본'].iloc[0]) / 2)
        '''
        # 4분기 이전 값 미존재
        self.Factor['ROE'].iloc[i] = tf['당기순이익'].iloc[0] / tf['자본'].iloc[0]
    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈 – RIM

3) 잔여이익모델: RIM



- getRIM

BPS = 자본 / 상장주식수

ROE3AVG = ROE 3년간 평균

RIM = BPS * ROE3AVG / con(상수)

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

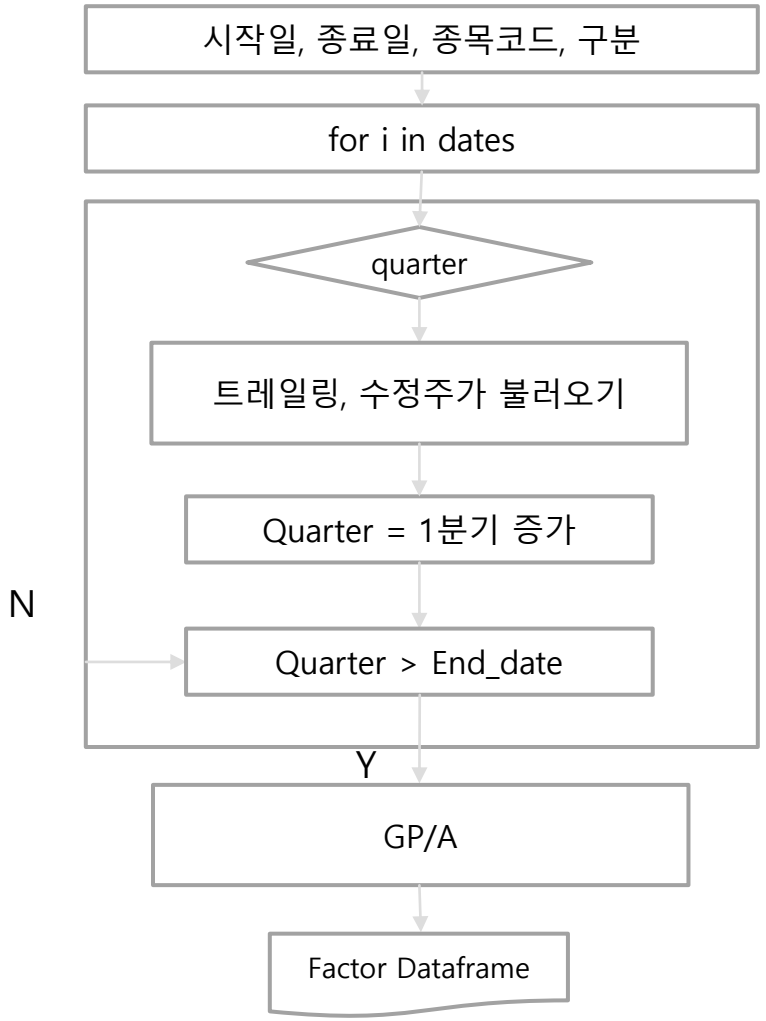
4. 종목 추출 모듈 - RIM

3) 잔여이익모델: RIM

```
# 잔여이익모델 RIM, P/RIM
def getRIM(self, con):
    def getBPS():
        self.Factor['BPS'] = np.NaN
        for i in range(len(self.Factor)):
            (tf, pf, valid) = self.getFilteredRow(i)
            if not valid:
                continue
            self.Factor['BPS'].iloc[i] = (tf['자본'].iloc[0] * self.won) / pf['상장주식수'].iloc[0]

    cons = '{:.2f}'.format(con)
    self.Factor['RIM' + str(cons)[-2:]] = np.NaN
    self.Factor['P/RIM']
    getBPS()
    self.getROE()
    self.Factor['ROE3AVG'] = self.Factor['ROE'].rolling(12).mean()
    self.Factor['RIM' + str(cons)[-2:]] = self.Factor['BPS'] * self.Factor['ROE3AVG'] / con
    self.Factor.loc[(self.Factor['당기순이익'] < 0) | (self.Factor['자본'] < 0)] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['P/RIM'].iloc[i] = pf['종가'].iloc[0] / self.Factor['RIM' + str(cons)[-2:]]

    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈 – GP/A

4) 영업효율이 좋은 기업: GP/A



- getGPA

GP/A = 매출총이익 / 자산

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – GP/A

4) 영업효율이 좋은 기업: GP/A

```
def getGPA(self):
    self.Factor['GP/A'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['GP/A'].iloc[i] = tf['매출총이익'].iloc[0] / tf['자산'].iloc[0]
    return self.Factor.set_index('종목코드')
```

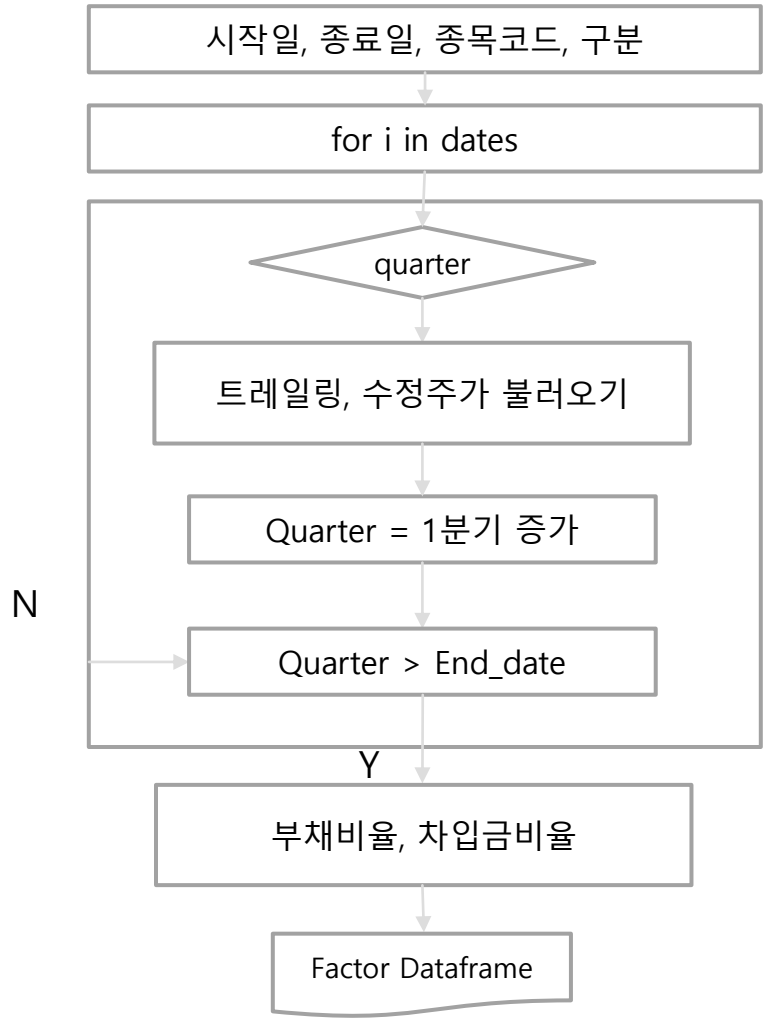
4. 종목 추출 모듈

4) 영업효율이 좋은 기업: GP/A

	종목코드	종목명	시장	기간	컬럼	GP/A
0	000020	동화약품	KOSPI	2022Q1	consolid...	0.08774
1	000040	KR모터스	KOSPI	2022Q1	consolid...	0.02620
2	000050	경방	KOSPI	2022Q1	consolid...	0.02816
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	0.00000
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	0.03998
5	000080	하이트진로	KOSPI	2022Q1	consolid...	0.06611
6	000100	유한양행	KOSPI	2022Q1	consolid...	0.05330
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	0.03036
8	000140	하이트진...	KOSPI	2022Q1	consolid...	0.05696
9	000150	두산	KOSPI	2022Q1	consolid...	0.02032
10	000180	성창기업...	KOSPI	2022Q1	consolid...	0.01093
11	000210	DL	KOSPI	2022Q1	consolid...	0.01403
12	000220	유유제약	KOSPI	2022Q1	consolid...	0.06202
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	0.06221
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	0.01920
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	0.06099
16	000270	기아	KOSPI	2022Q1	consolid...	0.05171
17	000300	대유플러스	KOSPI	2022Q1	consolid...	0.02456
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	0.04373
19	000370	한화손해...	KOSPI	2022Q1	consolid...	0.06780
20	000390	삼화페인트	KOSPI	2022Q1	consolid...	0.04042
21						
22						
23						

4. 종목 추출 모듈 – 안정성지표

5) 존버할 수 있는 기업: 안정성지표



- getStabilityIdx

부채비율 = 부채 / 자본 * 100

차입금비율 = (단기차입금+장기차입금) / 자본 * 100

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – GP/A

5) 존버할 수 있는 기업: 안정성지표

```
# 안정성지표
def getStabilityIdx(self):
    self.Factor['부채비율'] = np.NaN
    self.Factor['차입금비율'] = np.NaN
    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['부채비율'].iloc[i] = tf['부채'].iloc[0] / tf['자본'].iloc[0] * 100
        self.Factor['차입금비율'].iloc[i] = (tf['단기차입금'].iloc[0] + tf['장기차입금'].iloc[0]) / tf['자본'].iloc * 100

    return self.Factor.set_index('종목코드')
```

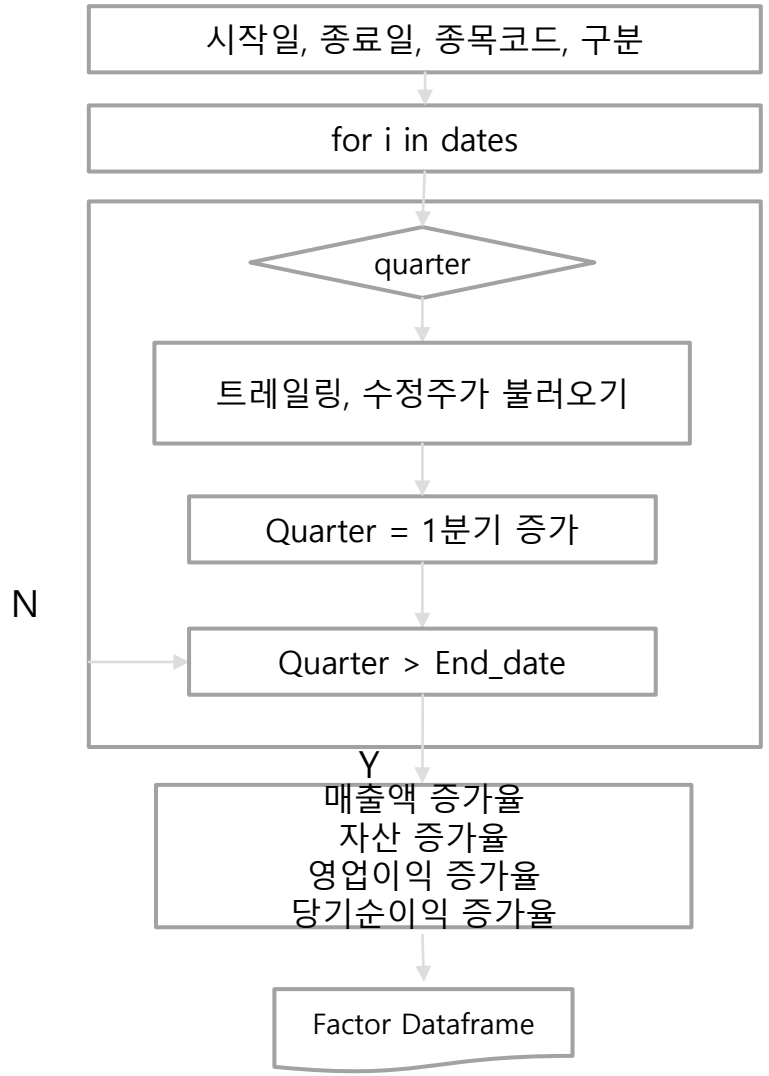
4. 종목 추출 모듈

5) 존버할 수 있는 기업: 안정성지표

	종목코드	종목명	시장	기간	컬럼	부채비율	차입금비율
0	000020	동화약품	KOSPI	2022Q1	consolid...	26.54371	2.23433
1	000040	KR모터스	KOSPI	2022Q1	consolid...	243.19198	17.39828
2	000050	경방	KOSPI	2022Q1	consolid...	69.38768	27.33333
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	1124.32336	0.00000
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	82.26127	15.23357
5	000080	하이트진로	KOSPI	2022Q1	consolid...	228.65606	44.49539
6	000100	유한양행	KOSPI	2022Q1	consolid...	28.93148	5.44789
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	128.63120	17.07529
8	000140	하이트진...	KOSPI	2022Q1	consolid...	284.13608	70.72194
9	000150	두산	KOSPI	2022Q1	consolid...	216.84690	70.41990
10	000180	성창기업...	KOSPI	2022Q1	consolid...	32.00814	11.03696
11	000210	DL	KOSPI	2022Q1	consolid...	100.40660	57.39761
12	000220	유유제약	KOSPI	2022Q1	consolid...	50.32121	8.94512
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	275.62553	87.84406
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	9.49536	3.18236
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	38.82903	4.76064
16	000270	기아	KOSPI	2022Q1	consolid...	93.91341	13.17022
17	000300	대유플러스	KOSPI	2022Q1	consolid...	297.23584	78.37840
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	84.29707	24.47429
19	000370	한화손해...	KOSPI	2022Q1	consolid...	1399.04565	593.39307
20	000390	삼화페인트	KOSPI	2022Q1	consolid...	109.50416	39.25663
21							
22							
23							
24							

4. 종목 추출 모듈 – 성장률 지표

6) 쑥쑥 자라는 기업: 성장률 지표



- getGrowthRate

![지표] = 12분기 전 지표

매출액증가율 = (매출액 + !매출액) / !매출액

자산증가율 = (자산+!자산)/!자산

영업이익증가율=(영업이익+!영업이익)/!영업이익

당기순이익증가율=(당기순이익+!당기순이익)/!당기순이익

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

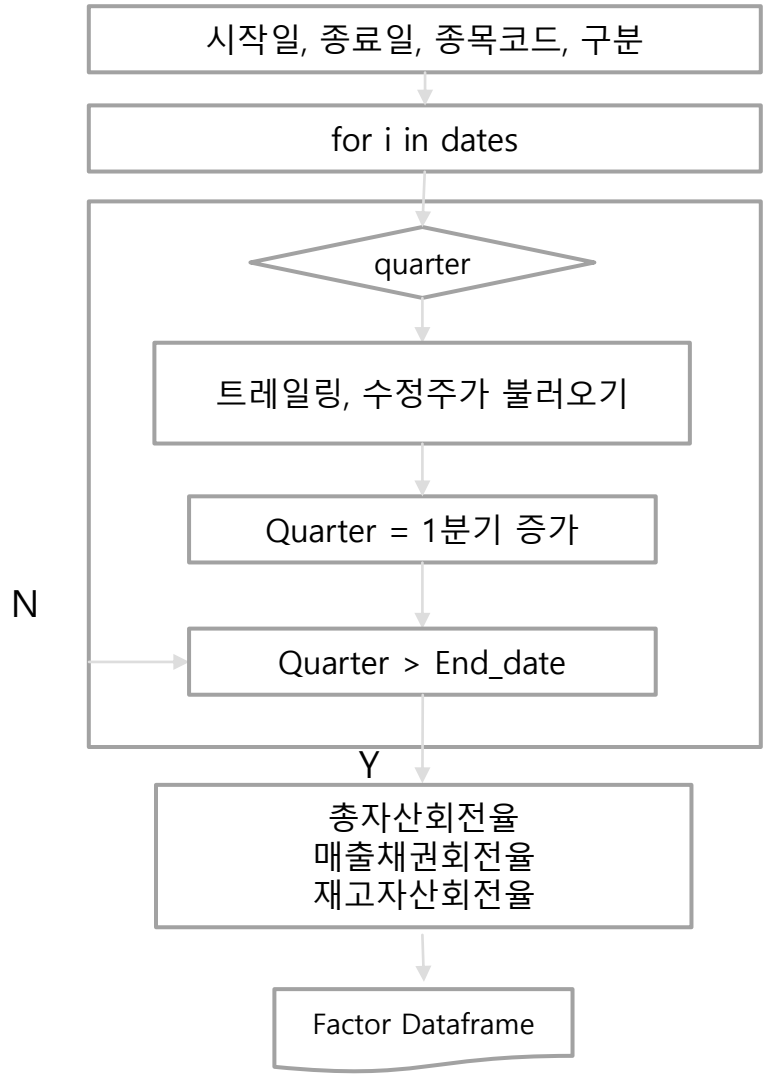
4. 종목 추출 모듈 – 성장률 지표

6) 쑥쑥 자라는 기업: 성장률 지표

```python
# 성장율 지표
def getGrowthRate(self):
    def get12PreQ(period):
        return str(int(period[0:5]) - 3) + period[5:]

    self.Factor['매출액증가율'] = np.NaN
    self.Factor['자산증가율'] = np.NaN
    self.Factor['영업이익증가율'] = np.NaN
    self.Factor['당기순이익증가율'] = np.NaN

    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        sq12 = get12PreQ(self.Factor['기간'].iloc[i])  # 12분기 이전의 기간(3년)
        # 4분기 이전 트레일링 데이터
        stf = self.trailingData[(self.trailingData['종목코드'] == self.Factor['종목코드'].iloc[i]) & (
                self.trailingData['기간'] == sq12)]
        # 4분기 이전의 값 존재하지 않을 경우
        if stf.empty:
            continue
        self.Factor['매출액증가율'].iloc[i] = (tf['매출액'].iloc[0] + stf['매출액'].iloc[0]) / abs(
            stf['매출액'].iloc[0])
        self.Factor['자산증가율'].iloc[i] = (tf['자산'].iloc[0] + stf['자산'].iloc[0]) / abs(
            stf['자산'].iloc[i])
        self.Factor['당기순이익증가율'].iloc[i] = (tf['당기순이익'].iloc[0] + stf['당기순이익'].iloc[0]) / abs(
            stf['당기순이익'].iloc[0])
        self.Factor['영업이익증가율'].iloc[i] = (tf['영업이익'].iloc[0] + stf['영업이익'].iloc[0]) / abs(
            stf['영업이익'].iloc[0])

    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈 – 회전율 지표

7) 부지런한 기업: 회전율 지표



- getTurnoverRatio

![지표] = 4분기 전 지표

총자산회전율 = 매출액 / ((자본+!자본)/2)

매출채권회전율 = 매출액 / ((매출채권수익 +! 매출채권수익)/2)

재고자산회전율 = 매출원가 / ((재고자산+!재고자산)/2)

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

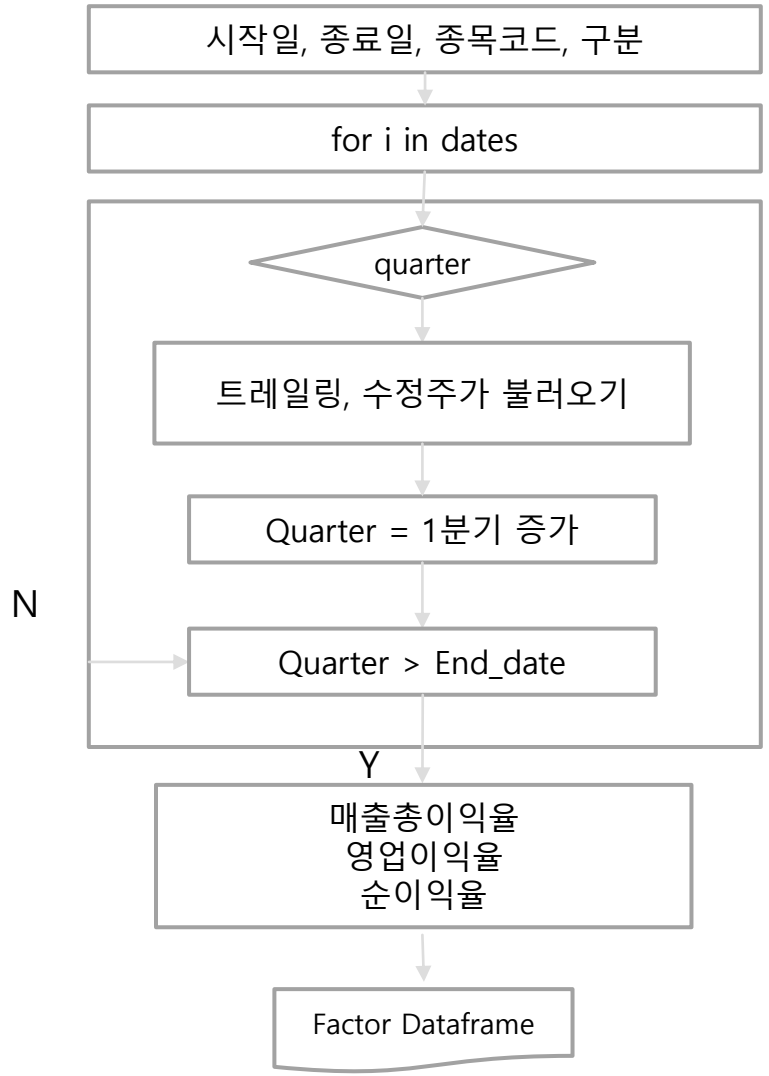
4. 종목 추출 모듈 – 회전율 지표

7) 부지런한 기업: 회전율 지표

```
# 회전율 지표
def getTurnoverRatio(self):
    def get4PreQ(period):
        return str(int(period[0:5]) - 3) + period[5:]

    self.Factor['총자산회전율'] = np.NaN
    self.Factor['매출채권회전율'] = np.NaN
    self.Factor['재고자산회전율'] = np.NaN

    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue

        sq4 = get4PreQ(self.Factor['기간'].iloc[i])  # 4분기 이전의 기간
        # 4분기 이전 트레일링 데이터
        stf = self.trailingData[(self.trailingData['종목코드'] == self.Factor['종목코드'].iloc[i]) & (
                self.trailingData['기간'] == sq4)]
        # 4분기 이전의 값 존재하지 않을 경우
        if stf.empty:
            continue
        self.Factor['총자산회전율'].iloc[i] = tf['매출액'].iloc[0] / ((tf['자본'].iloc[0] + stf['자본'].iloc) / 2)
        self.Factor['매출채권회전율'].iloc[i] = tf['매출액'].iloc[0] / (
                ((tf['매출채권및기타유동채권'].iloc[0] + tf['장기매출채권및기타비유동채권'].iloc[0]) + (
                        stf['매출채권및기타유동채권'].iloc[0] + stf['장기매출채권및기타비유동채권'].iloc[0])) / 2)
        self.Factor['재고자산회전율'].iloc[i] = tf['매출원가'].iloc[0] / ((tf['재고자산'].iloc[0] + stf['재고자산'].iloc[0]) / 2)

    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈 – 이익율 지표

8) 해자가 있는 기업: 이익률 지표



- getMoatIdx

매출총이익율 = 매출총이익 / 매출액

영업이익율 = 영업이익 / 매출액

순이익율 = 당기순이익 / 매출액

Parameters

- Factor 클래스의 변수 사용
- Self.Factor, self.trailingData, self.price_data

Return type

- dataFrame: 종목데이터 및 팩터 반환

4. 종목 추출 모듈 – 이익율 지표

8) 해자가 있는 기업: 이익률 지표

```python
# 해자가 있는 기업
def getMoatIdx(self):
    self.Factor['매출총이익율'] = np.NaN
    self.Factor['영업이익율'] = np.NaN
    self.Factor['순이익율'] = np.NaN

    for i in range(len(self.Factor)):
        (tf, pf, valid) = self.getFilteredRow(i)
        if not valid:
            continue
        self.Factor['매출총이익율'].iloc[i] = tf['매출총이익'].iloc[0] / tf['매출액'].iloc[0]
        self.Factor['영업이익율'].iloc[i] = tf['영업이익'].iloc[0] / tf['매출액'].iloc[0]
        self.Factor['순이익율'].iloc[i] = tf['당기순이익'].iloc[0] / tf['매출액'].iloc[0]

    return self.Factor.set_index('종목코드')
```

4. 종목 추출 모듈

8) 해자가 있는 기업: 이익률 지표

	종목코드	종목명	시장	기간	컬럼	매출총이익율	영업이익율	순이익율
0	000020	동화약품	KOSPI	2022Q1	consolid...	0.51607	0.08624	0.06484
1	000040	KR모터스	KOSPI	2022Q1	consolid...	0.12934	-0.02254	-0.09206
2	000050	경방	KOSPI	2022Q1	consolid...	0.36233	0.14866	0.07689
3	000060	메리츠화재	KOSPI	2022Q1	consolid...	nan	inf	inf
4	000070	삼양홀딩스	KOSPI	2022Q1	consolid...	0.21827	0.09832	0.07385
5	000080	하이트진로	KOSPI	2022Q1	consolid...	0.41798	0.07963	0.03540
6	000100	유한양행	KOSPI	2022Q1	consolid...	0.30559	0.02372	0.05529
7	000120	CJ대한통운	KOSPI	2022Q1	consolid...	0.09587	0.03228	0.01487
8	000140	하이트진...	KOSPI	2022Q1	consolid...	0.42355	0.08645	0.03384
9	000150	두산	KOSPI	2022Q1	consolid...	0.16506	0.06329	0.01966
10	000180	성창기업...	KOSPI	2022Q1	consolid...	0.15231	0.02165	-0.00500
11	000210	DL	KOSPI	2022Q1	consolid...	0.18119	0.06822	0.23878
12	000220	유유제약	KOSPI	2022Q1	consolid...	0.38643	0.01055	-0.01232
13	000230	일동홀딩스	KOSPI	2022Q1	consolid...	0.36817	-0.12432	-0.39026
14	000240	한국앤컴...	KOSPI	2022Q1	consolid...	0.31309	0.20527	0.17877
15	000250	삼천당제약	KOSDAQ	2022Q1	consolid...	0.50173	-0.04409	-0.02431
16	000270	기아	KOSPI	2022Q1	consolid...	0.19015	0.07811	0.06642
17	000300	대유플러스	KOSPI	2022Q1	consolid...	0.10879	0.03181	0.00575
18	000320	노루홀딩스	KOSPI	2022Q1	consolid...	0.19189	0.02614	0.00775
19	000370	한화손해...	KOSPI	2022Q1	consolid...	0.10304	0.00309	0.00221
20	000390	삼화페인트	KOSPI	2022Q1	consolid...	0.15382	0.00439	-0.00271

4. 종목 추출 모듈 – 이익율 지표

```python
# 결측치 처리
def setMissingValue(self, by=None):
    if by is None:
        column = self.Factor.columns[5:].tolist()
        for c in column:
            self.Factor.loc[self.Factor[c] < 0, c] = 0
        self.Factor = self.Factor.replace([np.inf, -np.inf], 0)
        self.Factor = self.Factor.dropna()
        self.Factor = self.Factor.reset_index(drop=True)
    else:
        self.Factor.loc[self.Factor[by] < 0, by] = 0
        self.Factor[by] = self.Factor[by].replace([np.inf, -np.inf], 0)
        self.Factor = self.Factor.dropna()
        self.Factor = self.Factor.reset_index(drop=True)
```

- setMissingValue

팩터값의 결측치를 처리

By='지표' 입력시 하나의 지표에 대해서만 처리

음수값에 대하여 0으로 조정 및 np.inf, -np.inf 값을 0으로 만들어준다. 또한 np.NaN 결측치가 있는 종목을 삭제한다.

감사합니다.