In [104]:

```
!pip install plotly
```

```
Collecting plotly
  Downloading plotly-4.10.0-py2.py3-none-any.whl (13.0 MB)
     |████████████████████████████████| 13.0 MB 7.9 MB/s
eta 0:00:01
Collecting retrying>=1.3.3
  Downloading retrying-1.3.3.tar.gz (10 kB)
Requirement already satisfied: six in /Users/jaeyoungshin/opt/anaconda
3/lib/python3.8/site-packages (from plotly) (1.15.0)
Building wheels for collected packages: retrying
  Building wheel for retrying (setup.py) ... done
  Created wheel for retrying: filename=retrying-1.3.3-py3-none-any.whl
size=11430 sha256=3b1e070d6b7883d1bf29cea65e9288aba3eb1716cddb4607fc36
c913dc6137ec
  Stored in directory: /Users/jaeyoungshin/Library/Caches/pip/wheels/c
4/a7/48/0a434133f6d56e878ca511c0e6c38326907c0792f67b476e56
Successfully built retrying
Installing collected packages: retrying, plotly
Successfully installed plotly-4.10.0 retrying-1.3.3
```

In [1]:

```
import pandas as pd
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
from matplotlib import font_manager,rc,style
import seaborn as sns
style.use('ggplot')
plt.rc('font',family='AppleGothic')
plt.rc('axes', unicode_minus=False)
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
```

앱 순위를 크롤링 사이트: 모바일 인덱스
기간 : 2019년 1월 ~ 12월 / 2020년 1월 ~ 9월
기간별 어플 순위 / 구글 플레이 스토어(안드로이드) 데이터를 크롤링

In [57]:

```
appRank_free_playstore_19_df = pd.read_csv('/Users/jaeyoungshin/Desktop/빅콘/pickle파
appRank_free_playstore_20_df = pd.read_csv('/Users/jaeyoungshin/Desktop/빅콘/pickle파
```

In [58]:

```
fp19 = appRank_free_playstore_19_df.drop(['Unnamed: 0'],axis=1)
fp20 = appRank_free_playstore_20_df.drop(['Unnamed: 0'],axis=1)
```

In [60]:

```
fp19.to_pickle('appRank_free_playstore_19_df.pkl')
fp20.to_pickle('appRank_free_playstore_20_df.pkl')
```

In [27]:

```
fp19 = pd.read_pickle('appRank_free_playstore_19_df.pkl')
fp20 = pd.read_pickle('appRank_free_playstore_20_df.pkl')
```

19년의 건강/운동 항목에는 어떤 어플이 유행했을까?

- 8월 Mi Fit
- 10월 ANT+ Plugins Service
- 11월 ANT+ Plugins Service

20년의 건강/운동 항목에는 어떤 어플이 해당될까?

- 코로나맵 - 신종 코로나 바이러스 현황 (2월)
- 코로나맵 - 코로나알리미 (2월)
- 굿닥 – 마스크스캐너, 병원 약국 찾기 (3월)
- 전자출입명부(시설용) 보건복지부 (6월)
- 전자출입명부(시설용) 보건복지부 (8월)
- 건강상태 자가진단(교육부) (9월)
- 질병관리본부 예방접종도우미(9월)

19년도와는 달리 미용 목적의 건강/운동 항목이 아니라 코로나와 건강, 질병 관련 어플들이 랭킹에 올라와있음을 확인 가능
꾸준히 관심이 지속되고 있음도 확인 가능
자가진단, 예방접종도우미 어플 : 코로나가 장기화됨에 따라 현재는 단순 현황 뿐 아니라 개인의 건강을 직접 체크하고 예방하려는 양상이 보인다.

20년에 등장한 의료 항목에는 어떤 어플이 있을까?

- 코백-코로나 100m 근접시 푸시알림, 확진자 경로추적, 코로나19, 우한폐렴 정보서비스
- 코로나 알리미 - 우한 폐렴, 코로나 바이러스, 중국 폐렴, 바이러스
- 의료 어플은 현재까지 20년 월별 50위 랭킹 순위에 있는 건 이 두 개밖에 없고, 2월에만 등장

In [364]:

```
fp20
```

42	700만이 선택한 No.1 인테리어 필수앱	부동산/홈 인테리어	No.1 지도 앱 (지도 / 내비게이션 / 대중교통 / 로드뷰)	여행 및 지역정보	NH스마트뱅킹	금융	청약홈	금융	신한페이판	
43	바이오인증 공동앱	금융	타임커머스 티몬	쇼핑	굿닥 – 마스크스캐너, 병원 약국 찾기	건강/운동	KT 금융유심관리	금융	네이버 웹툰 - Naver Webtoon	
44	네이버 파파고 - AI 통번역	도구	(New) 우리은행 우리 WON뱅킹	금융	정부24(구 민원24)	생산성	Facebook	소셜	스타벅스	엔
45	NH콕뱅크	금융	아이디어스 (idus) - 핸드메이드/수공예 장터	쇼핑	wavve(웨이브) - 재미의 파도를 타다!	엔터테인먼트	카카오맵 - 대한민국 No.1 지도 앱 (지도 / 내비게이션 / 대중교통 / 로드뷰)	여행 및 지역정보	카카오내비	가

In [365]:

```
fp19
```

Out[365]:

	19_1	19_1_gen	19_2	19_2_gen	19_3	19_3_gen	19_4	19_4_gen	19_5	19_5_g
0	국세청 홈택스	도구	Magic Booster - 제일 좋아하는 청리도구	도구	Samsung Notes	생산성	배달의민족	식음료	Snapchat	ㅅ
1	TikTok 틱톡	동영상 플레이어/편집기	배달요기요	식음료	NH스마트뱅킹	금융	SRT - 수서고속철도(NEW)	여행 및 지역정보	i-ONE Bank - IBK 기업은행	ㄹ
2	Netflix(넷플릭스)	엔터테인먼트	커넥츠: 무료 공부질문 앱-500만 다운로드 (문제해설, 노하우, 과외, 인강)	교육	삼성 음성 녹음	도구	Super Magic Cleaner - 안드로이드 클리너 휴대폰 쿨러 안티바이러스	도구	국세청 홈택스	ㄱ

In [28]:

```
# 19,20년 전체 데이터프레임 월별 장르별 점수 데이터프레임
def score_df(df,month,yr):
    tmp = [i for i in list(df) if 'gen' in i]
    genre_list = list(set(np.array(df[tmp]).reshape(len(tmp)*50,)))
    genre_yr = []
    for i in tmp:
        genre_dic = {key: 0 for key in dict.fromkeys(genre_list).keys()}
        score = 50
        for j in df.dropna()[i]:
            genre_dic[j] = genre_dic[j]+score
            score -= 1
        genre_yr.append(genre_dic)

    col = []
    for i in range(month):
        col.append(f'{yr}_'+str(i+1))

    app_yr = pd.DataFrame(genre_yr)
    app_yr = app_yr.T
    app_yr.columns = col
    app_yr.drop([app_yr.index[0]],inplace=True)

    return app_yr


# 19년도 1~10월 평균을 기준으로 설정
def standard_df(df,month):
    if month == 10:
        standard_19 = df[['19_1','19_2','19_3','19_4','19_5','19_6','19_7','19_8','1
        standard_19 = standard_19.reset_index()
        standard_19 = standard_19.set_index('index')

        standard_19['average'] = 0.000
        for i in range(len(standard_19)):
            total = 0
            for j in range(0,10):
                total += standard_19.iloc[i][j]
            standard_19['average'][i] = total.mean()

        standard_19 = standard_19[['average']]
        standard_19 = standard_19.reset_index()
        genre = list(standard_19['index'])
        std_df = standard_19.set_index('index')
    return genre,std_df


# 증감율 계산 : (시점값 - 기준값) / 기준값
# 19년은 12월까지이므로 12입력하면 19년도 데이터프레임 반환
def rate_df(genre,std_df,df,month):
    if month == 12:
        app_yr = df[['19_11','19_12']]
        col = []
        for i in range(11,13):
            col.append(str(i)+'월')

        app_yr = app_yr.reset_index()

        for i in range(len(col)):
            app_yr[col[i]] = 0.000
```

```
    for i in range(len(app_yr)):
        base = app_yr['index'][i]
        if base in genre:
            idx = genre.index(base)
            base = std_df.iloc[idx]['average']
        else:
            base = 1.0
        for j in range(2,4):
            if base == 0:
                value = 0.0
            else:
                value = (float(app_yr.iloc[i][j]) - base) / base
            app_yr[col[j-2]][i] = value

    app_yr = app_yr[['index','11월','12월']]
    app_yr.columns = ['genre','11월','12월']

    app_fig = pd.DataFrame()
    genre_lst, val_tmp, mth_tmp = [], [], []
    for i in range(len(app_yr.columns)-1):
        genre_lst.append(list(app_yr['genre']))
        val_tmp.append(app_yr.iloc[:,i+1].tolist())
        for j in range(len(app_yr)):
            mth_tmp.append(app_yr.columns[i+1])

    genre_lst = [y for x in genre_lst for y in x]
    val_tmp = [y for x in val_tmp for y in x]

    app_fig['genre'] = genre_lst
    app_fig['month'] = mth_tmp
    app_fig['value'] = val_tmp
    return app_fig

if month == 9:
    app_yr = df
    col = []
    for i in range(1,10):
        col.append(str(i)+'월')

    app_yr = app_yr.reset_index()

    for i in range(len(col)):
        app_yr[col[i]] = 0.000

    for i in range(len(app_yr)):
        base = app_yr['index'][i]
        if base in genre:
            idx = genre.index(base)
            base = std_df.iloc[idx]['average']
        else:
            base = 1.0
        for j in range(1,10):
            if base == 0:
                value = 0.0
            else:
                value = ((float(app_yr.iloc[i][j]) - base) / base)*100
            app_yr[col[j-1]][i] = value

    app_yr = app_yr[['index','1월','2월','3월','4월','5월','6월','7월','8월','9월']]
    app_yr.columns = ['genre','1월','2월','3월','4월','5월','6월','7월','8월','9월']
```

```
        app_fig = pd.DataFrame()
        genre_lst, val_tmp, mth_tmp = [], [], []
        for i in range(len(app_yr.columns)-1):
            genre_lst.append(list(app_yr['genre']))
            val_tmp.append(app_yr.iloc[:,i+1].tolist())
            for j in range(len(app_yr)):
                mth_tmp.append(app_yr.columns[i+1])

        genre_lst = [y for x in genre_lst for y in x]
        val_tmp = [y for x in val_tmp for y in x]

        app_fig['genre'] = genre_lst
        app_fig['month'] = mth_tmp
        app_fig['value'] = val_tmp
        return app_fig

    #app19_graph = pd.concat([pay_playstore_19_graph,pay_playstore_19_fig], axis = (
```

In [29]:

```
score_19 = score_df(fp19,12,19)
genre,std_df = standard_df(score_df(fp19,12,19),10)
rate_19_grph = rate_df(genre,std_df,score_19,12)
```

```
<ipython-input-28-e7c0cd21913a>:38: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)

<ipython-input-28-e7c0cd21913a>:73: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
```

In [5]:

```
# 19년 1~10월을 기준지표로 했을 때 나타난 19년 11~12월 증감율
rate_19_grph
```

Out[5]:

	genre	month	value
0	기타	11월	-1.000000
1	도서/참고자료	11월	-0.876380
2	여행 및 지역정보	11월	-0.834555
3	소셜	11월	-0.894799
4	자동차	11월	-1.000000
5	예술/디자인	11월	-1.000000
6	식음료	11월	-0.883049
7	라이프스타일	11월	-1.000000
8	비즈니스	11월	-0.947368
9	음악/오디오	11월	-1.000000
10	도구	11월	-0.951675
11	날씨	11월	-1.000000
12	커뮤니케이션	11월	-0.933227
13	만화	11월	-1.000000
14	뉴스/잡지	11월	-1.000000
15	쇼핑	11월	-0.881356
16	엔터테인먼트	11월	-0.787429
17	교육	11월	-1.000000
18	건강/운동	11월	-1.000000
19	사진	11월	-0.885563
20	지도/내비게이션	11월	-0.875208
21	생산성	11월	-0.986450
22	동영상 플레이어/편집기	11월	-0.747899
23	부동산/홈 인테리어	11월	-0.905983
24	기타	12월	-1.062500
25	도서/참고자료	12월	-1.001935
26	여행 및 지역정보	12월	-1.000874
27	소셜	12월	-1.001058
28	자동차	12월	-1.090909
29	예술/디자인	12월	-1.142857
30	식음료	12월	-1.001160
31	라이프스타일	12월	-1.004237
32	비즈니스	12월	-1.024931

	genre	month	value
33	음악/오디오	12월	-1.005000
34	도구	12월	-1.000523
35	날씨	12월	-1.004785
36	커뮤니케이션	12월	-1.001484
37	만화	12월	-1.142857
38	뉴스/잡지	12월	-1.100000
39	쇼핑	12월	-1.000622
40	엔터테인먼트	12월	-1.000900
41	교육	12월	-1.015152
42	건강/운동	12월	-1.035714
43	사진	12월	-1.001559
44	지도/내비게이션	12월	-1.001456
45	생산성	12월	-1.002673
46	동영상 플레이어/편집기	12월	-1.002095
47	부동산/홈 인테리어	12월	-1.003872

In [30]:

```
score_20 = score_df(fp20,9,20)
rate_20_grph = rate_df(genre,std_df,score_20,9)
```

```
<ipython-input-28-e7c0cd21913a>:117: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
```

In [31]:

```
graph_ = pd.concat([rate_19_grph,rate_20_grph],axis=0)
graph_
```

Out[31]:

	genre	month	value
0	지도/내비게이션	11월	-0.875208
1	뉴스/잡지	11월	-1.000000
2	부동산/홈 인테리어	11월	-0.905983
3	비즈니스	11월	-0.947368
4	자동차	11월	-1.000000
...	...	...	...
220	여행 및 지역정보	9월	-96.649215
221	사진	9월	-100.000000
222	커뮤니케이션	9월	-93.481717
223	교육	9월	-100.000000

In [10]:

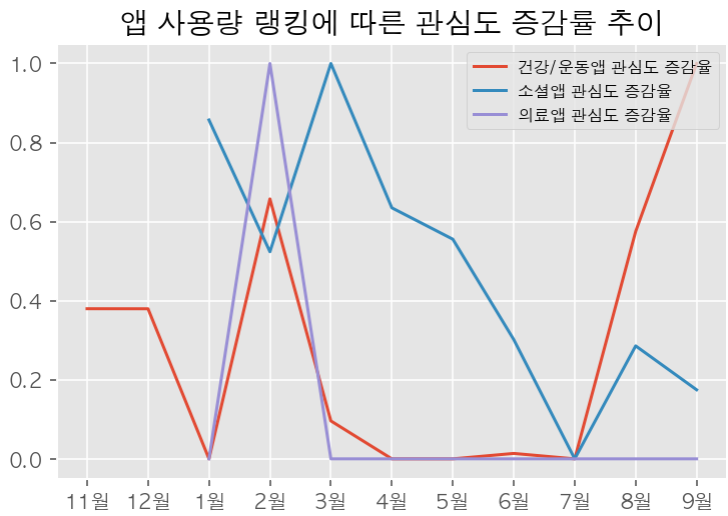
```
def float_genre(genre):
    tmp = graph_[(graph_.genre ==f'{genre}')]
    minMaxScaler = MinMaxScaler()
    scaled_val = minMaxScaler.fit_transform(np.array(tmp['value']).reshape(-1,1)).tc
    plt.plot(tmp.month,scaled_val, label=f'{genre}앱 관심도 증감율')
    plt.title('앱 사용량 랭킹에 따른 관심도 증감률 추이')
    plt.legend(fontsize=8)
    plt.rc('font',family='AppleGothic')
    plt.rc('axes', unicode_minus=False)
```

In [9]:

```
import numpy as np
from sklearn.preprocessing import MinMaxScaler
minMaxScaler = MinMaxScaler()
health = minMaxScaler.fit_transform(np.array(graph_[(graph_.genre =='건강/운동')]['val
health = np.ravel(health).tolist()
social = minMaxScaler.fit_transform(np.array(graph_[(graph_.genre =='소셜')]['value']
social = np.ravel(social).tolist()
```

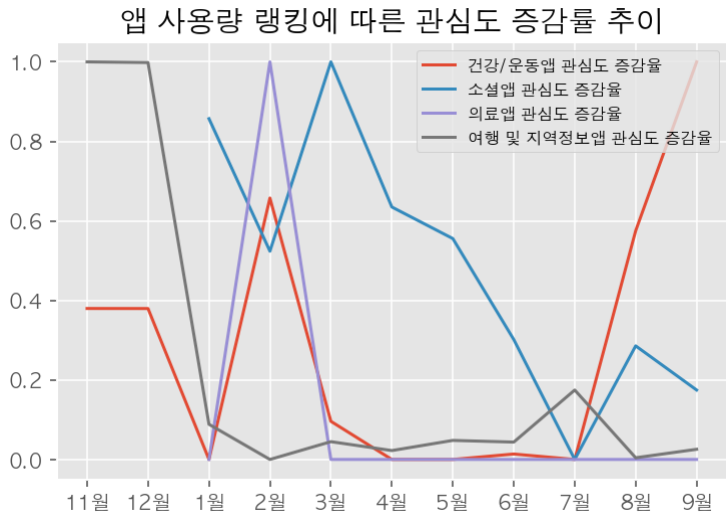
In [440]:

```
float_genre('건강/운동')
float_genre('소셜')
float_genre('의료')
```



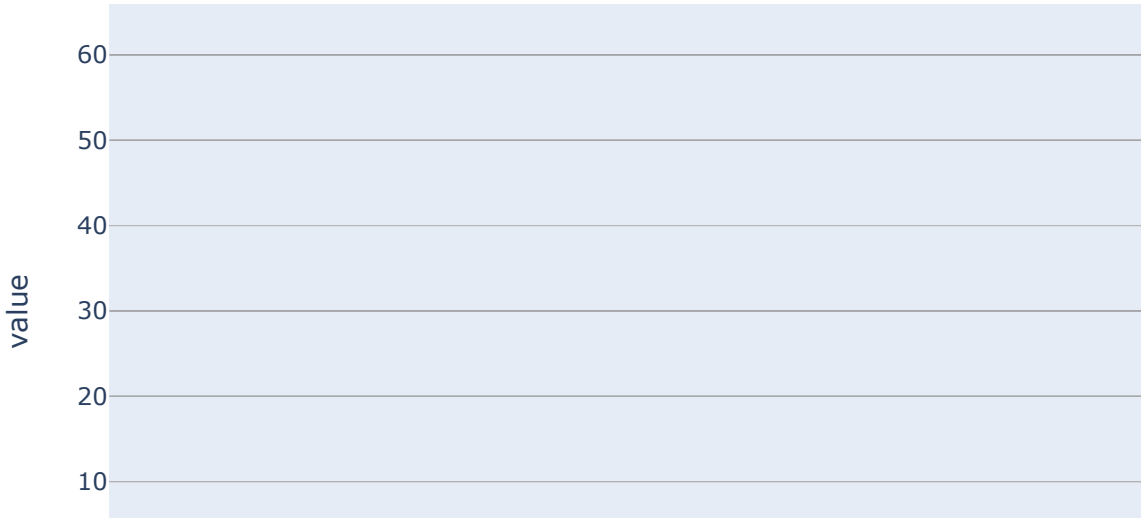
In [441]:

```
float_genre('건강/운동')
float_genre('소셜')
float_genre('의료')
float_genre('여행 및 지역정보')
```



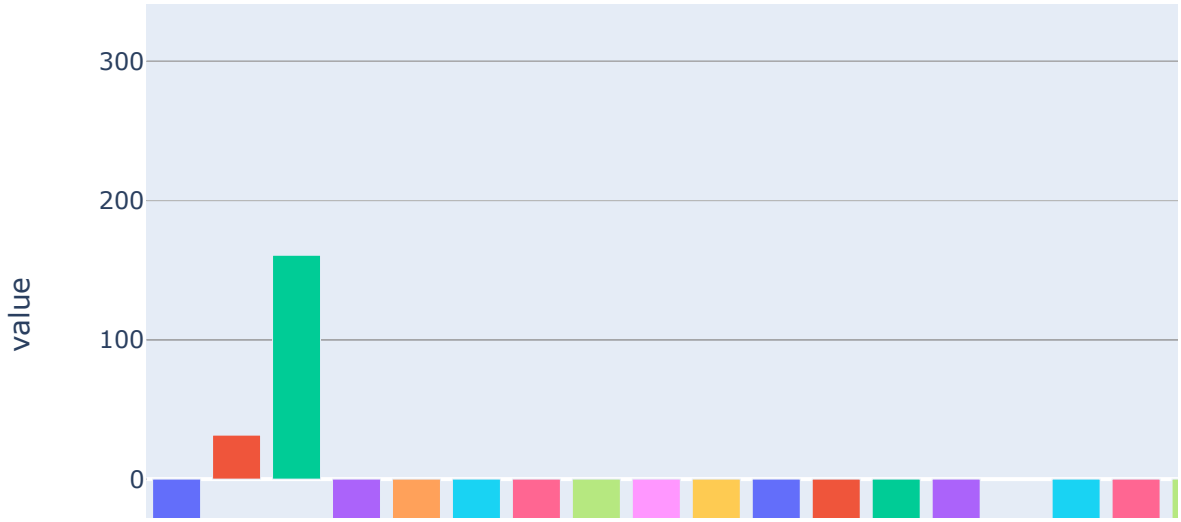
In [444]:

```
# 소셜은 사용량이 매우 크므로 다른 어플의 추이를 더 보기 위해 제거해보자
graph_wo_social = graph_[(graph_.genre !='소셜')].reset_index(drop=True)
fig = px.bar(graph_wo_social, x='genre',y='value',color='genre',animation_frame='mon
fig.show()
```



In [443]:

```
# max값이 크게 나타났던 소셜과 스포츠,의료 장르를 제외하고 사용랭킹에 따른 앱 관심도 증감률을 봐보자
graph_wo_ = graph_[(graph_.genre !='소셜') & (graph_.genre !='스포츠')& (graph_.genre
fig = px.bar(graph_wo_, x='genre',y='value',color='genre',animation_frame='month',ar
fig.show()
```



In [90]:

```
std_df = standard_df(score_df(fp19,12),10)
std_df
```

```
<ipython-input-87-c8f81bc25f1e>:37: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  standard_19['average'][i] = total/10
```

Out[90]:

index	average
사진	56.8
비즈니스	3.8
건강/운동	2.8
부동산/홈 인테리어	23.4
교육	6.6
날씨	20.9
쇼핑	141.6
만화	0.7
동영상 플레이어/편집기	35.7
지도/내비게이션	60.1
도구	182.1
도서/참고자료	45.3
커뮤니케이션	62.9
금융	203.7
기타	1.6
음악/오디오	20.0
뉴스/잡지	1.0
자동차	1.1
예술/디자인	0.7
생산성	36.9
라이프스타일	23.6
식음료	76.1
여행 및 지역정보	95.5
엔터테인먼트	87.5