

모듈 импорт

In [1]:

```
import numpy as np
import pandas as pd
import plotly.offline as pyo
import plotly.graph_objs as go
import matplotlib.pyplot as plt
import plotly.express as px
import matplotlib.pyplot as plt
from matplotlib import cm
import requests
import json
import datetime as dt
from pandas import DataFrame
from matplotlib import pyplot
from IPython.display import SVG # jupyter 상에서 SVG 이미지를 표시하기 위한 패키지
from bs4 import BeautifulSoup
plt.rc('font', family='Malgun Gothic')
import pickle
from tqdm.notebook import tqdm
import plotly.express as px
import os
```

데이터 불러들이기

In [2]:

```
df = pd.read_csv('지역별_2010부터_감염병발생빈도.csv')
data = pd.read_csv('지역별_2018부터_10만명당감염병발생비율.csv')
age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)
with open('season.pickle', 'rb') as f:
    season = pickle.load(f)
```

In [3]:

```
# 파일 저장하는거
# conda install -c plotly plotly-orca
# 이거 하면 이미지로 저장돼
```

빈도수 기반 top10 만들기

In [4]:

```
#Series sort_values : [: :-1] -> 내림차순 -> 10개 -> 의 인덱스
k = df.groupby('년').sum().sum()
pd.DataFrame(k.sort_values()[: :-1][:10]).index

top10 = list(pd.DataFrame(k.sort_values()[: :-1][:10]).index)
```

나이 연령대 만들기

In [5]:

```
ages_range = list(age[0][1:])
```

시각화를 위한 함수

In [6]:

```

index_dict = {
    '강원': '강원도',
    '경기': '경기도',
    '경남': '경상남도',
    '경북': '경상북도',
    '광주': '광주광역시',
    '대구': '대구광역시',
    '대전': '대전광역시',
    '부산': '부산광역시',
    '서울': '서울특별시',
    '세종': '세종특별자치시',
    '울산': '울산광역시',
    '인천': '인천광역시',
    '전남': '전라남도',
    '전북': '전라북도',
    '제주': '제주특별자치도',
    '충남': '충청남도',
    '충북': '충청북도'}

#####
# 총 감염병수를 연령별로 나눠 파이차트 그리기#
#####
def plot_pie_chart_by_age(disease):
    age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)
    col = ['나이', '에볼라바이러스병', '마버그열', '라싸열', '크리미안콩고출혈열', '남아메리카출혈열',
            '중증급성호흡기증후군(SARS)', '중증호흡기증후군(MERS)', '동물인플루엔자 인체감염증', '신종인플루엔자',
            '세균성이질', '장출혈성대장균감염증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전)', '유행성출혈열',
            '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페뎀내성장루브러균 감염증',
            'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '쯔쯔가무시증', '립티스균 감염증',
            '황열', '뎡기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열', '중증급성호흡기증후군(MERS)']
    age.columns = col
    group_names = age['나이'][1:]
    #age[disease] = age[disease].apply(lambda x : 0 if x == '-' else float(x))
    group_sizes = age[disease][1:]
    if group_sizes[1] == '-':
        print('나이별로 집계 안 됨')

    else:
        fig = px.pie(age, values=disease, names=age['나이'], title='나이별 {} 발생비율'.format(disease),
                    color = age['나이'],
                    )

        fig.show()

    # Pie chart
    # plt.figure(figsize = (7,7))
    # plt.pie(group_sizes,
    #         labels=group_names,
    #         # colors= cs,
    #         autopct='%1.2f%%', # second decimal place
    #         shadow=True,
    #         textprops={'fontsize': 10}) # text font size
    # plt.axis('equal') # equal length of X and Y axis
    # plt.title('{} 연령별 총 분포'.format(disease), fontsize=20)
    # plt.show()

def plot_pie_chart_by_age_color(disease):
    age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)

```

```

age = age[1:]
column = ['나이', '에볼라바이러스병', '마버그열', '라싸열', '크리미안콩고출혈열', '남아메리카출혈열',
'중증급성호흡기증후군(SARS)', '중증호흡기증후군(MERS)', '동물인플루엔자 인체감염증', '신종인플루엔자',
'세균성이질', '장출혈성대장균감염증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전)', '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페뎀내성장루브러균 감염증', 'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '쯔쯔가무시증', '렙토스피라 증식열', '황열', '뎅기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열', '중증급성호흡기증후군(MERS)']

age.columns = column
group_names = age['나이']
#age[disease] = age[disease].apply(lambda x : 0 if x == '-' else float(x))
group_sizes = age[disease]

# 색깔 만들기
color_age = {}
for i in range(len(ages_range)):
    color_age[ages_range[i]] = px.colors.sequential.RdBu[i]

if not os.path.exists("images_age"):
    os.mkdir("images_age")

if group_sizes[1] == '-':
    print('나이별로 집계 안 됨 : {}'.format(disease))

else:
    # 제목 넣고싶으면 이거 fig에 추가하기 : title='나이별 {} 발생비율'.format(disease),
    fig = px.pie(age, values=disease, names=age['나이'],
        color = age['나이'],
        color_discrete_map=color_age

    )
    fig.update_traces(textposition='inside', textinfo='percent+label', textfont_size=20)
    fig.write_image("images_age/{}.png".format(disease))

    fig.show()

#####
# 총 감염병수를 계절별로 나눠 파이차트 그리기#
#####

def plot_pie_chart_by_season_color(disease):
    with open('season.pickle', 'rb') as f:
        season = pickle.load(f)
    group_names = season['계절']
    group_sizes = season[disease]

    if not os.path.exists("images_season"):
        os.mkdir("images_season")

    plt.figure(figsize = (7,7))
    # 제목 넣고싶으면 title='계절별 {} 발생비율'.format(disease), 아래에 넣자
    fig = px.pie(season, values=disease, names=season['계절'], color=season['계절'],
        color_discrete_map={'봄': 'lightcyan',
        '여름': 'royalblue',
        '가을': 'cyan',

```

```
'겨울': 'darkblue'}, custom_data= ['계절']
```

```
)
fig.update_traces(textposition='inside', textinfo='percent+label', textfont_size=20)
#category_orders={"계절": ["봄", "여름", "가을", "겨울"]})

fig.write_image("images_season/{disease}.png".format(disease))

fig.show()

def plot_pie_chart_by_season(disease):
    with open('season.pickle', 'rb') as f:
        season = pickle.load(f)
        group_names = season['계절']
        group_sizes = season[disease]

    plt.figure(figsize = (7,7))
    fig = px.pie(season, values=disease, names=season['계절'], title='계절별 {} 발생비율'.format(disease))
    fig.show()

    ## plt.pie(group_sizes,
    ##         labels=group_names,
    ##         # colors= cs,
    ##         autopct='%1.2f%%', # second decimal place
    ##         shadow=True,
    ##         textprops={'fontsize': 10}) # text font size
    ## plt.axis('equal') # equal length of X and Y axis
    ## plt.title('{} 계절별 총 분포'.format(disease), fontsize=20)
    ## plt.show()

#####
# 총 감염병 확진자를 월별로 그냥 그리기#####
#####

def montly_trend(disease):
    df = pd.read_csv('지역별_2010부터_감염병발생빈도.csv')
    df['시'] = df['구분'].apply(lambda x : x[:2])
    df[disease] = df[disease].apply(lambda x : 0 if x == '-' else float(x))
    date = df[[disease, '날짜', '구분', '시', '년']].groupby('날짜').sum().reset_index()
    plt.figure(figsize = (30,10))
    plt.plot(date['날짜'][:-1], date[disease][:-1])
    plt.xticks(rotation=45)
    plt.title('{} 월별 증감수'.format(disease), fontdict = {'fontsize': 40})
    plt.show()

#####
# 10만명당 발생률 비율로 , 가장 최근달인 8월을 기준으로 발생률 나눔#####
#####

def preprocess_plot(disease):
    data = pd.read_csv('지역별_2018부터_10만명당감염병발생비율.csv')
    data[disease] = data[disease].apply(lambda x : 0 if x == '-' else float(x))
    data['지역'] = data['구분'].apply(lambda x : x[:2])

    #last = data[data['날짜'] == data['날짜'].values.max()].groupby('지역').mean().reset_index()
    last = data[data['날짜'] == data['날짜'].values.max()].groupby('지역').mean().reset_index()
    k = last[['지역', disease]]
    결과df = k.set_index('지역')
    최종df = 결과df.rename(index=index_dict)
    return 최종df
```

```
#####
# 0.2, 0.4, 0.6, 0.8, 기준으로 나눴읍니다 (진해질수록 많아) #####
#####
def make_quantile(data, disease):
    Q1 = data[disease].quantile(.20) # 7월달 기준 20% 미만은 안전(대체)
    Q2 = data[disease].quantile(.40)
    Q3 = data[disease].quantile(.90)
    Q4 = data[disease].quantile(.975)

    return Q1, Q2, Q3, Q4

def ratio_disease_plot(data, disease, Q1, Q2, Q3, Q4):

    with open('Administrative_divisions_map_of_South_Korea.svg', 'r', encoding='utf-8') as f:
        map_svg = f.read()

    soup = BeautifulSoup(map_svg, "lxml")
    glist = soup.select("svg > g[id], svg > path[id]")

    # 빨간색 or 파란색?

    # colors = ['#eaeceb', '#bfbfbf', '#b2e1ff', '#15a8de', '#118ac9']
    colors = px.colors.sequential.RdBu[:5][::-1]

    for item in glist:
        # svg 파일에서 추출한 id값이 데이터프레임의 index에 속하지 않았다면 다음으로 넘어감
        if item['id'] not in data.index:
            continue

        # item['id'] -> 지도에서 뽑아낸 id
        # 최종df에서 인덱스 이름이 지도에서 뽑아낸 id와 일치하는 행의 의료기관수를 추출
        count = data.loc[ item['id'], disease ]

        # 누적확진자수에 따라 단계값 설정 (단계는 색상값의 수에 따름)
        if Q4 == 0:
            color_index = 0
        else:
            if count <= Q1: color_index = 0
            elif count < Q2: color_index = 1
            elif count < Q3: color_index = 2
            elif count < Q4: color_index = 3
            else: color_index = 4

        #print("%s: 확진자수: %d, 색상인덱스: %d" % (item['id'], count, color_index))

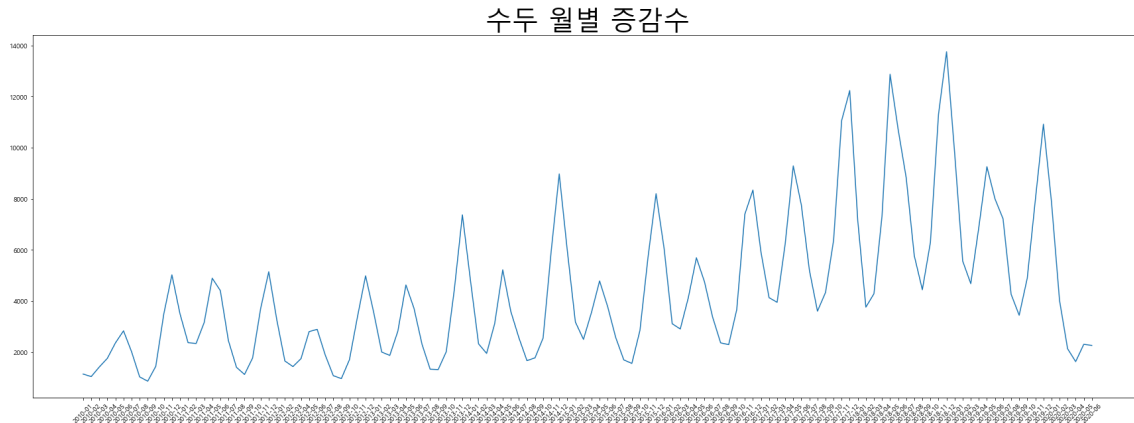
        # item에 fill 속성이 있는 경우 -> 세종특별자치시
        if 'fill' in item.attrs:
            item['fill'] = colors[color_index]
        # 그렇지 않은 경우 -> 나머지 지역
        else:
            # 현재 요소(item)의 하위 항목들에게 색상을 적용해야 한다.
            for p in item.select('g, path'):
                if 'fill' in p.attrs:
                    p['fill'] = colors[color_index]

    지역별확진자svg = soup.prettify()
    return SVG(지역별확진자svg)
```

```
def oneshot(disease):
    montly_trend(disease)
    plot_pie_chart_by_age_color(disease)
    plot_pie_chart_by_season_color(disease)
    최종df = preprocess_plot(disease)
    print(최종df.loc[최종df[disease].sort_values()[::-1][:5].index])
    Q1, Q2, Q3, Q4 = make_quantile(최종df, disease)
    return ratio_disease_plot(최종df, disease, Q1, Q2, Q3, Q4)
```

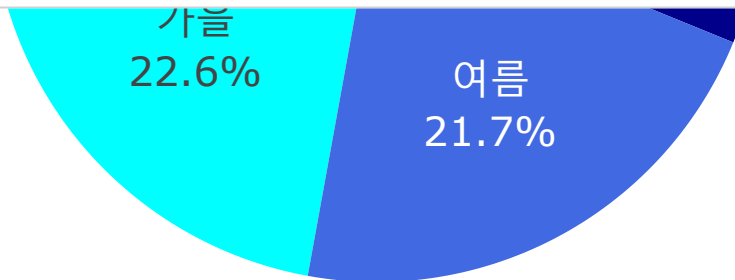
In [7]:

```
oneshot('수두')
```



In [8]:

```
oneshot('수두')
```

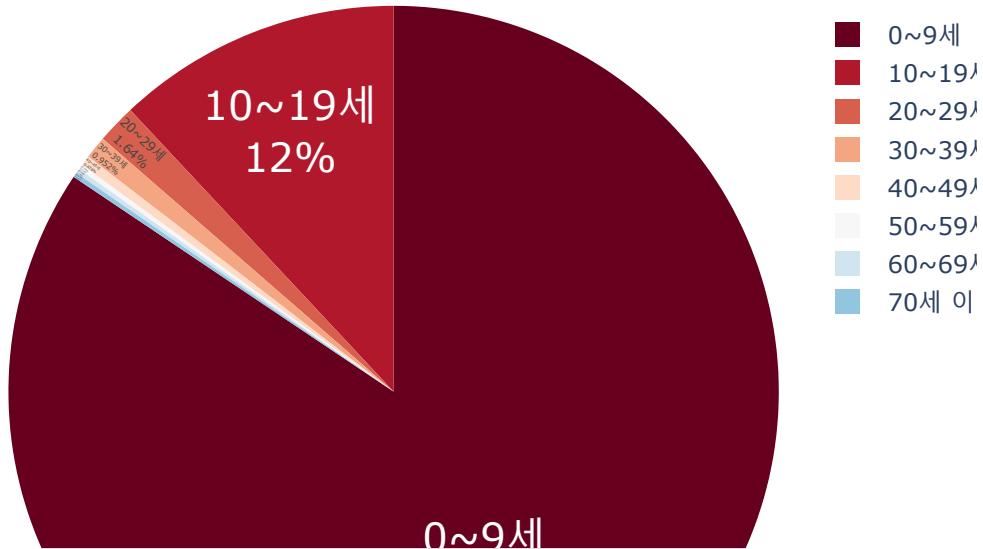


수두

지역	수두
광주광역시	5.404000
제주특별자치도	5.155000
전라남도	4.333636
울산광역시	4.328000
대전광역시	4.327500

In [9]:

```
for i in top10:
    plot_pie_chart_by_age_color(i)
    plot_pie_chart_by_season_color(i)
```



In [10]:

```
def plot_pie_chart_by_season(disease):
    with open('season.pickle', 'rb') as f:
        season = pickle.load(f)
        group_names = season['계절']
        group_sizes = season[disease]

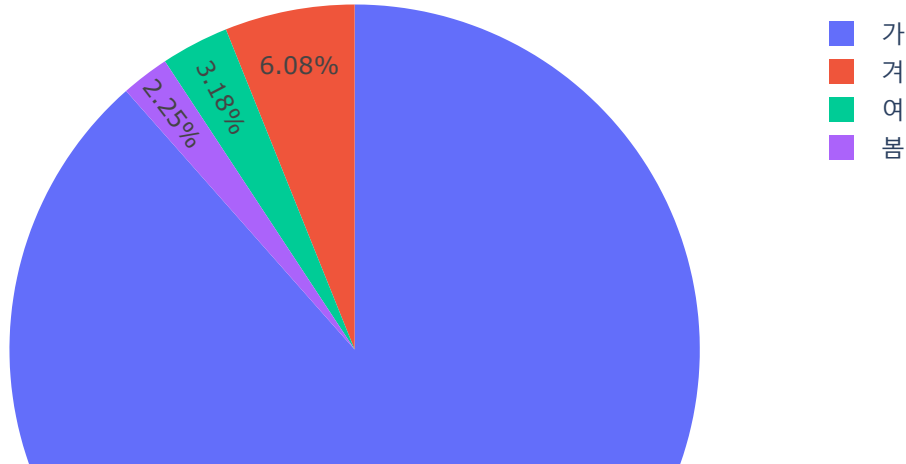
    plt.figure(figsize = (7,7))
    fig = px.pie(season, values=disease, names=season['계절'], title='계절별 {} 발생비율'.format(disease))
    fig.show()

# plt.pie(group_sizes,
#         labels=group_names,
#         colors= cs,
#         autopct='%1.2f%%', # second decimal place
#         shadow=True,
#         textprops={'fontsize': 10}) # text font size
# plt.axis('equal') # equal length of X and Y axis
# plt.title('{} 계절별 총 분포'.format(disease), fontsize=20)
# plt.show()
```


In [11]:

```
plot_pie_chart_by_season('쯔쯔가무시증')
```

계절별 쯔쯔가무시증 발생비율



In [12]:

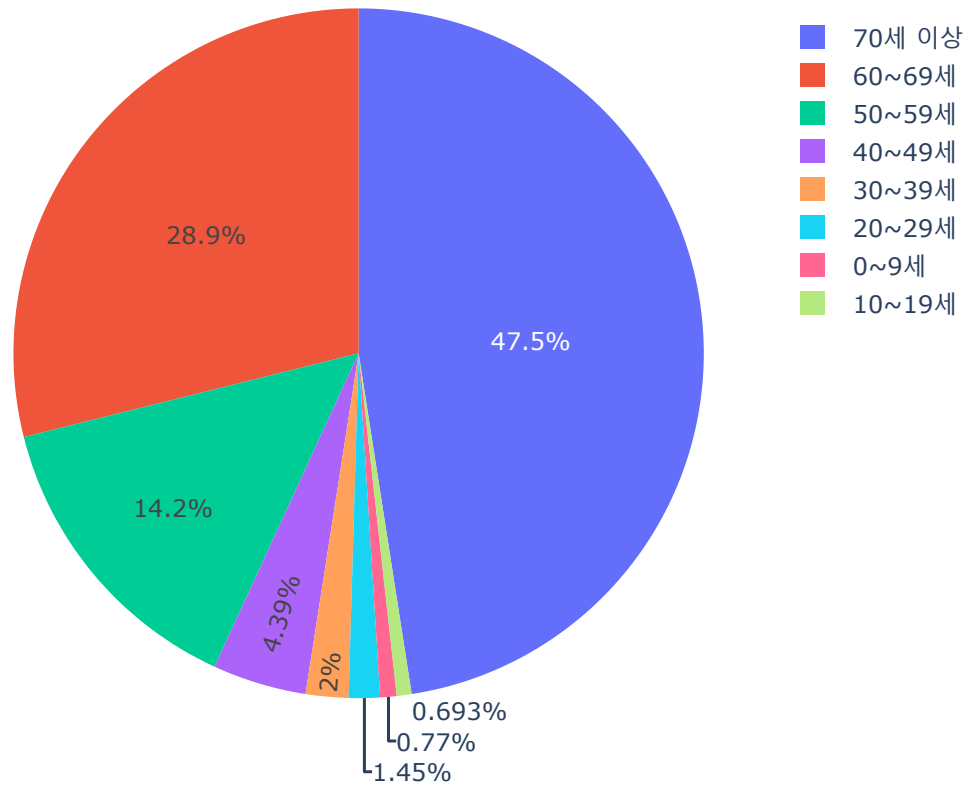
```
def plot_pie_chart_by_age(disease):
    age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)
    colum = ['나이', '에볼라바이러스병', '마버그열', '라싸열', '크리미안콩고출혈열', '남아메리카출혈열',
              '중증급성호흡기증후군(SARS)', '중증호흡기증후군(MERS)', '동물인플루엔자 인체감염증', '신종인플루엔자',
              '세균성이질', '장출혈성대장균감염증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전)',
              '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페넴내성균 감염증',
              'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '쯔쯔가무시증', '립열',
              '황열', '뎡기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열', '중증급성호흡기증후군(MERS)']
    age.columns = colum
    group_names = age['나이'][1:]
    #age[disease] = age[disease].apply(lambda x : 0 if x == '-' else float(x))
    group_sizes = age[disease][1:]
    if group_sizes[1] == '-':
        print('나이별로 집계 안 됨')
    else:
        fig = px.pie(age.iloc[1:][:], values= disease, names=group_names, title='나이별 {} 발생비율')
        fig.show()

    # Pie chart
    # plt.figure(figsize = (7,7))
    # plt.pie(group_sizes,
    #         labels=group_names,
    #         colors= cs,
    #         autopct='%1.2f%%', # second decimal place
    #         shadow=True,
    #         textprops={'fontsize': 10}) # text font size
    # plt.axis('equal') # equal length of X and Y axis
    # plt.title('{} 연령별 총 분포'.format(disease), fontsize=20)
    # plt.show()
```

In [13]:

```
plot_pie_chart_by_age('프프가무시증')
```

나이별 프프가무시증 발생비율



In [14]:

```

import plotly.graph_objs as go
def plot_pie_chart_by_season_color(disease):
    with open('season.pickle', 'rb') as f:
        season = pickle.load(f)
        group_names = season['계절']
        group_sizes = season[disease]

    if not os.path.exists("images_season"):
        os.mkdir("images_season")

    plt.figure(figsize = (7,7))
    # 제목 넣고싶으면 title='계절별 {} 발생비율'.format(disease), 아래에 넣자
    fig = px.pie(season, values=disease, names=season['계절'], color=season['계절'],
                 color_discrete_map={'봄': 'lightcyan',
                                     '여름': 'royalblue',
                                     '가을': 'cyan',
                                     '겨울': 'darkblue'}, custom_data= ['계절'],

    )

    fig.update_traces(textposition='inside', textinfo='percent+label', textfont_size=20)
    #category_orders={"계절": ["봄", "여름", "가을", "겨울"]})

    fig.write_image("images_season/{disease}.png".format(disease=disease))

    fig.show()

```

In [15]:

px.pie

Out [15]:

```

<function plotly.express._chart_types.pie(data_frame=None, names=None, values=None,
color=None, color_discrete_sequence=None, color_discrete_map={}, hover_name=None, ho
ver_data=None, custom_data=None, labels={}, title=None, template=None, width=None, h
eight=None, opacity=None, hole=None)>

```

빈도수 기반 top10

In [16]:

```
k.sort_values()[::-1][:10]
```

Out[16]:

수두	552335	
유행성이하선염	161294	
쯔쯔가무시증	80117	
성홍열	77978	
카바페넴내성장내세균속균종(CRE) 감염증	42550	
A형간염	41987	
C형간염	34107	
말라리아	7487	
신증후군출혈열	4492	
B형간염	2931	

dtype: int64

In [17]:

```
#Series sort_values : [::-1] -> 내림차순 -> 10개 -> 의 인덱스
k = df.groupby('년').sum().sum()
pd.DataFrame(k.sort_values()[::-1][:10]).index

top10 = list(pd.DataFrame(k.sort_values()[::-1][:10]).index)
```

In [18]:

```
top10
```

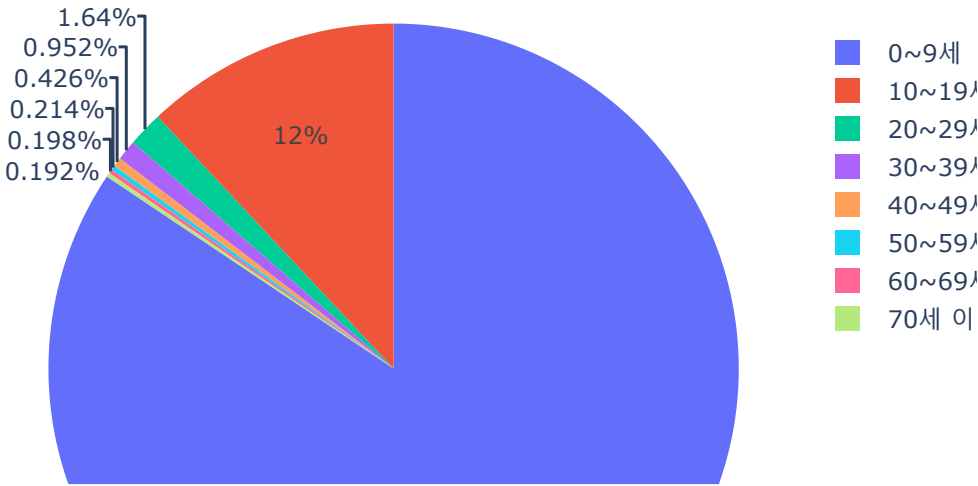
Out[18]:

```
['수두',
 '유행성이하선염',
 '쯔쯔가무시증',
 '성홍열',
 '카바페넴내성장내세균속균종(CRE) 감염증',
 'A형간염',
 'C형간염',
 '말라리아',
 '신증후군출혈열',
 'B형간염']
```

In [19]:

```
# 10개의 그래프를 일률적으로 그려보아야하나
for disease in top10:
    plot_pie_chart_by_age(disease)
```

나이별 수두 발생비율



In [21]:

```
plot_pie_chart_by_age_color('프프가무시증')
```

