Assignment# 2

Step# 1

jupyter Assignment__2 Last Checkpoint: 22 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [ ]:

In [9]:
```
import numpy as np
import matplotlib.pyplot as plt
import geoplot as gplt
import geopandas as gpd
import geoplot.crs as gcrs
import pandas as pd
import pycountry
import mapclassify as mc
import plotly.express as px
import seaborn as sns
from io import BytesIO
from zipfile import ZipFile
from nibabel import FileHolder
from nibabel import AnalyzeImage
```

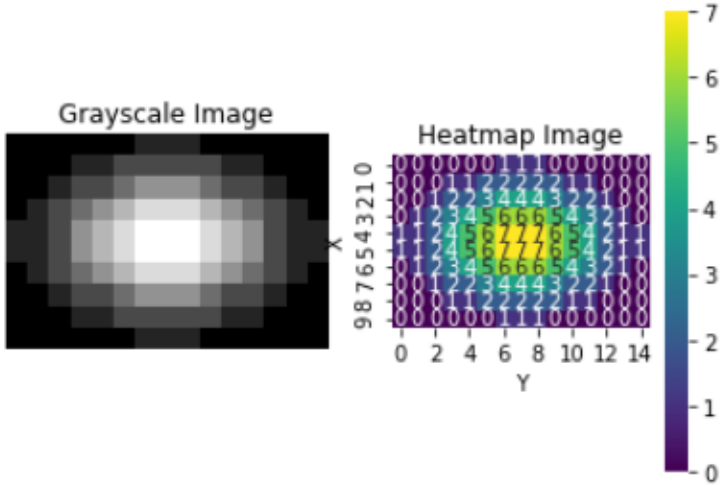
In [13]:
```
rows = 10
columns= 15
x , y = np.meshgrid(np.linspace(-1,1, columns), np.linspace(-1, 1, rows))
d = np.sqrt(x*x+y*y)
sigma = 0.5
myGaussianDisc = (8*np.exp(-((d)**2 / (2.0*sigma**2)))).astype ('uint8')
```

```
x = 3
y = 5
display(myGaussianDisc[x,y])

f,axes = plt.subplots(1,2)
(aGrayscale, aHeatmap) = axes.flatten()
aGrayscale.imshow(myGaussianDisc, cmap="gray")
aGrayscale.set_axis_off()
aGrayscale.set_title("Grayscale Image")


sns.heatmap(myGaussianDisc,cmap="viridis",square=True,annot=True,ax=aHeatmap)
aHeatmap.set_ylim(0,10)
aHeatmap.invert_yaxis()
aHeatmap.set_ylabel('X')
aHeatmap.set_xlabel('Y')
aHeatmap.set_title("Heatmap Image")
plt.show()
```

jupyter Assignment__2 Last Checkpoint: 25 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code



Step# 2

jupyter Assignment__2 Last Checkpoint: 26 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [14]:
```
rows = 10
columns = 15
x, y = np.meshgrid(np.linspace(-1,1,columns), np.linspace(-1,1,rows))
d = np.sqrt(x*x+y*y)
sigma = 0.5
disc = (8*np.exp(-((d)**2 / (2.0*sigma**2)))).astype('uint')
myRGBColorArray = np.stack([disc,np.roll(disc,2,axis=0),np.roll(disc,2,axis=1)],axis=2)
print("Red:")
display(myRGBColorArray[:,:,1])
```

```
Red:

array([[0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0]], dtype=uint32)
```

Step# 3

In [15]:
```
rows = 10
columns = 15
x, y = np.meshgrid(np.linspace(-1,1,columns), np.linspace(-1,1,rows))
d = np.sqrt(x*x+y*y)
sigma = 0.5
disc = (8*np.exp(-((d)**2 / (2.0*sigma**2)))).astype('uint')
myRGBColorArray = np.stack([disc,np.roll(disc,2,axis=0),np.roll(disc,2,axis=1)],axis=2)
print("Red channel:")
display(myRGBColorArray[:,:,0])
print("Green Channel:")
display(myRGBColorArray[:,:,1])
print("Blue channle:")
display(myRGBColorArray[:,:,2])
print("Composite channel:")
display(myRGBColorArray)
```

```
display(myRGBColorArray)
```

```
Red channel:

array([[0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0]], dtype=uint32)

Green Channel:

array([[0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0]], dtype=uint32)

Blue channle:

array([[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0],
       [0, 0, 0, 0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1],
       [1, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2],
       [1, 1, 1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2],
       [1, 1, 1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2],
       [1, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2],
       [0, 0, 0, 0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1],
       [0, 0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0]], dtype=uint32)

Composite channel:

array([[[0, 0, 0],
        [0, 0, 0],
```

Step# 4

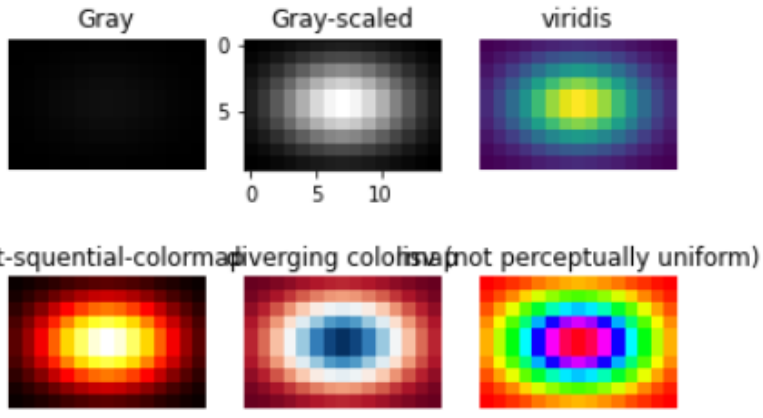
jupyter Assignment__2 Last Checkpoint: 30 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [16]:
```
rows = 10
columns = 15
x, y = np.meshgrid(np.linspace(-1,1,columns), np.linspace(-1,1,rows))
d = np.sqrt(x*x+y*y)
sigma = 0.5
myl2BitArray = ((2**12-1)*np.exp(-( (d)**2 / (2.0 * sigma**2 )))).astype('uint16')
f, axes = plt.subplots(2,3)
(aG, aS, aV, aH, aRB, aHSV) = axes.flatten()

aG.imshow(myl2BitArray, cmap="gray", vmin=0, vmax=(2**16)-1)
aG.set_axis_off()
aG.set_title("Gray")
aS. imshow(myl2BitArray, cmap="gray")
aG.set_axis_off()
aS.set_title("Gray-scaled")

aV.imshow(myl2BitArray, cmap="viridis")
aV.set_axis_off()
aV.set_title("viridis")

aH.imshow(myl2BitArray, cmap="hot")
aH.set_axis_off()
aH.set_title("Hot-squential-colormap")

aRB.imshow(myl2BitArray, cmap="RdBu")
aRB.set_axis_off()
aRB.set_title("diverging colormap")
```

```
aHSV.imshow(myl2BitArray, cmap="hsv")
aHSV.set_axis_off()
aHSV.set_title("lisv (not perceptually uniform)")
plt.show()
```



Step# 5

jupyter Assignment__2 Last Checkpoint: 31 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code

In [17]:
```
zip_archive = ZipFile('MRI-Dataset.zip')
```

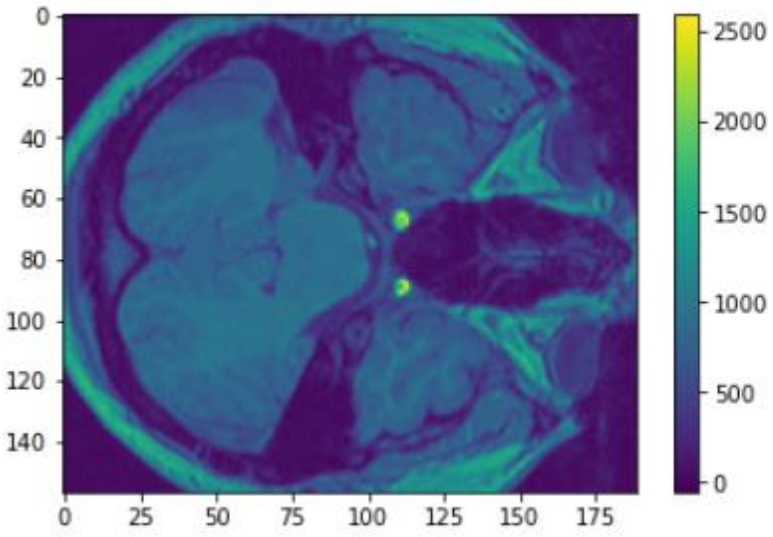
In [18]:
```
file_img = zip_archive.read('attention/structural/nsM00587_0002.img')
```

In [19]:
```
header = BytesIO(zip_archive.read('attention/structural/nsM00587_0002.hdr'))
image = BytesIO(zip_archive.read('attention/structural/nsM00587_0002.img'))
img = AnalyzeImage.from_file_map({'header': FileHolder(fileobj=header), 'image': FileHolder(fileobj=image) })
arr = img.get_fdata()
print(arr.shape)
```

```
(157, 189, 68)
```

In [20]:
```
plt.imshow(arr[:,:,5])
plt.colorbar()
plt.plot()
```

Out[20]: []



Step# 6

In [65]:
```
%matplotlib inline
usa = gpd.read_file("C:/Users/mubas/Downloads/us_state/us_state.shp")
usa.head()
```

Out[65]:

	STATEFP	STATENS	AFFGEOID	GEOID	STUSPS	NAME	LSAD	ALAND	AWATER	geometry
0	24	01714934	0400000US24	24	MD	Maryland	00	25151100280	6979966958	MULTIPOLYGON (((-76.04621 38.02553, -76.00734 ...
1	19	01779785	0400000US19	19	IA	Iowa	00	144661267977	1084180812	POLYGON ((-96.62187 42.77925, -96.57794 42.827...
2	10	01779781	0400000US10	10	DE	Delaware	00	5045925646	1399985648	POLYGON ((-75.77379 39.72220, -75.75323 39.757...
3	39	01085497	0400000US39	39	OH	Ohio	00	105828882568	10268850702	MULTIPOLYGON (((-82.86334 41.69369, -82.82572 ...
4	42	01779798	0400000US42	42	PA	Pennsylvania	00	115884442321	3394589990	POLYGON ((-80.51989 40.90666, -80.51964 40.987...

Step# 7

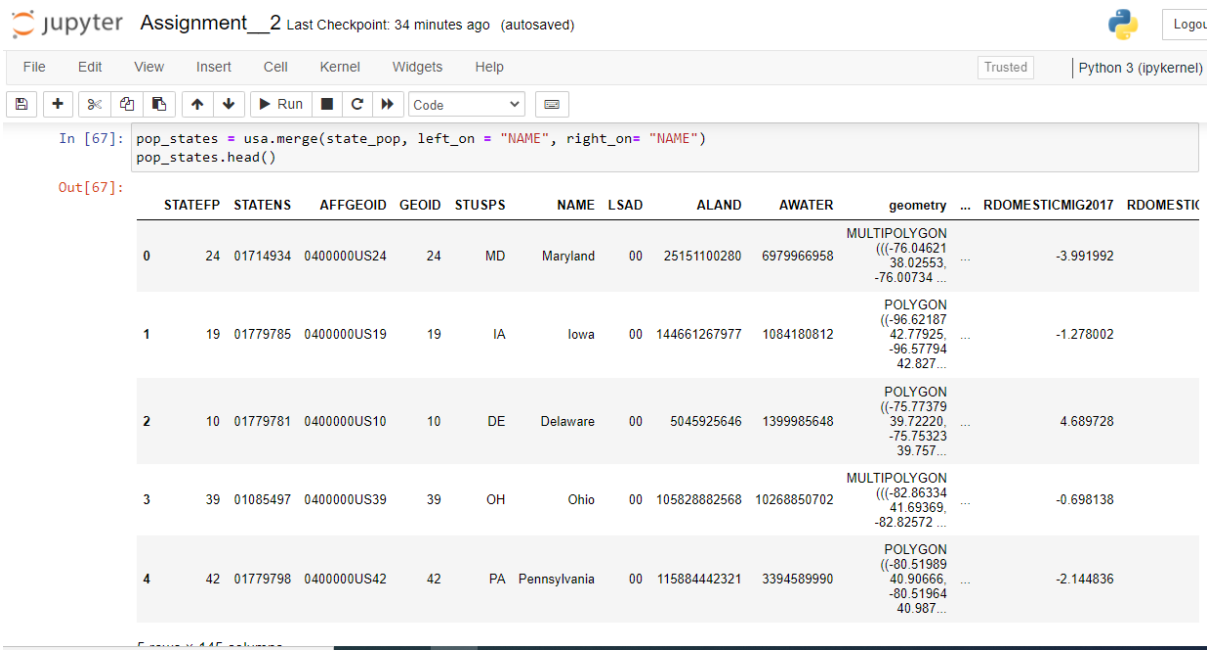
In [66]:
```
state_pop = pd.read_csv("C:/Users/mubas/Downloads/us_state_est_population.csv",encoding = "utf-8")
state_pop.head()
```

Out[66]:

	SUMLEV	REGION	DIVISION	STATE	NAME	CENSUS2010POP	ESTIMATESBASE2010	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	...	RD
0	10	0	0	0	United States	308745538	308758105	309326085	311580009	313874218	...	
1	20	1	0	0	Northeast Region	55317240	55318430	55380645	55600532	55776729	...	
2	20	2	0	0	Midwest Region	66927001	66929743	66974749	67152631	67336937	...	
3	20	3	0	0	South Region	114555744	114563045	114867066	116039399	117271075	...	
4	20	4	0	0	West Region	71945553	71946887	72103625	72787447	73489477	...	

5 rows × 136 columns

Step# 8



In [67]:
```
pop_states = usa.merge(state_pop, left_on = "NAME", right_on= "NAME")
pop_states.head()
```

Out[67]:

	STATEFP	STATENS	AFFGEOID	GEOID	STUSPS	NAME	LSAD	ALAND	AWATER	geometry	...	RDOMESTICMIG2017	RDOMESTIC
0	24	01714934	0400000US24	24	MD	Maryland	00	25151100280	6979966958	MULTIPOLYGON (((-76.04621 38.02553, -76.00734 ...	...	-3.991992	
1	19	01779785	0400000US19	19	IA	Iowa	00	144661267977	1084180812	POLYGON ((-96.62187 42.77925, -96.57794 42.827...	...	-1.278002	
2	10	01779781	0400000US10	10	DE	Delaware	00	5045925646	1399985648	POLYGON ((-75.77379 39.72220, -75.75323 39.757...	...	4.689728	
3	39	01085497	0400000US39	39	OH	Ohio	00	105828882568	10268850702	MULTIPOLYGON (((-82.86334 41.69369, -82.82572 ...	...	-0.698138	
4	42	01779798	0400000US42	42	PA	Pennsylvania	00	115884442321	3394589990	POLYGON ((-80.51989 40.90666, -80.51964 40.987...	...	-2.144836	

Step# 9

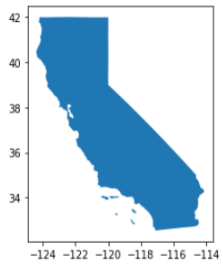
```
In [79]: pop_states[pop_states.NAME == "California"].plot()
         pop_states.head()
```

Out[79]:

	STATEFP	STATENS	AFFGEOID	GEOID	STUSPS	NAME	LSAD	ALAND	AWATER	geometry	...	RDOMESTICMIG2017	RDOMESTIC
0	24	01714934	0400000US24	24	MD	Maryland	00	25151100280	6979966958	MULTIPOLYGON (((-76.04621 38.02553, -76.00734 ...	...	-3.991992	
1	19	01779785	0400000US19	19	IA	Iowa	00	144661267977	1084180812	POLYGON ((-96.62187 42.77925, -96.57794 42.827...	...	-1.278002	
2	10	01779781	0400000US10	10	DE	Delaware	00	5045925646	1399985648	POLYGON ((-75.77379 39.72220, -75.75323 39.757...	...	4.689728	
3	39	01085497	0400000US39	39	OH	Ohio	00	105828882568	10268850702	MULTIPOLYGON (((-82.86334 41.69369, -82.82572 ...	...	-0.698138	
4	42	01779798	0400000US42	42	PA	Pennsylvania	00	115884442321	3394589990	POLYGON ((-80.51989 40.90666, -80.51964 40.987...	...	-2.144836	

4	42	01779798	0400000US42	42	PA	Pennsylvania	00	115884442321	3394589990	((-80.51989 40.90666, -80.51964 40.987...	...	-2.144836	

5 rows × 145 columns



Step# 10

```
In [80]: path = gplt.datasets.get_path("contiguous_usa")
         contiguous_usa= gpd.read_file(path)
         contiguous_usa.head()
```

Out[80]:

	state	adm1_code	population	geometry
0	Minnesota	USA-3514	5303925	POLYGON ((-89.59941 48.01027, -89.48888 48.013...
1	Montana	USA-3515	989415	POLYGON ((-111.19419 44.56116, -111.29155 44.7...
2	North Dakota	USA-3516	672591	POLYGON ((-96.60136 46.35136, -96.53891 46.199...
3	Idaho	USA-3518	1567582	POLYGON ((-111.04973 44.48816, -111.05025 42.0...
4	Washington	USA-3519	6724540	POLYGON ((-116.99807 46.33017, -116.90653 46.1...

```
In [48]: gplt.polyplot(contiguous_usa)
```

```
Out[48]: <AxesSubplot:>
```

Step# 11

jupyter Assignment__2 Last Checkpoint: 38 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
In [61]: path = gplt.datasets.get_path("usa_cities")
usa_cities= gpd.read_file(path)
usa_cities.head()
```

Out[61]:

	id	POP_2010	ELEV_IN_FT	STATE	geometry
0	53	40888.0	1611.0	ND	POINT (-101.29627 48.23251)
1	101	52838.0	830.0	ND	POINT (-97.03285 47.92526)
2	153	15427.0	1407.0	ND	POINT (-98.70844 46.91054)
3	177	105549.0	902.0	ND	POINT (-96.78980 46.87719)
4	192	17787.0	2411.0	ND	POINT (-102.78962 46.87918)

Step# 12

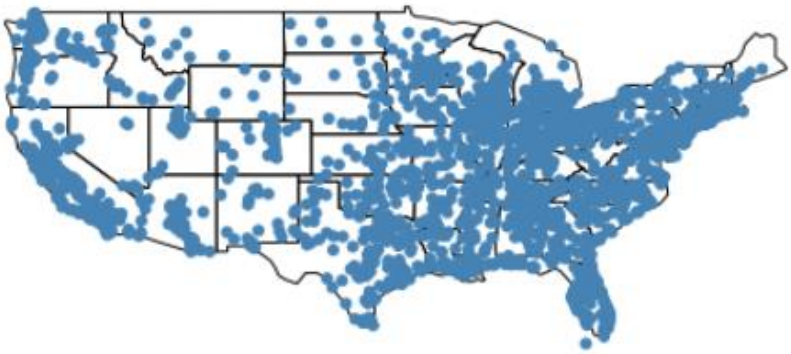
```
In [51]: continental_usa_cities = usa_cities.query('STATE not in ["HI", "AK", "PR"]')
         gplt.pointplot(continental_usa_cities)
```

```
Out[51]: <AxesSubplot:>
```

Step# 13

```
In [52]: ax = gplt.polyplot(contiguous_usa)
         gplt.pointplot(continental_usa_cities, ax= ax)
```

```
Out[52]: <AxesSubplot:>
```



Step# 14

```
In [53]: ax = gplt.polyplot(contiguous_usa, projection=gcrs.AlbersEqualArea())
         gplt.pointplot(continental_usa_cities, ax= ax)
```

```
Out[53]: <GeoAxesSubplot:>
```

Step# 15

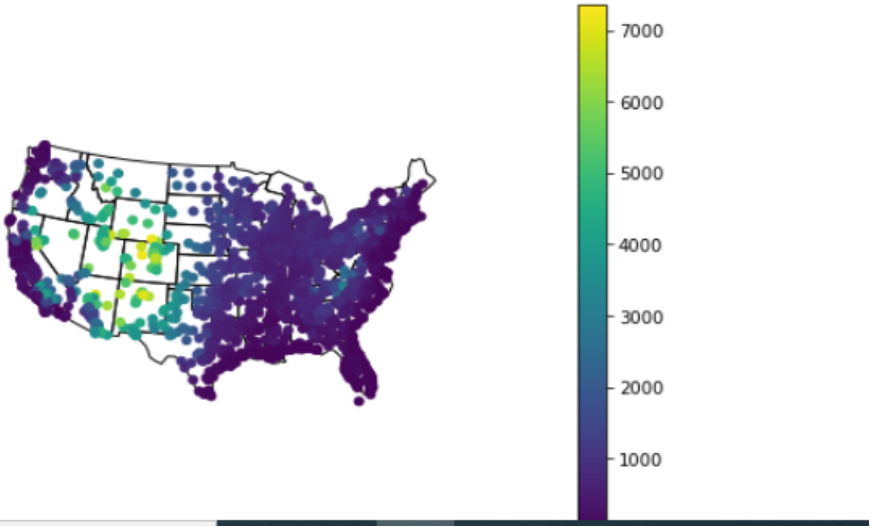
jupyter Assignment__2 Last Checkpoint: 41 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
In [54]: ax = gplt.polyplot(contiguous_usa, projection=gcrs.AlbersEqualArea())
         gplt.pointplot(
          continental_usa_cities, ax=ax,
          hue="ELEV_IN_FT",
          legend=True
          )
```

```
Out[54]: <GeoAxesSubplot:>
```



Step# 16

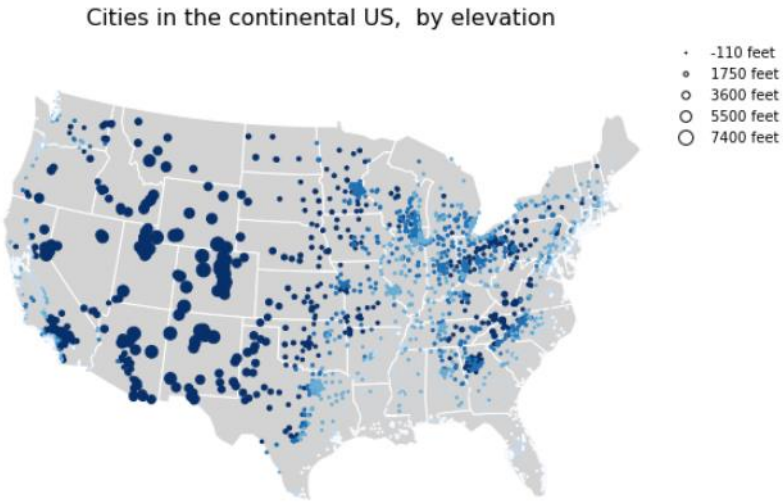
In [55]:
```
ax = gplt.polyplot(
    contiguous_usa,
    edgecolor = "white",
    facecolor = "lightgray",
    figsize = (12, 8),
    projection = gcrs.AlbersEqualArea()
  )
gplt.pointplot(
    continental_usa_cities,
    ax = ax,
    hue = 'ELEV_IN_FT',
   cmap = "Blues",
   scheme = 'quantiles',
   scale = 'ELEV_IN_FT',
   limits = (1, 10),
   legend = True,
   legend_var = 'scale',
   legend_kwargs = {"frameon": False},
   legend_values = [-110, 1750, 3600, 5500, 7400],
   legend_labels = ["-110 feet", "1750 feet", "3600 feet", "5500 feet", "7400 feet"]
)
ax.set_title("Cities in the continental US,  by elevation", fontsize = 16)
```

jupyter Assignment__2 Last Checkpoint: 42 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

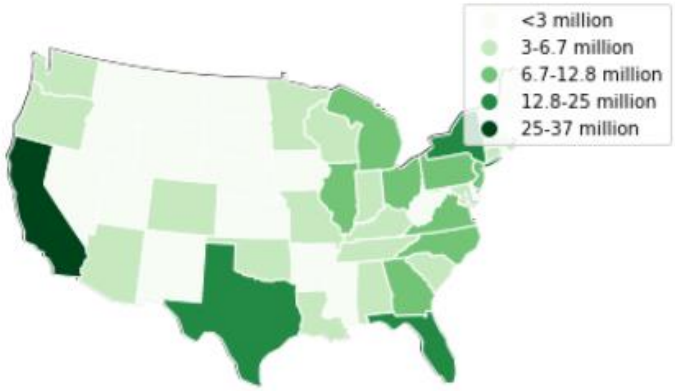
Out[55]: Text(0.5, 1.0, 'Cities in the continental US, by elevation')



Step# 17

```
In [56]: ax = gplt.polyplot(contiguous_usa, projection= gcrs.AlbersEqualArea())

gplt.choropleth(
    contiguous_usa,
    hue = "population",
    edgecolor = "white",
    linewidth = 1,
    cmap = "Greens",
    legend = True,
    scheme = "FisherJenks",
    legend_labels= [ "<3 million", "3-6.7 million", "6.7-12.8 million", "12.8-25 million" , "25-37 million"
   ],
    projection = gcrs.AlbersEqualArea(),
    ax=ax
    )
```

Out[56]: <GeoAxesSubplot:>



Step# 18

jupyter Assignment__2 Last Checkpoint: 44 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
In [60]: obesity = pd.read_csv(gplt.datasets.get_path("obesity_by_state"), sep= "\t")
obesity.head()
```

Out[60]:

	State	Percent
0	Alabama	32.4
1	Missouri	30.4
2	Alaska	28.4
3	Montana	24.6
4	Arizona	26.8

```
In [59]: geo_obesity = contiguous_usa.set_index("state").join(obesity.set_index("State"))
geo_obesity.head()
```

Out[59]:

state	adm1_code	population	geometry	Percent
Minnesota	USA-3514	5303925	POLYGON ((-89.59941 48.01027, -89.48888 48.013...	25.5
Montana	USA-3515	989415	POLYGON ((-111.19419 44.56116, -111.29155 44.7...	24.6
North Dakota	USA-3516	672591	POLYGON ((-96.60136 46.35136, -96.53891 46.199...	31.0
Idaho	USA-3518	1567582	POLYGON ((-111.04973 44.48816, -111.05025 42.0...	29.6
Washington	USA-3519	6724540	POLYGON ((-116.99807 46.33017, -116.90653 46.1...	27.2

Step# 19

jupyter Assignment__2 Last Checkpoint: 44 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [62]:
```
gplt.cartogram(
     geo_obesity,
     scale ="Percent",
     projection= gcrs.AlbersEqualArea()
)
```

Out[62]: <GeoAxesSubplot:>

Challenging Questions:

Q#1

```
Red channel:

array([[0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
       [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
       [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
       [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0]], dtype=uint32)


  Green Channel:

  array([[0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
         [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0],
         [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0],
         [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
         [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
         [1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2, 1, 1],
         [0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2, 1, 0],
         [0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1, 0, 0]], dtype=uint32)
```

```
Blue channle:

array([[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0],
       [0, 0, 0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1],
       [1, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2],
       [1, 1, 1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2],
       [1, 1, 1, 1, 2, 4, 5, 6, 7, 7, 7, 6, 5, 4, 2],
       [1, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 5, 4, 3, 2],
       [0, 0, 0, 0, 1, 2, 2, 3, 4, 4, 4, 3, 2, 2, 1],
       [0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0]], dtype=uint32)

Composite channel:

array([[[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0],
        [0, 1, 0],
        [0, 1, 0],
        [0, 2, 0],
        [1, 2, 0],
        [1, 2, 0],
        [1, 2, 1],
        [0, 2, 1],
        [0, 1, 1],
        [0, 1, 0],
        [0, 0, 0],
        [0, 0, 0],
```

Q#2

Quantile in basic words is the mid-range worth of information that partitions the information into comparative gatherings. There can be upper quantile or lower quantile. Quantile might be named as half or 0.5 on the grounds that previously or after there would be half qualities. It shows how our information is united or veered from the mid-range esteem. In an information on the off chance that we add more qualities the mid-range information might be changed appropriately, and our perception result may likewise change.

Q#3

jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

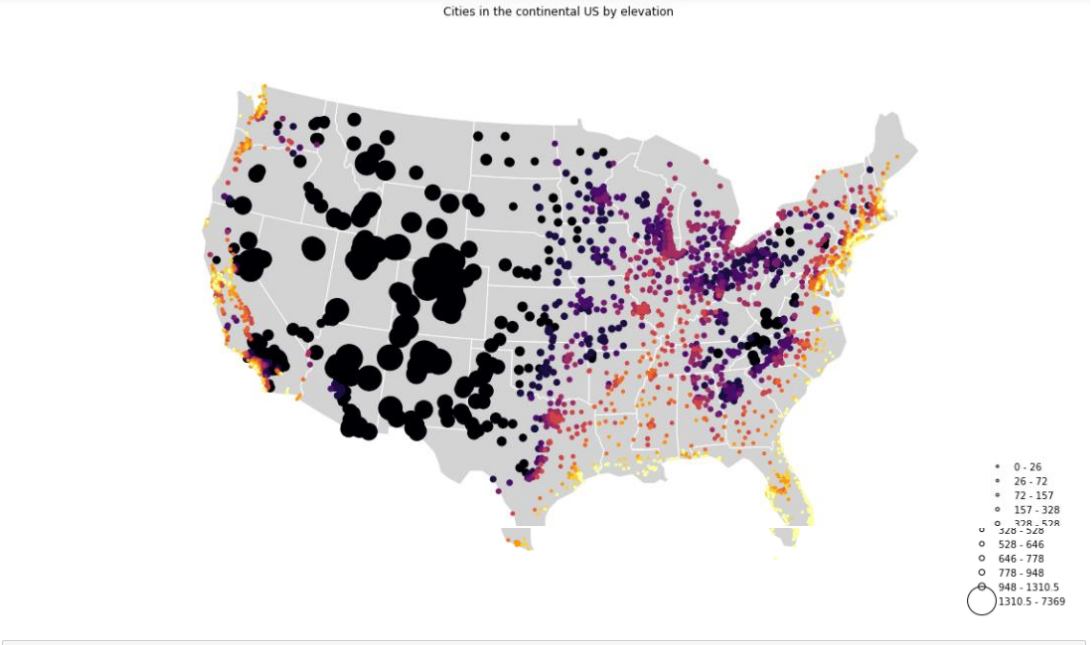
Run Code

In [68]:
```
usa = gpd.read_file("C:/Users/mubas/Downloads/us_state/us_state.shp")
continental_usa_cities = gpd.read_file(gplt.datasets.get_path('usa_cities'))
continental_usa_cities = continental_usa_cities.query('STATE not in ["AK", "HI", "PR"]')
contiguous_usa = gpd.read_file(gplt.datasets.get_path('contiguous_usa'))

scheme = mc.Quantiles(continental_usa_cities['ELEV_IN_FT'], k=10)
ax = gplt.polyplot(
    contiguous_usa,
    zorder=-1,
    linewidth =1 ,
    projection=gcrs.AlbersEqualArea(),
    edgecolor = 'white',
    facecolor ='lightgray',
    figsize =(20, 24))
gplt.pointplot(
    continental_usa_cities, scale ="ELEV_IN_FT" , limits= (2, 30),
    hue= "ELEV_IN_FT", cmap= 'inferno_r', scheme =scheme, legend=True,
    legend_var ='scale', legend_values=[26, 72, 157, 328, 528, 646, 778, 948, 1310.5, 7369],
    legend_labels = ['0 - 26','26 - 72', '72 - 157', '157 - 328', '328 - 528','528 - 646', '646 - 778', '778 - 948', '948 - 1310.
    legend_kwargs = {'frameon': False, 'loc':'lower right'},
    ax=ax
    )

plt.title("Cities in the continental US by elevation")
```

jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykern

Run Code

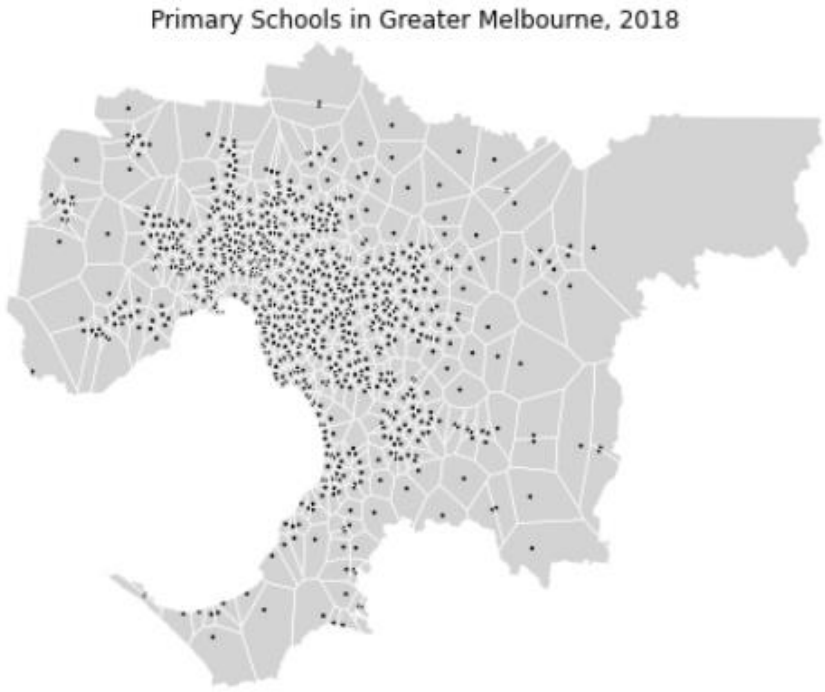


jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [70]:
```
melbourne = gpd.read_file(gplt.datasets.get_path("melbourne"))
df = gpd.read_file(gplt.datasets.get_path("melbourne_schools"))
melbourne_primary_schools = df.query('School_Type == "Primary"')

ax = gplt.voronoi(
   melbourne_primary_schools,
    clip = melbourne,
    linewidth = 0.5,
    edgecolor = "white",
    projection = gcrs.Mercator()
   )

gplt.polyplot(
    melbourne,
    edgecolor="None",
    facecolor= "lightgray",
    ax=ax
   )
gplt.pointplot(
    melbourne_primary_schools,
    color= "black",
    ax=ax,
    s=1,
    extent=melbourne.total_bounds
  )
plt.title("Primary Schools in Greater Melbourne, 2018")
plt.show()
```



Q#4

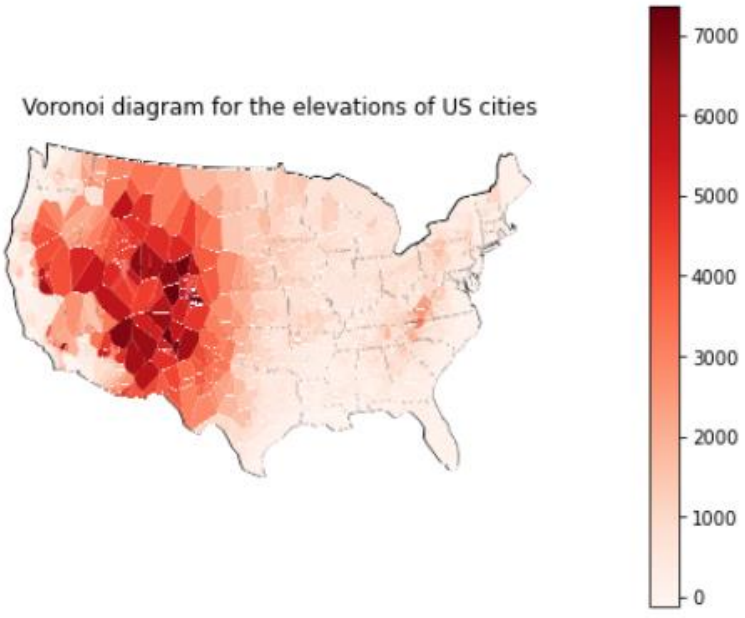
jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code

In [72]:
```
import geopandas as gpd
import geoplot as gplt
import geoplot.crs as gcrs
import matplotlib.pyplot as plt
import mapclassify as mc
import warnings
warnings.filterwarnings("ignore","Geoseries.isna", UserWarning)
import matplotlib.cbook
warnings.filterwarnings("ignore",category=matplotlib.cbook.mplDeprecation)

path = gplt.datasets.get_path("contiguous_usa")
contiguous_usa= gpd.read_file(path)
usa = gpd.read_file("C:/Users/mubas/Downloads/us_state/us_state.shp")
continental_usa_cities = gpd.read_file(gplt.datasets.get_path('usa_cities'))
continental_usa_cities = continental_usa_cities.query('STATE not in ["AK", "HI", "PR"]')
contiguous_usa_cities = gpd.read_file(gplt.datasets.get_path('contiguous_usa'))

continental_usa_cities= continental_usa_cities.drop_duplicates(subset=['geometry'])
proj = gplt.crs.AlbersEqualArea(central_longitude=-98, central_latitude = 39.5)

ax = gplt.voronoi(continental_usa_cities, hue="ELEV_IN_FT", clip=contiguous_usa,
    projection=proj, cmap="Reds", legend=True, edgecolor="white", linewidth=0.01)
gplt.polyplot(contiguous_usa, ax=ax, extent=contiguous_usa.total_bounds, edgecolor="black", linewidth=1, zorder=1)
plt.title("Voronoi diagram for the elevations of US cities")
```

Out[72]: Text(0.5, 1.0, 'Voronoi diagram for the elevations of US cities')



jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Run Code

In [73]:
```
df_confirmedGlobal  = pd.read_csv("C:/Users/mubas/Downloads/time_series_covid19_confirmed_global.csv")
df_confirmedGlobal
```

Out[73]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/31/20	11/1/20	11/2/20	11/3/20	11/4/20	11
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	41425	41501	41633	41728	41814	4
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	20875	21202	21523	21904	22300	2
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	57942	58272	58574	58979	59527	6
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	4756	4825	4888	4910	5045	
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	10805	11035	11228	11577	11813	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
263	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	53520	54060	54775	55408	56090	5
264	NaN	Western Sahara	24.215500	-12.885800	0	0	0	0	0	0	...	10	10	10	10	10	
265	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	2063	2063	2063	2063	2063	
266	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	16432	16480	16543	16661	16698	1
267	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	8367	8374	8389	8410	8427	

268 rows × 297 columns

In [74]:
```
df_confirmedGlobal = df_confirmedGlobal.drop(columns=['Province/State', 'Lat', 'Long'])
df_confirmedGlobal = df_confirmedGlobal.groupby('Country/Region').agg('sum')
date_list = list(df_confirmedGlobal.columns)
```

In [75]:
```
def get_country_code(name):
    try:
        return pycountry.countries.lookup(name).alpha_3
    except:
        return None

df_confirmedGlobal['country'] = df_confirmedGlobal.index
df_confirmedGlobal['iso_alpha_3'] = df_confirmedGlobal['country'].apply(get_country_code)
```

In [76]:
```
df_long = pd.melt(df_confirmedGlobal, id_vars=['country', 'iso_alpha_3'], value_vars=date_list)
df_long
```

Out[76]:

	country	iso_alpha_3	variable	value
0	Afghanistan	AFG	1/22/20	0
1	Albania	ALB	1/22/20	0
2	Algeria	DZA	1/22/20	0
3	Andorra	AND	1/22/20	0
4	Angola	AGO	1/22/20	0
...	...	...	...	...
55665	West Bank and Gaza	None	11/9/20	58838
55666	Western Sahara	ESH	11/9/20	10
55667	Yemen	YEM	11/9/20	2071
55668	Zambia	ZMB	11/9/20	16971
55669	Zimbabwe	ZWE	11/9/20	8561

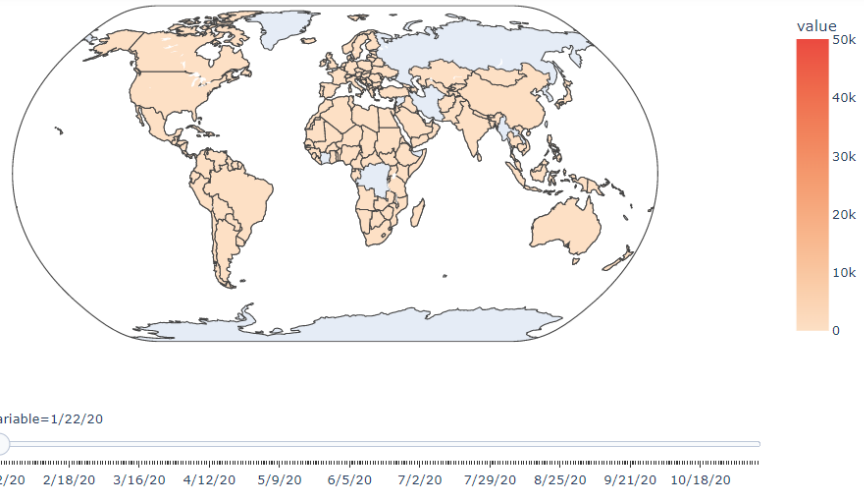
55670 rows × 4 columns

jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [77]:
```
fig = px.choropleth(df_long,
                    locations="iso_alpha_3",
                    color="value",
                    hover_name="country",
                    animation_frame="variable",
                    projection="natural earth",
                    color_continuous_scale='Peach',
                    range_color=(0, 50000)
                    )
fig.show()
fig.write_html("Covid19 map.html")
```



Q#5

jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [82]:
```
df_confirmedGlobal  = pd.read_csv("C:/Users/mubas/Downloads/time_series_covid19_recovered_global.csv")
df_confirmedGlobal
```

Out[82]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/31/20	11/1/20	11/2/20	11/3/20	11/4/20	11
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	34321	34326	34342	34355	34362	
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	11189	11246	11367	11473	11578	
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	40201	40395	40577	40577	41001	
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	3475	3475	3548	3627	3734	
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	4523	4920	5172	5230	5266	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
250	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	46309	46773	47169	47744	48224	
251	NaN	Western Sahara	24.215500	-12.885800	0	0	0	0	0	0	...	8	8	8	8	8	
252	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	1366	1366	1375	1375	1375	
253	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	15680	15733	15733	15763	15819	
254	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	7894	7927	7939	7942	7967	

255 rows × 297 columns

In [83]:
```
df_confirmedGlobal = df_confirmedGlobal.drop(columns=['Province/State', 'Lat', 'Long'])
df_confirmedGlobal = df_confirmedGlobal.groupby('Country/Region').agg('sum')
date_list = list(df_confirmedGlobal.columns)
```

In [84]:
```
def get_country_code(name):
    try:
        return pycountry.countries.lookup(name).alpha_3
    except:
        return None

df_confirmedGlobal['country'] = df_confirmedGlobal.index
df_confirmedGlobal['iso_alpha_3'] = df_confirmedGlobal['country'].apply(get_country_code)
```

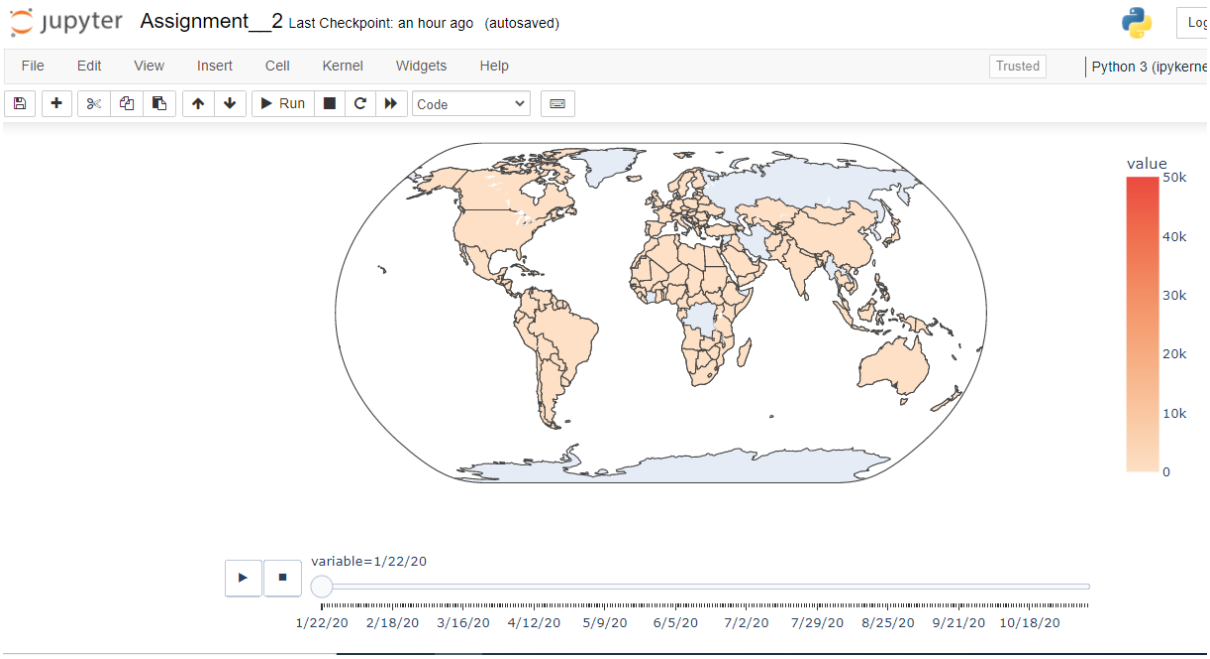
In [85]:
```
df_long = pd.melt(df_confirmedGlobal, id_vars=['country', 'iso_alpha_3'], value_vars=date_list)
df_long
```

Out[85]:

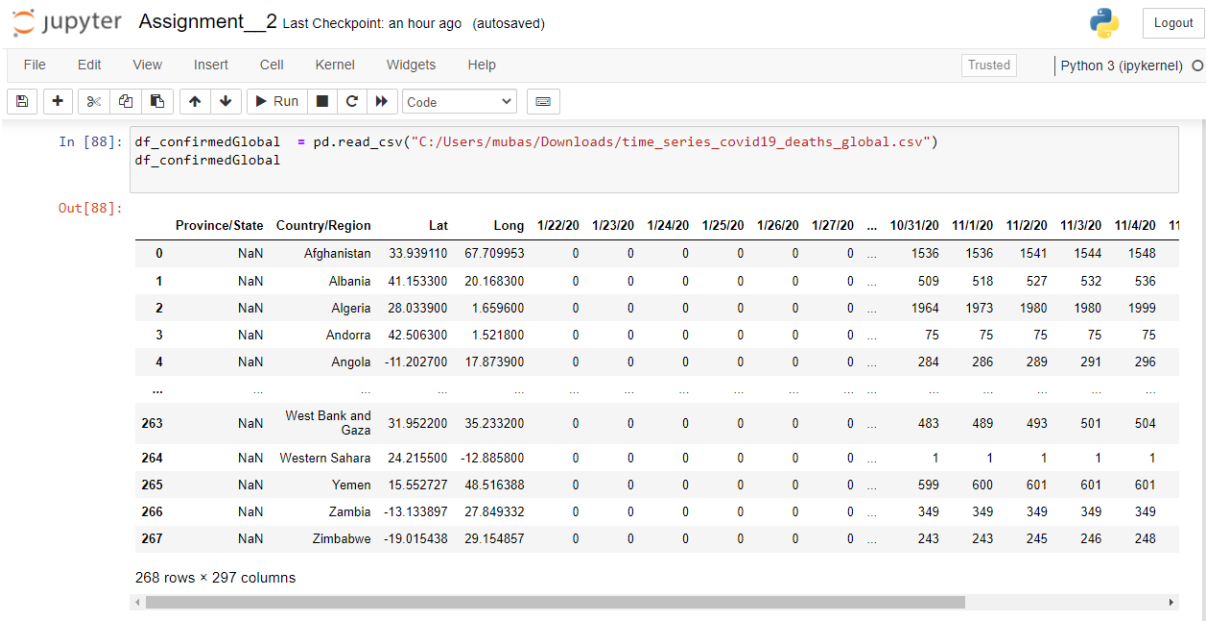
	country	iso_alpha_3	variable	value
0	Afghanistan	AFG	1/22/20	0
1	Albania	ALB	1/22/20	0
2	Algeria	DZA	1/22/20	0
3	Andorra	AND	1/22/20	0
4	Angola	AGO	1/22/20	0
...	...	...	...	...
55665	West Bank and Gaza	None	11/9/20	50877
55666	Western Sahara	ESH	11/9/20	8
55667	Yemen	YEM	11/9/20	1394
55668	Zambia	ZMB	11/9/20	16011
55669	Zimbabwe	ZWE	11/9/20	8023

55670 rows × 4 columns

In [86]:
```
fig = px.choropleth(df_long,
  locations="iso_alpha_3", color="value", hover_name="country", animation_frame="variable", projection="natural earth",
   color_continuous_scale='Peach', range_color=(0, 50000)
                    )
fig.show()
fig.write_html("Covid19 map_recovered.html")
```



Q#6



In [88]:
```
df_confirmedGlobal  = pd.read_csv("C:/Users/mubas/Downloads/time_series_covid19_deaths_global.csv")
df_confirmedGlobal
```

Out[88]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/31/20	11/1/20	11/2/20	11/3/20	11/4/20	11
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	1536	1536	1541	1544	1548	
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	509	518	527	532	536	
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	1964	1973	1980	1980	1999	
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	75	75	75	75	75	
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	284	286	289	291	296	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
263	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	483	489	493	501	504	
264	NaN	Western Sahara	24.215500	-12.885800	0	0	0	0	0	0	...	1	1	1	1	1	
265	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	599	600	601	601	601	
266	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	349	349	349	349	349	
267	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	243	243	245	246	248	

268 rows × 297 columns

jupyter Assignment__2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [89]:
```
df_confirmedGlobal = df_confirmedGlobal.drop(columns=['Province/State', 'Lat', 'Long'])
df_confirmedGlobal = df_confirmedGlobal.groupby('Country/Region').agg('sum')
date_list = list(df_confirmedGlobal.columns)
```

In [90]:
```
def get_country_code(name):
    try:
        return pycountry.countries.lookup(name).alpha_3
    except:
        return None

df_confirmedGlobal['country'] = df_confirmedGlobal.index
df_confirmedGlobal['iso_alpha_3'] = df_confirmedGlobal['country'].apply(get_country_code)
```

In [91]:
```
df_long = pd.melt(df_confirmedGlobal, id_vars=['country', 'iso_alpha_3'], value_vars=date_list)
df_long
```

Out[91]:

	country	iso_alpha_3	variable	value
0	Afghanistan	AFG	1/22/20	0
1	Albania	ALB	1/22/20	0
2	Algeria	DZA	1/22/20	0
3	Andorra	AND	1/22/20	0
4	Angola	AGO	1/22/20	0
...	...	...	...	...
55665	West Bank and Gaza	None	11/9/20	521
55666	Western Sahara	ESH	11/9/20	1
55667	Yemen	YEM	11/9/20	605
55668	Zambia	ZMB	11/9/20	349
55669	Zimbabwe	ZWE	11/9/20	254

55670 rows × 4 columns

In [92]:
```
fig = px.choropleth(df_long,
  locations="iso_alpha_3", color="value", hover_name="country", animation_frame="variable", projection="natural earth",
   color_continuous_scale='Peach', range_color=(0, 50000)
                    )
fig.show()
fig.write_html("Covid19 map_death.html")
```