In [ ]:

In [1]:

```
import psycopg2
%load_ext autoreload
%autoreload 2
import os
import sys
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import clean_data_functions as cdf
# module_path = os.path.abspath(os.path.join(os.pardir, os.pardir))
# if module_path not in sys.path:
#     sys.path.append(module_path)
# from src.data import data_collection
```

In [2]:

```
DBNAME = "opportunity_youth"
conn = psycopg2.connect(dbname=DBNAME)
#reproduce provided table
pd.read_sql("SELECT * FROM pums_2017 LIMIT 10;", conn);
```

In [3]:

```
# SQL query to select OY data in South King County
QUERY = """
    SELECT *
    FROM pums_2017 ps
    JOIN puma_names_2010 pn
    ON ps.puma = pn.puma
    WHERE pn.puma SIMILAR TO '1161(0|1|2|3|4|5)'
    AND ps.agep >= 16
    AND ps.agep <= 24
    AND ps.sch = '1'
    AND ps.dis = '2'
    AND ps.esr SIMILAR TO '%(3|6)%'
    ORDER BY pn
    """
db_south = pd.read_sql(QUERY, conn)
db_south
```

Out[3]:

	rt	serialno	division	sporder	puma	region	st	adjinc	pwgtp	agep	...	pwgt
0	P	2013000059060	9	02	11610	4	53	1061971	19.0	22.0	...	
1	P	2017000407899	9	02	11610	4	53	1011189	19.0	22.0	...	
2	P	2017000148009	9	04	11610	4	53	1011189	7.0	18.0	...	
3	P	2013000413288	9	03	11610	4	53	1061971	38.0	18.0	...	
4	P	2013000431365	9	03	11610	4	53	1061971	18.0	19.0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
317	P	2014001475081	9	07	11612	4	53	1045195	15.0	20.0	...	
318	P	2017000165892	9	06	11612	4	53	1011189	26.0	17.0	...	
319	P	2016000201076	9	02	11612	4	53	1029257	71.0	19.0	...	
320	P	2014000856804	9	04	11612	4	53	1045195	89.0	19.0	...	
321	P	2015000288090	9	02	11612	4	53	1035988	14.0	24.0	...	

322 rows × 293 columns

In [4]:

```
# declare df with desired pumas without duplicates from columns
db_south_pumas = db_south['puma'].drop_duplicates()
# remove duplicate column, coonvert topo list of pumas
db_south_pumas = db_south_pumas.loc[:,~db_south_pumas.columns.duplicated()]
pumas = list(db_south_pumas['puma'])
pumas
```

Out[4]:

```
['11610', '11613', '11614', '11615', '11611', '11612']
```

In [5]:

```
# load king county by king county puma data
filename = "../../src/data/shapefiles/tl_2017_53_puma10/tl_2017_53_puma10.shp"
all_pumas_shp = gpd.read_file(filename)
all_pumas_shp.head()
```

Out[5]:

	STATEFP10	PUMACE10	GEOID10	NAMELSAD10	MTFCC10	FUNCSTAT10	ALAND10	A
0	53	10200	5310200	Skagit, Island & San Juan Counties PUMA	G6120	S	5470622131	2
1	53	10100	5310100	Whatcom County--Bellingham City PUMA	G6120	S	5459332804	1
2	53	10400	5310400	Stevens, Okanogan, Pend Oreille & Ferry Counti...	G6120	S	29389124389	
3	53	10504	5310504	Spokane County (Outer)--Cheney City PUMA	G6120	S	3983412021	
4	53	10503	5310503	Spokane County (East Central)--Greater Spokane...	G6120	S	270926976	

In [6]:

```
# convert pumas column to int to match gdf/df entry types
all_pumas_shp.PUMACE10 = all_pumas_shp.PUMACE10.astype('int64')
# define new gdf, use range of pumas to represent south king county
south_king_shp = all_pumas_shp[all_pumas_shp.PUMACE10.between(11610,11615)]
south_king_shp
```

Out[6]:

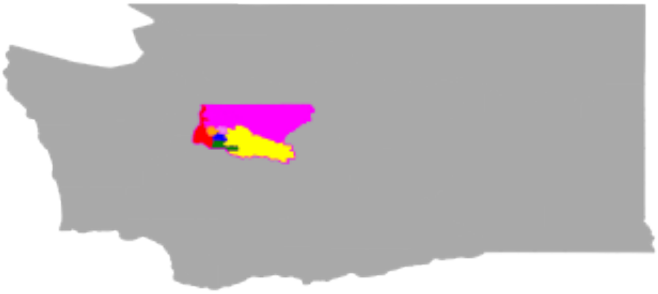
	STATEFP10	PUMACE10	GEOID10	NAMELSAD10	MTFCC10	FUNCSTAT10	ALAND10	A
9	53	11612	5311612	King County (Far Southwest)--Federal Way, Des ...	G6120	S	160638807	2
35	53	11611	5311611	King County (West Central)--Burien, SeaTac, Tu...	G6120	S	104685305	
41	53	11615	5311615	King County (Southeast)--Maple Valley, Covingt...	G6120	S	1704963276	
42	53	11614	5311614	King County (Southwest)--Auburn City & Lakelan...	G6120	S	177945706	
44	53	11613	5311613	King County (Southwest Central)--Kent City PUMA	G6120	S	96646675	
52	53	11610	5311610	King County (Central)--Renton City, Fairwood, ...	G6120	S	75931302	

In [7]:

```
# define new gdf with only king county shp info
king_pumas_shp = all_pumas_shp[(all_pumas_shp.PUMACE10.between(11601, 11616))]
```

In [8]:

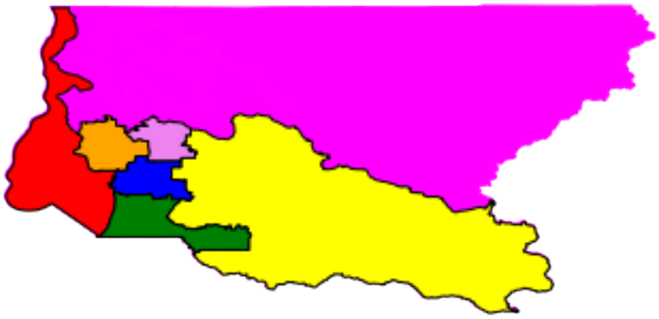
```
# create png of King County on Washington state
ax = all_pumas_shp.plot(color='darkgray', edgecolor='darkgrey', zorder=0 )
fig = king_pumas_shp.plot(ax=ax, color='fuchsia', edgecolor='fuchsia', zorder=1
)
plt.xticks([])
plt.yticks([])

labels = []
plots = []
colors = ['red', 'orange', 'yellow', 'green', 'blue', 'violet']

for i in range(len(south_king_shp)):
    row_entry = gpd.GeoDataFrame(south_king_shp.iloc[i]).transpose()
    region_name = row_entry.NAMELSAD10.iloc[0]
    labels.append(region_name)
    row_entry.plot(ax=ax, facecolor=colors[i], edgecolor='none', zorder=1, label
="1")
    plots.append(row_entry)

# plt.title('Regions of South King County')
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.axis('off');

ax.axis('off')
plt.savefig('figures/king_on_WA.png', transparent=True)
```



In [9]:

```
# create png of king county with color coated regions of south king county
# define county color for easy modification of King County on plot
king_county_color = 'magenta'

# create axis using king couty shapefile
ax = king_pumas_shp.plot(alpha=1, facecolor=king_county_color, edgecolor=king_co
unty_color)

# instantiate lists to be used in for loop
labels = []
plots = []
colors = ['red', 'orange', 'yellow', 'green', 'blue', 'violet']

# for loop to plot regions of south king county with distingct colors
for i in range(len(south_king_shp)):
    row_entry = gpd.GeoDataFrame(south_king_shp.iloc[i]).transpose()
    region_name = row_entry.NAMELSAD10.iloc[0]
    labels.append(region_name)
    row_entry.plot(ax=ax, facecolor=colors[i], edgecolor='k', zorder=1, label="1
")
    plots.append(row_entry)
    print(colors[i], '\n', region_name)

# turn off axes of plot
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.axis('off');

# save png of King county and disting southern regions
plt.savefig('figures/south_king_regions_on_king.png', transparent=True)
```

```
red
 King County (Far Southwest)--Federal Way, Des Moines Cities & Vasho
n Island PUMA
orange
 King County (West Central)--Burien, SeaTac, Tukwila Cities & White
Center PUMA
yellow
 King County (Southeast)--Maple Valley, Covington & Enumclaw Cities
PUMA
green
 King County (Southwest)--Auburn City & Lakeland PUMA
blue
 King County (Southwest Central)--Kent City PUMA
violet
 King County (Central)--Renton City, Fairwood, Bryn Mawr & Skyway PU
MA
```



In [10]:

```
# create dict formatted for pandas dataframe, utilizing user-defined functions
oy_pct_dict = cdf.get_puma_oy_percentages()
print("oy_pct_dict:", oy_pct_dict)

oy_pct_list = [[x for x in oy_pct_dict.keys()],[x for x in oy_pct_dict.values()]
]
print("oy_pct_list:", oy_pct_list)

oy_pct_df_dict = {'percent': oy_pct_list[1], 'puma': oy_pct_list[0]}
print("oy_pct_df_dict:", oy_pct_df_dict)

# finally, create dataframe
df_oy_pct = pd.DataFrame.from_dict(oy_pct_df_dict)

#change data type to match for comparison to shapefile
df_oy_pct.puma = df_oy_pct.puma.astype('int64')
df_oy_pct
```

```
oy_pct_dict: {'11610': 13.2, '11611': 14.6, '11612': 13.0, '11613':
12.2, '11614': 10.7, '11615': 10.1}
oy_pct_list: [['11610', '11611', '11612', '11613', '11614', '11615']
, [13.2, 14.6, 13.0, 12.2, 10.7, 10.1]]
oy_pct_df_dict: {'percent': [13.2, 14.6, 13.0, 12.2, 10.7, 10.1], 'p
uma': ['11610', '11611', '11612', '11613', '11614', '11615']}
```

Out[10]:

	percent	puma
0	13.2	11610
1	14.6	11611
2	13.0	11612
3	12.2	11613
4	10.7	11614
5	10.1	11615

In [11]:

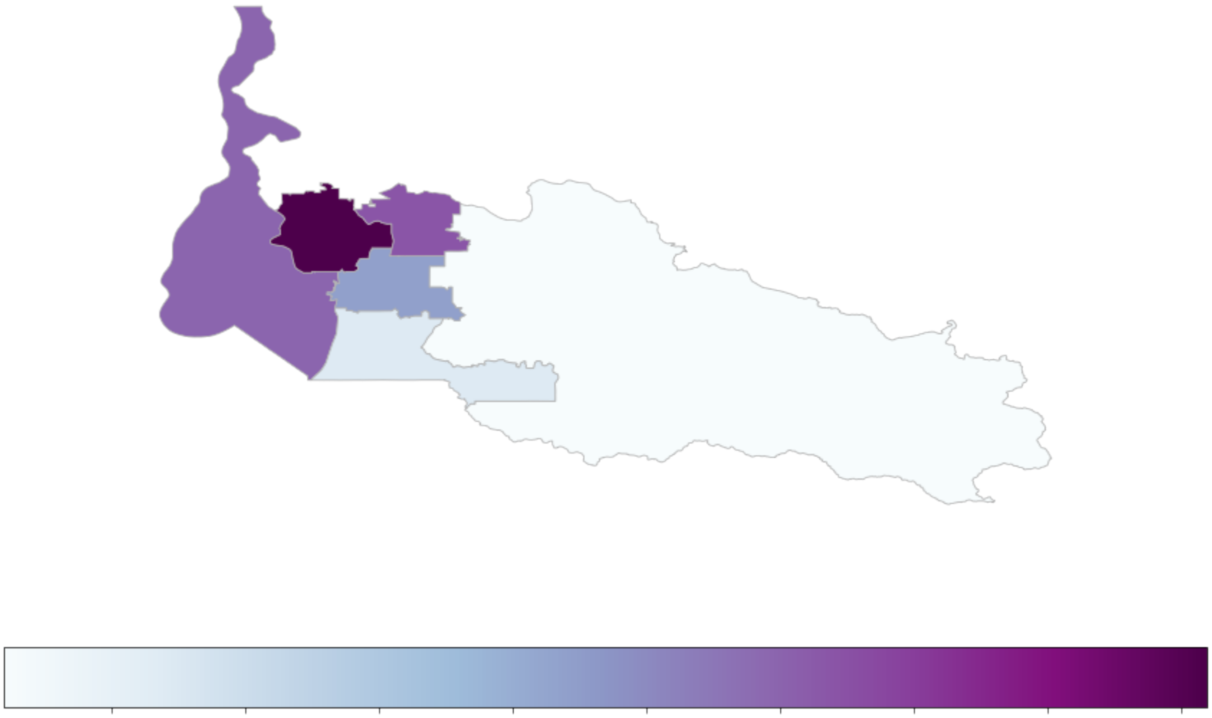
```
# create merged dataframe for choropleth map showing OY Percent By Region
merged_pct = south_king_shp.merge(df_oy_pct, left_on=south_king_shp.PUMACE10, ri
ght_on=df_oy_pct.puma)
merged_pct
```

Out[11]:

	key_0	STATEFP10	PUMACE10	GEOID10	NAMELSAD10	MTFCC10	FUNCSTAT10	ALANI
0	11612	53	11612	5311612	King County (Far Southwest)--Federal Way, Des ...	G6120	S	160638
1	11611	53	11611	5311611	King County (West Central)--Burien, SeaTac, Tu...	G6120	S	104685
2	11615	53	11615	5311615	King County (Southeast)--Maple Valley, Covingt...	G6120	S	1704963
3	11614	53	11614	5311614	King County (Southwest)--Auburn City & Lakelan...	G6120	S	177945
4	11613	53	11613	5311613	King County (Southwest Central)--Kent City PUMA	G6120	S	96646
5	11610	53	11610	5311610	King County (Central)--Renton City, Fairwood, ...	G6120	S	75931

In [12]:

```
#CHOROPLETH OY PERCENT BY REGION

# define column name for choro index
variable = 'percent'
# define range for choro intensities/alphas
vmin, vmax = min(oy_pct_dict.values()), max(oy_pct_dict.values())

#define figure/axis and figsize
fig, ax = plt.subplots(1, figsize=(15,10))

# choose styles for easy modification
cmap='BuPu'

merged_pct.plot(column=variable, cmap=cmap, linewidth=0.8, ax=ax, edgecolor='0.7
')
ax.axis('off')
# give title and extra space above map
# plt.title('Total Opportunity Youth By Region', fontsize=20, y=1.07)


# create choro legend
sm = plt.cm.ScalarMappable(cmap=cmap,
norm=plt.Normalize(vmin=vmin, vmax=vmax))

cbar = fig.colorbar(sm, orientation='horizontal')
# cbar.set_label('Total Opportunity Youth by Region', color='white', fontsize=15
)
plt.setp(plt.getp(cbar.ax.axes, 'xticklabels'), color='white', fontsize=25)

fig.savefig('figures/choropleth_percent_oy_by_region.png', transparent=True)
```



In [13]:

```
# get dict from team defined function
oy_count_dict = cdf.get_pums_oy_count()
oy_count_dict
```

Out[13]:

```
{'11610': 1853.0,
 '11611': 2038.0,
 '11612': 1977.0,
 '11613': 2006.0,
 '11614': 1530.0,
 '11615': 1210.0}
```

In [14]:

```
# create list to change to pandas friendly format
oy_count_list = [[x for x in oy_count_dict.keys()],[x for x in oy_count_dict.val
ues()]]
```

In [15]:

```
# create list to change to pandas friendly format
oy_df_dict = {'total': oy_count_list[1], 'puma': oy_count_list[0]}
oy_df_dict
```

Out[15]:

```
{'total': [1853.0, 2038.0, 1977.0, 2006.0, 1530.0, 1210.0],
 'puma': ['11610', '11611', '11612', '11613', '11614', '11615']}
```

In [16]:

```
# finally, create dataframe
df_oy = pd.DataFrame.from_dict(oy_df_dict)
#change data type to match for comparison to shapefile
df_oy.puma = df_oy.puma.astype('int64')
df_oy
```

Out[16]:

	total	puma
0	1853.0	11610
1	2038.0	11611
2	1977.0	11612
3	2006.0	11613
4	1530.0	11614
5	1210.0	11615

In [17]:

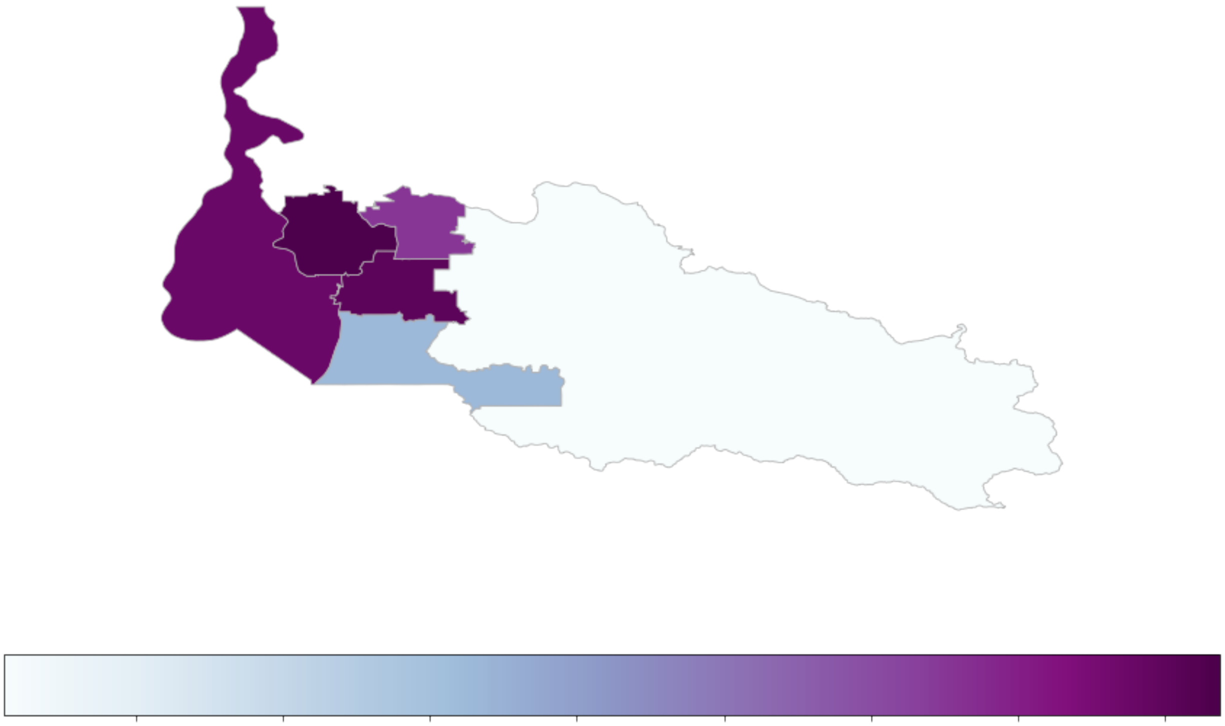
```
# create merged dataframe for choro
merged = south_king_shp.merge(df_oy, left_on=south_king_shp.PUMACE10, right_on=d
f_oy.puma)
merged
```

Out[17]:

	key_0	STATEFP10	PUMACE10	GEOID10	NAMELSAD10	MTFCC10	FUNCSTAT10	ALANI
0	11612	53	11612	5311612	King County (Far Southwest)--Federal Way, Des ...	G6120	S	160638
1	11611	53	11611	5311611	King County (West Central)--Burien, SeaTac, Tu...	G6120	S	104685
2	11615	53	11615	5311615	King County (Southeast)--Maple Valley, Covingt...	G6120	S	1704963
3	11614	53	11614	5311614	King County (Southwest)--Auburn City & Lakelan...	G6120	S	177945
4	11613	53	11613	5311613	King County (Southwest Central)--Kent City PUMA	G6120	S	96646
5	11610	53	11610	5311610	King County (Central)--Renton City, Fairwood, ...	G6120	S	75931

In [18]:

```
##CHOROPLETH OY TOTAL BY REGION

# define column name for choro index
variable = 'total'
# define range for choro intensities/alphas
vmin, vmax = min(oy_count_dict.values()), max(oy_count_dict.values())

#define figure/axis and figsize
fig, ax = plt.subplots(1, figsize=(15,10))

# choose styles for easy modification
cmap='BuPu'

merged.plot(column=variable, cmap=cmap, linewidth=0.8, ax=ax, edgecolor='0.7')
ax.axis('off')
# give title and extra space above map
# plt.title('Total Opportunity Youth By Region', fontsize=20, y=1.07)


# create choro legend
sm = plt.cm.ScalarMappable(cmap=cmap,
norm=plt.Normalize(vmin=vmin, vmax=vmax))

cbar = fig.colorbar(sm, orientation='horizontal')
# cbar.set_label('Total Opportunity Youth by Region', color='white', fontsize=15
)
plt.setp(plt.getp(cbar.ax.axes, 'xticklabels'), color='white', fontsize=25)

fig.savefig('figures/choropleth_total_oy_by_region.png', transparent=True)
```



In [ ]:

In [ ]:

In [19]:

```
def get_oy_db():
    #fetching oportunity youth in south king county
    skc_OY_df = pd.read_sql('''
        SELECT *
        FROM pums_2017
        WHERE puma SIMILAR TO '1161(0|1|2|3|4|5)'
        AND agep >= 16
        AND agep <= 24
        AND sch = '1'
        AND esr SIMILAR TO '%(3|6)%'
        ''', conn)

    return skc_OY_df

def get_all_youth_db():
    #fetching all residents from south king county within the OY age group
    skc_allRes_df = pd.read_sql('''
        SELECT *
        FROM pums_2017
        WHERE puma SIMILAR TO '1161(0|1|2|3|4|5)'
        AND agep >= 16
        AND agep <= 24
        ''', conn)

    return skc_allRes_df

def get_oy_2016_db():

    #fetching all opportunity youth from south king county in 2016

    csv_file_name = 'ss16pwa.csv'
    oy_2016_df = pd.read_csv(csv_file_name)
    puma_mask = oy_2016_df['PUMA'].isin(['11610', '11611', '11612', '11613', '11614', '11615'])
    oy_2016_df = oy_2016_df.loc[puma_mask]
    oy_mask = (oy_2016_df['AGEP'] >= 16) & (oy_2016_df['AGEP'] <= 24) & (oy_2016_df['SCH'].isin(['1'])) & (oy_2016_df['ESR'].isin(['3', '6']))
    oy_2016_df = oy_2016_df.loc[oy_mask]
    return oy_2016_df


def get_skc_oy_race():
    '''
    returns a dictionary with race names as keys and their coresponding pop_count as values
    '''
    skc_OY_df = get_oy_db()

    race_dict = {'1': 'White', '2': 'Black/ African American',
                 '3': 'American Indian or Alaska Native', '4': 'American Indian or Alaska Native',
                 '5': 'American Indian or Alaska Native', '6': 'Asian', '7': 'Native Hawaiian/ Pacific Islander',
                 '8': 'Other', '9': 'Two or More Races'}
    race_breakdown = skc_OY_df.groupby(by='rac1p').sum()['pwgtp']
    out_dict = {}
    for index in race_breakdown.index:
        if index in ['4', '5']:
            out_dict[race_dict[index]] += race_breakdown[index]
        else:
            out_dict[race_dict[index]] = race_breakdown[index]
    return out_dict


def get_skc_all_youth_race():
    '''
    returns a dictionary with race names as keys and their coresponding pop_count as values for all skc youth
    '''
    skc_allRes_df = get_all_youth_db()

    race_dict = {'1': 'White', '2': 'Black/ African American',
                 '3': 'American Indian or Alaska Native', '4': 'American Indian or Alaska Native',
                 '5': 'American Indian or Alaska Native', '6': 'Asian', '7': 'Native Hawaiian/ Pacific Islander',
                 '8': 'Other', '9': 'Two or More Races'}
    race_breakdown = skc_allRes_df.groupby(by='rac1p').sum()['pwgtp']
    out_dict = {}
    for index in race_breakdown.index:
        if index in ['4', '5']:
            out_dict[race_dict[index]] += race_breakdown[index]
        else:
            out_dict[race_dict[index]] = race_breakdown[index]
    return out_dict


def get_pums_youth_count():
    '''
    returns a dictionary with puma ID number as keys and their corresponding total youth count as values
    '''
    skc_all_youth_df = get_all_youth_db()

    puma_breakdown = skc_all_youth_df.groupby(by='puma').sum()['pwgtp']
    return puma_breakdown.to_dict()


def get_pums_oy_count():
    '''
    returns a dictionary with puma ID number as keys and their corresponding opportunity youth count as values
    '''
    skc_oy_df = get_oy_db()

    puma_breakdown = skc_oy_df.groupby(by='puma').sum()['pwgtp']
    return puma_breakdown.to_dict()
```

```
In [ ]:
```

In [20]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

In [21]:

```
youth_by_race = get_skc_all_youth_race()
```

In [22]:

```
youth_by_race
```

Out[22]:

```
{'White': 45663.0,
 'Black/ African American': 8920.0,
 'American Indian or Alaska Native': 961.0,
 'Asian': 13328.0,
 'Native Hawaian/ Pacific Islander': 1877.0,
 'Other': 7298.0,
 'Two or More Races': 7836.0}
```

In [23]:

```
oy_by_race = get_skc_oy_race()
```

In [24]:

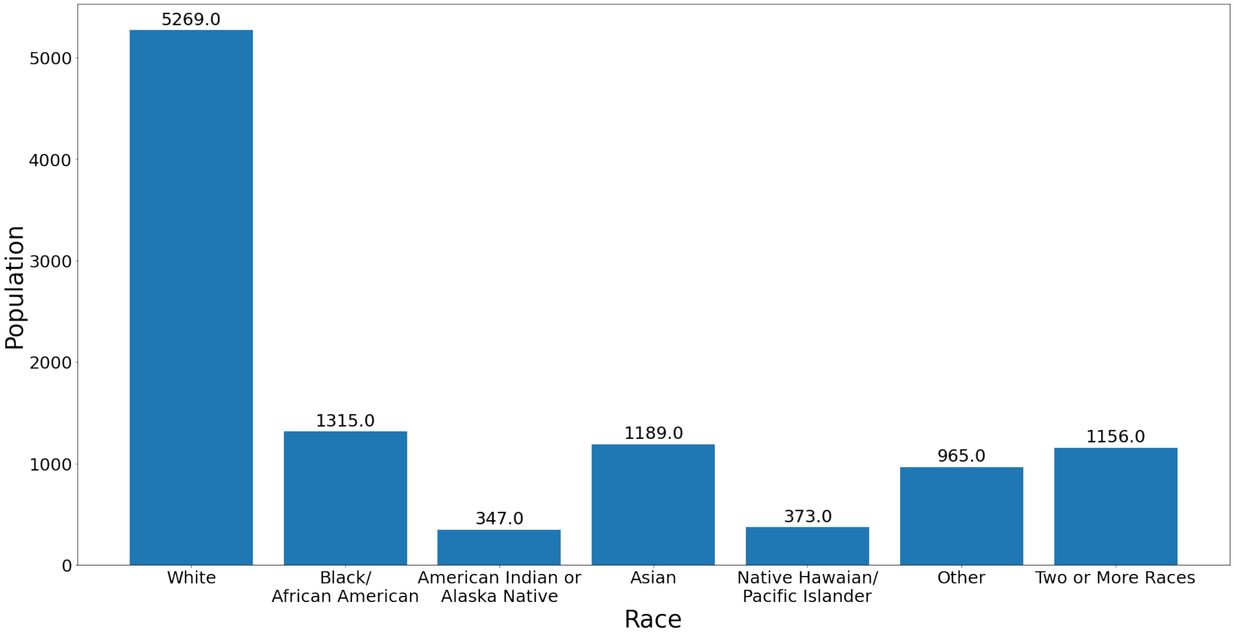
```
oy_by_race
```

Out[24]:

```
{'White': 5269.0,
 'Black/ African American': 1315.0,
 'American Indian or Alaska Native': 347.0,
 'Asian': 1189.0,
 'Native Hawaian/ Pacific Islander': 373.0,
 'Other': 965.0,
 'Two or More Races': 1156.0}
```

In [25]:

```
labels = ['White', 'Black/\nAfrican American', 'American Indian or\nAlaska Native',
          'Asian', 'Native Hawaiian/\nPacific Islander', 'Other', 'Two or More Races']

fig, ax =plt.subplots(figsize=(30, 15))
rect=plt.bar(x=labels, height=oy_by_race.values())
plt.xlabel('Race', fontsize=35)
plt.ylabel('Population', fontsize=35)
#plt.title('Opportunity Youth By Race in 2017', fontsize=25)
plt.xticks(fontsize=25)
plt.yticks(fontsize=25)
#plt.rcParams['figure.figsize'] = (30,10)

def autolabel(rects):
    for rect in rects:
        height = rect.get_height().round(1)
        ax.annotate('{}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom',
                    fontsize=25)
autolabel(rect)
plt.show()
#fig.savefig('OY_Race_2017.png')
```



In [26]:

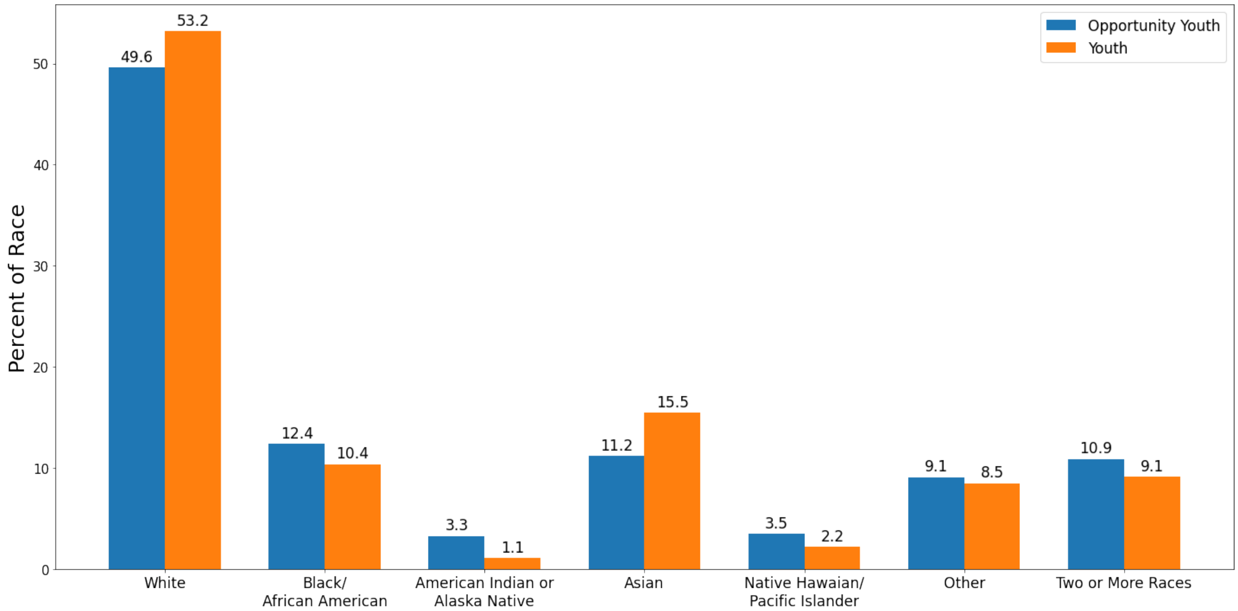
```
print(oy_by_race['White']/sum(oy_by_race.values()) * 100)
print(youth_by_race['White']/sum(youth_by_race.values()) * 100)
```

```
49.64198228754475
53.168845988146664
```

In [28]:

```
labels = ['White', 'Black/\nAfrican American', 'American Indian or\nAlaska Native',
          'Asian', 'Native Hawaian/\nPacific Islander', 'Other', 'Two or More Races']

oy_percent_by_race = [(oy_by_race['White']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['Black/ African American']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['American Indian or Alaska Native']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['Asian']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['Native Hawaian/ Pacific Islander']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['Other']/sum(oy_by_race.values())) * 100,
                      (oy_by_race['Two or More Races']/sum(oy_by_race.values())) * 100]

youth_percent_by_race = [(youth_by_race['White']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['Black/ African American']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['American Indian or Alaska Native']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['Asian']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['Native Hawaian/ Pacific Islander']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['Other']/sum(youth_by_race.values())) * 100,
                         (youth_by_race['Two or More Races']/sum(youth_by_race.values())) * 100]

x = np.arange(len(labels))
width = 0.35

fig, ax = plt.subplots(figsize=(20,10))
rects1 = ax.bar(x - width/2, oy_percent_by_race, width, label='Opportunity Youth')
rects2 = ax.bar(x + width/2, youth_percent_by_race, width, label='Youth')
```

```
ax.set_ylabel('Percent of Race', fontsize=25)
plt.yticks(fontsize=15)
#ax.set_title('Opportunity Youth % By Race vs Youth % By Race in 2017', fontsize=25)
ax.set_xticks(x)
ax.set_xticklabels(labels, fontsize=17)
ax.legend(fontsize=17)

def autolabel(rects):
    for rect in rects:
        height = rect.get_height().round(1)
        ax.annotate('{}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom',
                    fontsize=17)


autolabel(rects1)
autolabel(rects2)

fig.tight_layout()

fig1 = plt.gcf()
plt.show()
plt.draw()
#fig1.savefig('OY_Race_vs_Y_Race.png', dpi=100)
```



```
<Figure size 432x288 with 0 Axes>
```

In [29]:

```
skc_oy_race = get_skc_oy_race()
```

In [30]:

```
sum(skc_oy_race.values())
```

Out[30]:

```
10614.0
```

In [31]:

```
oy_in_skc = get_oy_db()
```

In [32]:

```
oy_in_skc_16_to_18_2017 = oy_in_skc.loc[(oy_in_skc['agep'] >= 16) & (oy_in_skc['agep'] <= 18)]
sum(oy_in_skc_16_to_18_2017['pwgtp'])

oy_in_skc_19_to_21_2017 = oy_in_skc.loc[(oy_in_skc['agep'] >= 19) & (oy_in_skc['agep'] <= 21)]
sum(oy_in_skc_19_to_21_2017['pwgtp'])

oy_in_skc_22_to_24_2017 = oy_in_skc.loc[(oy_in_skc['agep'] >= 22) & (oy_in_skc['agep'] <= 24)]
sum(oy_in_skc_22_to_24_2017['pwgtp'])
```

Out[32]:

```
4897.0
```

In [47]:

```
def get_oy_2016_db():

    #fetching all opportunity youth from south king county in 2016

    csv_file_name = 'ss16pwa.csv'
    oy_2016_df = pd.read_csv(''.join(['../../src/data/',csv_file_name]))
    puma_mask = oy_2016_df['PUMA'].isin(['11610', '11611', '11612', '11613', '11614', '11615'])
    oy_2016_df = oy_2016_df.loc[puma_mask]
    oy_mask = (oy_2016_df['AGEP'] >= 16) & (oy_2016_df['AGEP'] <= 24) & (oy_2016_df['SCH'].isin(['1'])) & (oy_2016_df['ESR'].isin(['3', '6']))
    oy_2016_df = oy_2016_df.loc[oy_mask]
    return oy_2016_df
```

In [48]:

```
oy_in_skc_16 = get_oy_2016_db()
```

In [49]:

```
oy_in_skc_16.head()
```

Out[49]:

	RT	SERIALNO	SPORDER	PUMA	ST	ADJINC	PWGTP	AGEP	CIT	CITWP	...
136	P	2012000003435	3	11615	53	1056030	17	19	1	NaN	...
1679	P	2012000039130	2	11611	53	1056030	13	24	1	NaN	...
1741	P	2012000040419	1	11613	53	1056030	33	19	5	NaN	...
3827	P	2012000083684	4	11614	53	1056030	30	17	5	NaN	...
3853	P	2012000084370	3	11610	53	1056030	43	22	1	NaN	...

5 rows × 283 columns

In [50]:

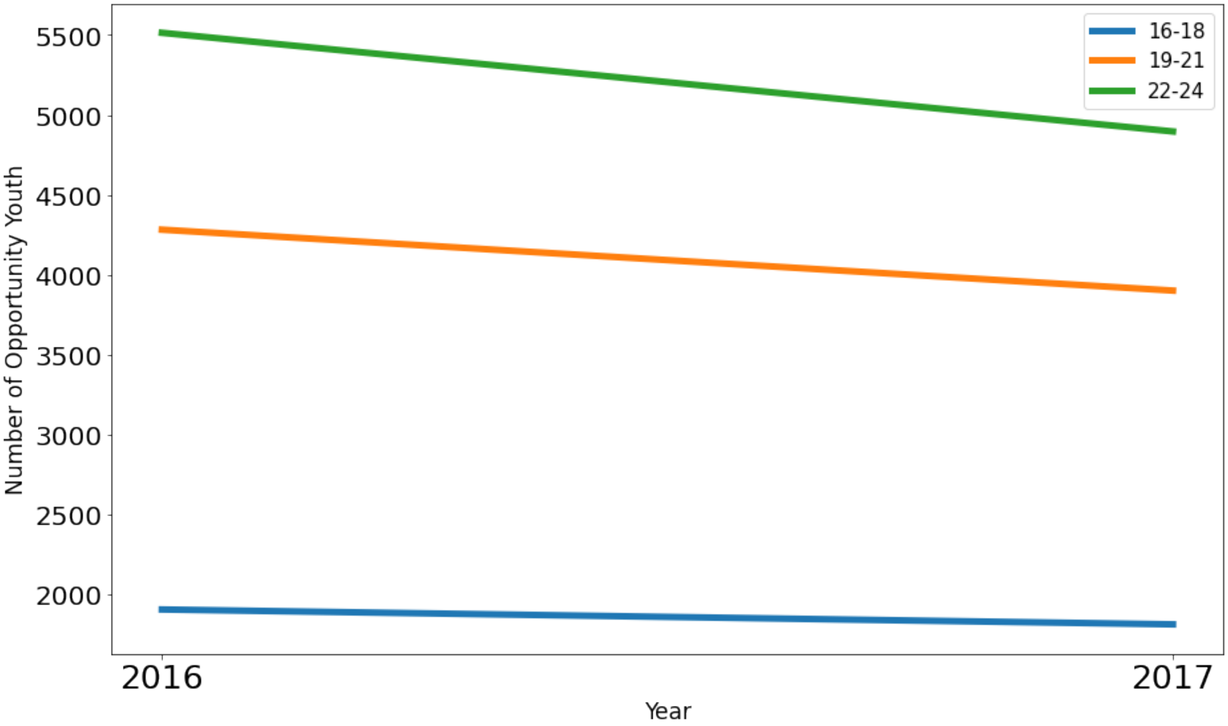
```
oy_in_skc_16_to_18_2016 = oy_in_skc_16.loc[(oy_in_skc_16['AGEP'] >= 16) & (oy_in_skc_16['AGEP'] <= 18)]
sum(oy_in_skc_16_to_18_2016['PWGTP'])

oy_in_skc_19_to_21_2016 = oy_in_skc_16.loc[(oy_in_skc_16['AGEP'] >= 19) & (oy_in_skc_16['AGEP'] <= 21)]
sum(oy_in_skc_19_to_21_2016['PWGTP'])

oy_in_skc_22_to_24_2016 = oy_in_skc_16.loc[(oy_in_skc_16['AGEP'] >= 22) & (oy_in_skc_16['AGEP'] <= 24)]
sum(oy_in_skc_22_to_24_2016['PWGTP'])
```

Out[50]:

```
5514
```

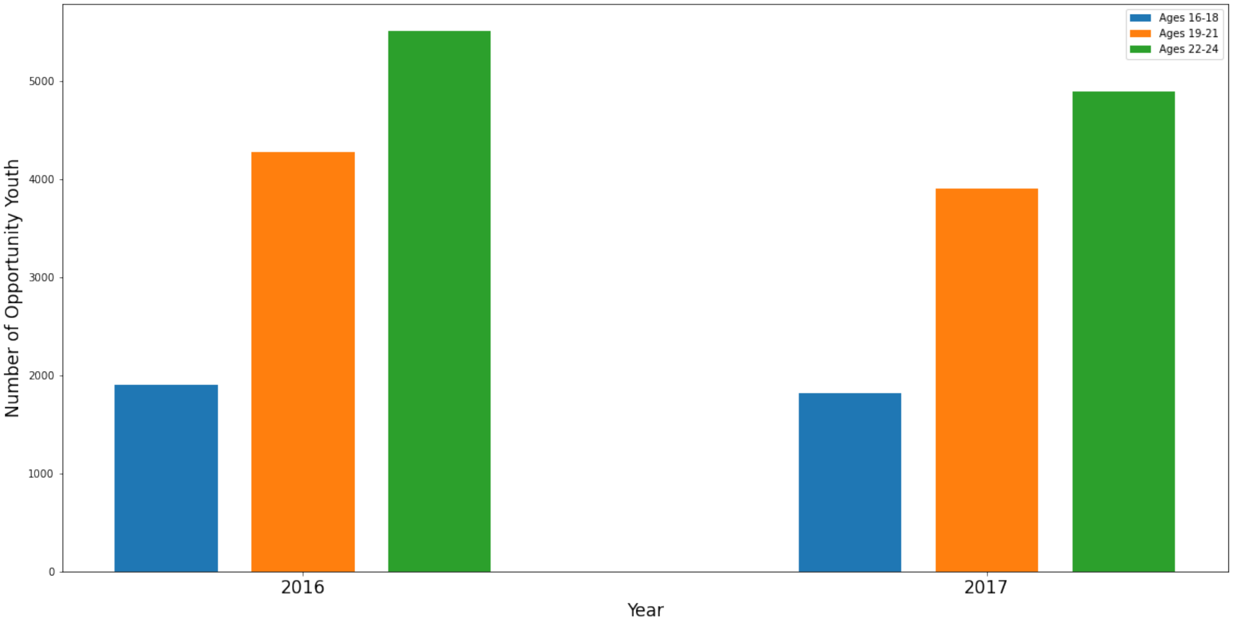
In [51]:

```
x1 = [2016, 2017]
y1 = [sum(oy_in_skc_16_to_18_2016['PWGTP']), sum(oy_in_skc_16_to_18_2017['pwgtp'
])]

x2 = [2016, 2017]
y2 = [sum(oy_in_skc_19_to_21_2016['PWGTP']), sum(oy_in_skc_19_to_21_2017['pwgtp'
])]

x3 = [2016, 2017]
y3 = [sum(oy_in_skc_22_to_24_2016['PWGTP']), sum(oy_in_skc_22_to_24_2017['pwgtp'
])]

fig, (ax) = plt.subplots(figsize=(15,9))
ax.plot(x1, y1, label='16-18', linewidth=5)
ax.plot(x2, y2, label='19-21', linewidth=5)
ax.plot(x3, y3, label='22-24', linewidth=5)
#ax.set_title('Comparison of Opportunity Youth by Age Group', fontsize=20)
ax.set_xlabel('Year', fontsize='17')
ax.set_xticks(np.arange(2016, 2018, step=1))
plt.xticks(fontsize=25)
plt.yticks(fontsize=20)
ax.set_ylabel('Number of Opportunity Youth', fontsize='17')
ax.legend(prop={'size':15})
plt.show()

plt.show()
#fig.savefig('OY_by_age_16_vs_17.png')
```


5500
5000
4500
4000
3500
3000
2500
2000
Number of Opportunity Youth
2016
2017
Year
16-18
19-21
22-24

In [52]:

```
labels = ['2016', '2017']

y = [sum(oy_in_skc_16_to_18_2016['PWGTP']), sum(oy_in_skc_16_to_18_2017['pwgtp'])]
z = [sum(oy_in_skc_19_to_21_2016['PWGTP']), sum(oy_in_skc_19_to_21_2017['pwgtp'])]
k = [sum(oy_in_skc_22_to_24_2016['PWGTP']), sum(oy_in_skc_22_to_24_2017['pwgtp'])]

x = np.arange(len(labels))

fig = plt.figure(figsize=(20, 10))
ax = plt.subplot()
ax.bar(x-0.2, y, width=0.15, align='center', label='Ages 16-18')
ax.bar(x, z, width=0.15, align='center', label='Ages 19-21')
ax.bar(x+0.2, k, width=0.15, align='center', label='Ages 22-24')
#ax.set_title('Comparison of Opportunity Youth by Age Group', fontsize=20)
ax.set_xlabel('Year', fontsize='17')
ax.set_ylabel('Number of Opportunity Youth', fontsize='17')
ax.set_xticks(x)
ax.set_xticklabels(labels, fontsize=17)
ax.legend()

plt.show()
#fig.savefig('OY_17_vs_16.png')
```



In [53]:

```
#fething oportunity youth in south king county

skc_OY_db = pd.read_sql('''
        SELECT *
        FROM pums_2017
        WHERE puma SIMILAR TO '1161(0|1|2|3|4|5)'
        AND agep >= 16
        AND agep <= 24
        AND sch = '1'
        AND esr SIMILAR TO '%(3|6)%'

''', conn)
skc_OY_db
```

Out[53]:

	rt	serialno	division	sporder	puma	region	st	adjinc	pwgtp	agep	...	pwg
0	P	2013000030421	9	02	11611	4	53	1061971	11.0	19.0	...	
1	P	2013000047506	9	04	11615	4	53	1061971	5.0	20.0	...	
2	P	2013000048962	9	05	11612	4	53	1061971	25.0	22.0	...	
3	P	2013000057563	9	05	11611	4	53	1061971	20.0	21.0	...	
4	P	2013000058010	9	02	11614	4	53	1061971	45.0	17.0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
386	P	2017001464049	9	01	11613	4	53	1011189	18.0	21.0	...	
387	P	2017001373291	9	02	11610	4	53	1011189	14.0	22.0	...	
388	P	2017001386502	9	01	11613	4	53	1011189	17.0	18.0	...	
389	P	2017001470135	9	01	11613	4	53	1011189	17.0	23.0	...	
390	P	2017001518359	9	01	11613	4	53	1011189	20.0	18.0	...	

391 rows × 286 columns

In [54]:

```
#fetching all residents from south king county within the OY age group

skc_allRes_db = pd.read_sql('''
        SELECT *
        FROM pums_2017
        WHERE puma SIMILAR TO '1161(0|1|2|3|4|5)'
        AND agep >= 16
        AND agep <= 24
''', conn)
skc_allRes_db
```

Out[54]:

	rt	serialno	division	sporder	puma	region	st	adjinc	pwgtp	agep	...	pwg
0	P	2013000003570	9	01	11610	4	53	1061971	20.0	24.0	...	
1	P	2013000003570	9	02	11610	4	53	1061971	15.0	24.0	...	
2	P	2013000007063	9	02	11612	4	53	1061971	30.0	19.0	...	
3	P	2013000008046	9	02	11613	4	53	1061971	36.0	17.0	...	
4	P	2013000010953	9	03	11610	4	53	1061971	15.0	18.0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
3477	P	2017001470135	9	01	11613	4	53	1011189	17.0	23.0	...	
3478	P	2017001503133	9	01	11610	4	53	1011189	12.0	18.0	...	
3479	P	2017001426098	9	01	11611	4	53	1011189	2.0	21.0	...	
3480	P	2017001474670	9	01	11611	4	53	1011189	5.0	16.0	...	
3481	P	2017001530818	9	01	11613	4	53	1011189	26.0	23.0	...	

3482 rows × 286 columns

In [55]:

```
#checking percentage of 16-24 year olds in skc who qualify as OY
#output is: percentage | # of OY in SKC | # of 16-24 year olds in SKC

print((sum(skc_OY_db['pwgtp'])/sum(skc_allRes_db['pwgtp']))*100, sum(skc_OY_db['
pwgtp']), sum(skc_allRes_db['pwgtp']))
```

```
12.358674009990334 10614.0 85883.0
```

create two subplots: one for each half of the OY status by age table

In [56]:

```
#create row and column labels for the total_pop table

age_range_labels = ['   ', '16-18', '19-21', '22-24', 'Totals']
tpop_rows = ['Total Poulation', 'Opportunity Youth', 'Working Without Diploma', 'Not an Opportunity Youth']

#create row and column labels for the opporunity_youth table

oy_rows = ['Oppurtunity Youth', 'No Diploma', 'Highschool Diploma or GED', 'Some College/ No Degree', 'Degree (Associate or Higher)']
```

In [57]:

```
#gather totals and percentages from colected data

age_ranges = [(16, 18), (19, 21), (22, 24)]
tpop_data = []
oy_data = []

#store totals for each age group for both tables
tpop_data.append([])
oy_data.append([])
total_tpop = 0
total_oy = 0
age_groupT_dfs = []
age_groupOY_dfs = []
for age_range in age_ranges:
    age_groupT_dfs.append(skc_allRes_db.loc[lambda db: (db['agep'] >= age_range[0]) & (db['agep'] <= age_range[1])])
    age_groupOY_dfs.append(skc_OY_db.loc[lambda db: (db['agep'] >= age_range[0]) & (db['agep'] <= age_range[1])])
for index in range(0, 3):
    tpop_total = age_groupT_dfs[index]['pwgtp'].sum()
    oy_total = age_groupOY_dfs[index]['pwgtp'].sum()
    tpop_data[0].append(['100%', tpop_total])
    total_tpop += tpop_total
    oy_data[0].append(['100%', oy_total])
    total_oy += oy_total
tpop_data[0].append(['100%', total_tpop])
oy_data[0].append(['100%', total_oy])

#fill in the rows for the total_pop table

oy_row = []
wnd_row = []
not_oy_row = []
for index in range(0, 3):
    wnd_mask = ((age_groupT_dfs[index]['esr'] == '1') | (age_groupT_dfs[index]['esr'] == '4')) & ((age_groupT_dfs[index]['schl'] == '14') | (age_groupT_dfs[index]['schl'] == '15'))
    wnd_df = age_groupT_dfs[index].loc[wnd_mask]
    not_oy_df = age_groupT_dfs[index].loc[wnd_mask ^ True]
    oy_row.append([f'{round((oy_data[0][index][1] / tpop_data[0][index][1]) * 100, 1)}%', oy_data[0][index][1]])
    wnd_total = wnd_df['pwgtp'].sum()
    wnd_row.append([f'{round((wnd_total / tpop_data[0][index][1]) * 100, 1)}%', wnd_total])
    not_oy = tpop_data[0][index][1] - oy_row[index][1] - wnd_row[index][1]
    not_oy_row.append([f'{round((not_oy / tpop_data[0][index][1]) * 100, 1)}%', not_oy])
tpop_data.append(oy_row)
tpop_data.append(wnd_row)
tpop_data.append(not_oy_row)


#fill in the totals for total_pop rows
for row in range(1, 4):
    row_total = sum([item[1] for item in tpop_data[row]])
    tpop_data[row].append([f'{round((row_total / tpop_data[0][3][1]) * 100, 1)}%', row_total])
    #sum_totals += row_total


#fill in row values and percentages for OY_pop table
no_dip_row = []
dip_row = []
no_deg_row = []
deg_row = []

###relevant schl value description
#
# '15': 12th grade/ no diploma
# '16' & '17': diploma/GED
# '18' & '19': some college but no degree
# '20' -> end: associates degree or better
#
###

for index in range(0, 3):
    col = age_groupOY_dfs[index]
    edu_breakdown = col.groupby(by='schl').sum()['pwgtp']
    no_dip = edu_breakdown.loc[:'15'].sum()
    dip = edu_breakdown.loc['16':'17'].sum()
    no_deg = edu_breakdown.loc['18':'19'].sum()
    deg = edu_breakdown.loc['20':].sum()
    no_dip_row.append([f'{round((no_dip/oy_data[0][index][1]) * 100, 1)}%', no_dip])
    dip_row.append([f'{round((dip/oy_data[0][index][1]) * 100, 1)}%', dip])
    no_deg_row.append([f'{round((no_deg/oy_data[0][index][1]) * 100, 1)}%', no_deg])
    deg_row.append([f'{round((deg/oy_data[0][index][1]) * 100, 1)}%', deg])
oy_data.append(no_dip_row)
oy_data.append(dip_row)
oy_data.append(no_deg_row)
oy_data.append(deg_row)

#fill in totals column

for row in range(1, 5):
    row_total = sum([item[1] for item in oy_data[row]])
    oy_data[row].append([f'{round((row_total / oy_data[0][3][1]) * 100, 1)}%', row_total])
    #sum_totals += row_total
```

In [58]:

```
#format data into 2d list of strings for entry into tables

tpop_cell_text = []
oy_cell_text = []
for row in tpop_data:
    text_row = [tpop_rows[tpop_data.index(row)]]
    for col in row:
        text_row.append(f'{col[0]}   {int(col[1])}')
    tpop_cell_text.append(text_row)
tpop_reformated = [[] for item in age_range_labels]
for row in tpop_cell_text:
    for cell in row:
        tpop_reformated[row.index(cell)].append(cell)
for row in oy_data:
    text_row = [oy_rows[oy_data.index(row)]]
    for col in row:
        text_row.append(f'{col[0]}   {int(col[1])}')
    oy_cell_text.append(text_row)
oy_reformated = [[] for item in age_range_labels]
for row in oy_cell_text:
    for cell in row:
        oy_reformated[row.index(cell)].append(cell)
oy_reformated
```

Out[58]:

```
[['Oppurtunity Youth',
  'No Diploma',
  'Highschool Diploma or GED',
  'Some College/ No Degree',
  'Degree (Associate or Higher)'],
 ['100%   1815', '50.5%   916', '43.0%   781', '6.5%   118', '0.0%
0'],
 ['100%   3902', '28.5%   1112', '55.8%   2176', '13.4%   521', '2.4
%   93'],
 ['100%   4897', '27.5%   1349', '43.6%   2135', '20.4%   1000', '8.
4%   413'],
 ['100%   10614',
  '31.8%   3377',
  '48.0%   5092',
  '15.4%   1639',
  '4.8%   506']]
```

In [59]:

```
tpop_cell_text
```

Out[59]:

```
[['Total Poulation',
  '100%   30141',
  '100%   25486',
  '100%   30256',
  '100%   85883'],
 ['Opportunity Youth',
  '6.0%   1815',
  '15.3%   3902',
  '16.2%   4897',
  '12.4%   10614'],
 ['Working Without Diploma',
  '8.8%   2655',
  '4.1%   1045',
  '3.0%   914',
  '5.4%   4614'],
 ['Not an Opportunity Youth',
  '85.2%   25671',
  '80.6%   20539',
  '80.8%   24445',
  '82.3%   70655']]
```

In [61]:

```
import plotly.graph_objects as go

tpop_fig = go.Figure(data=[go.Table(header=dict(values= age_range_labels, line_color='darkslategray', fill_color = 'lightblue', align='center'),
                                cells=dict(values= tpop_reformated, line_color='darkslategray', fill_color='lightgray', align='center'))])

oy_fig = go.Figure(data=[go.Table(header=dict(values= age_range_labels, line_color='darkslategray', fill_color = 'lightblue', align='center'),
                                cells=dict(values= oy_reformated, line_color='darkslategray', fill_color='lightgray', align='center'))])


tpop_fig.show()
oy_fig.show()
```

In [62]:

```
conn.close()
```

In [ ]:

```

```