Toronto Dwellings Analysis

This fundamental analysis is for the Toronto dwellings market to allow potential real estate investors to choose rental investment properties.

In [1]:

```
# initial imports
import os
import pandas as pd
import matplotlib.pyplot as plt
import hvplot.pandas
import plotly.express as px
from pathlib import Path
from dotenv import load_dotenv
import matplotlib
import numpy as np

%matplotlib inline
```

In [2]:

```
# Read the Mapbox API key
load_dotenv()
map_box_api = os.getenv("mapbox")
```

Load Data

In [3]:

```
# Read the census data into a Pandas DataFrame
file_path = Path("../Data/toronto_neighbourhoods_census_data.csv")
to_data = pd.read_csv(file_path, index_col="year")
to_data.head()
```

Out[3]:

year	neighbourhood	single_detached_house	apartment_five_storeys_plus	movable_dwelling	sem
2001	Agincourt North	3715	1480	0	
2001	Agincourt South-Malvern West	3250	1835	0	
2001	Alderwood	3175	315	0	
2001	Annex	1060	6090	5	
2001	Banbury-Don Mills	3615	4465	0	

Dwelling Types Per Year

In this section, you will calculate the number of dwelling types per year. Visualize the results using bar charts and the Pandas plot function.

Hint: Use the Pandas `groupby` function.

Optional challenge: Plot each bar chart in a different color.

In [5]:

```
# Calculate the mean number of dwelling types units per year (hint: use groupby)
housing_units = to_data.groupby(['year']).sum()
housing_units.head()
```

Out[5]:

year	single_detached_house	apartment_five_storeys_plus	movable_dwelling	semi_detached_house
2001	300930	355015	75	90995
2006	266860	379400	165	69430
2011	274940	429220	100	72480
2016	269680	493270	95	71200

In [6]:

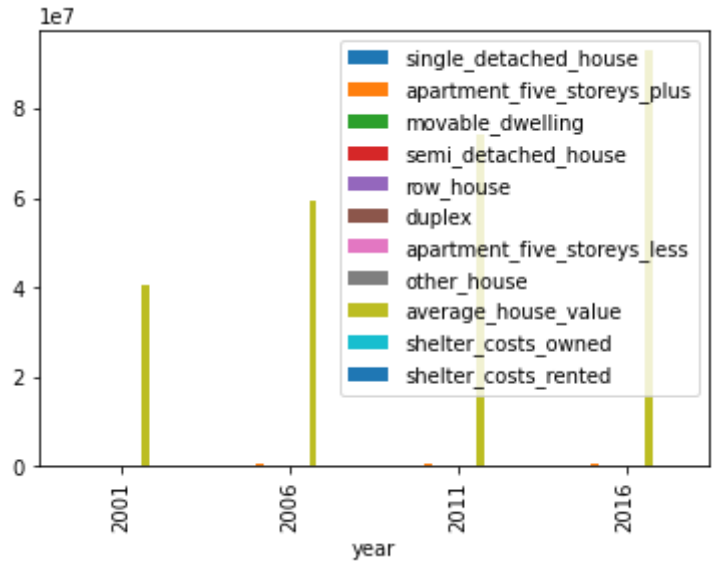
```
# Save the dataframe as a csv file
housing_units.to_csv('housing_type_avg_by_year.csv')
```

In [11]:

```
housing_units.plot.bar(2001)
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f851eba0b90>
```



In [ ]:

```
housing_units.drop(['average_house_value', 'shelter_costs_owned', 'shelter_costs_re
nted'],
                   axis=1,
                   inplace=True)
```

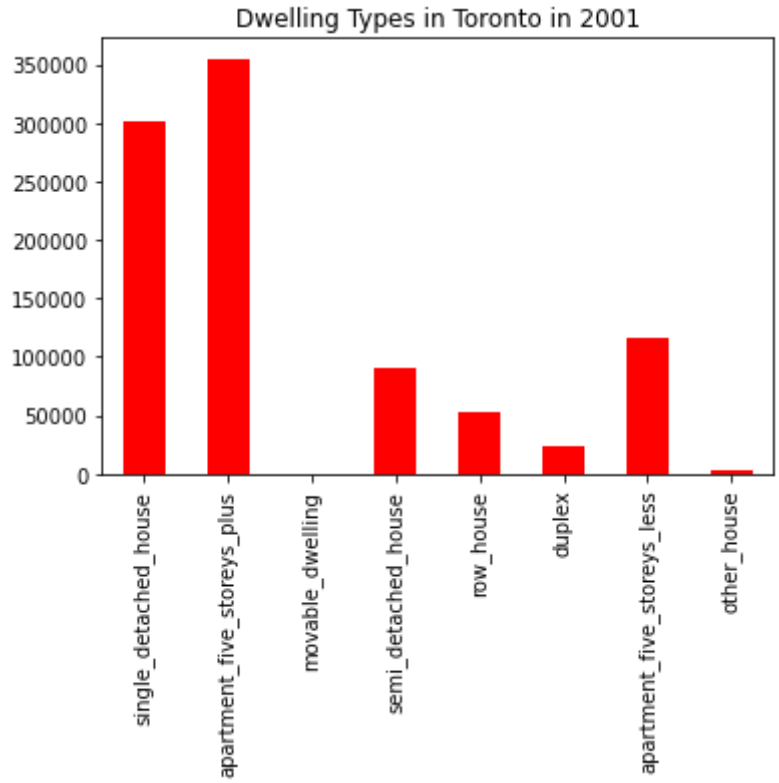
In [24]:

```
# Helper create_bar_chart function
#def create_bar_chart(data, title, xlabel, ylabel, color)

def create_bar_chart(year, data, xlabel, ylabel, color):
    fig = plt.figure()
    data.plot.bar(title = f'Dwelling Types in Toronto in {year}',
                        color = color)
    plt.show()
```

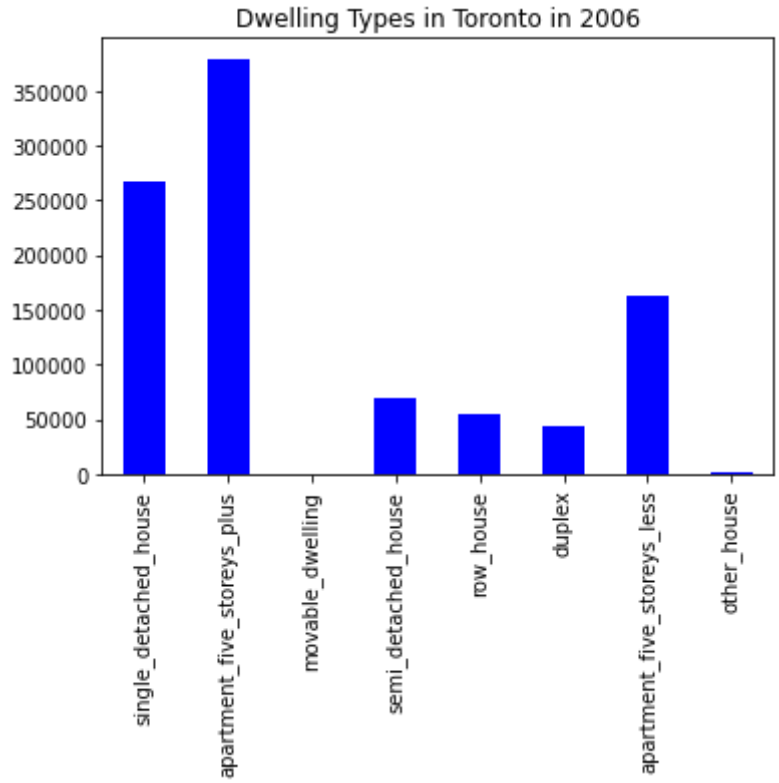
In [25]:

```
create_bar_chart('2001', housing_units.iloc[0],'','', 'red')
```



In [26]:

```
create_bar_chart('2006', housing_units.iloc[1],'','', 'blue')
```



In [11]:

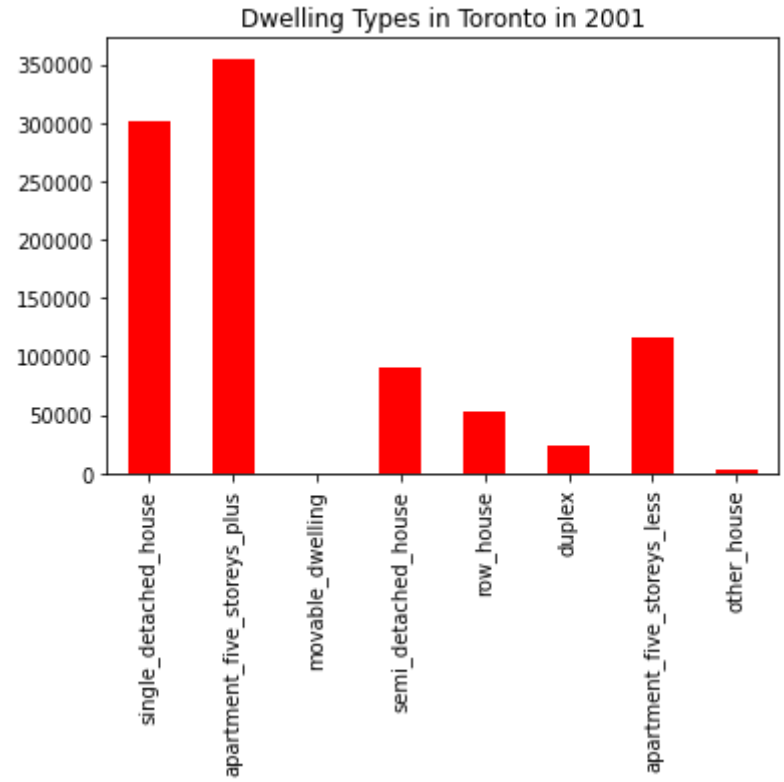
```
#Create a bar chart per year to show the number of dwelling types
housing_units.drop(['average_house_value', 'shelter_costs_owned', 'shelter_costs_re
nted'],
                   axis=1,
                   inplace=True)

dwellings_2001 = housing_units.iloc[0]

dwellings_2001.plot.bar(title = 'Dwelling Types in Toronto in 2001',
                        color = 'r')
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x12220de50>
```

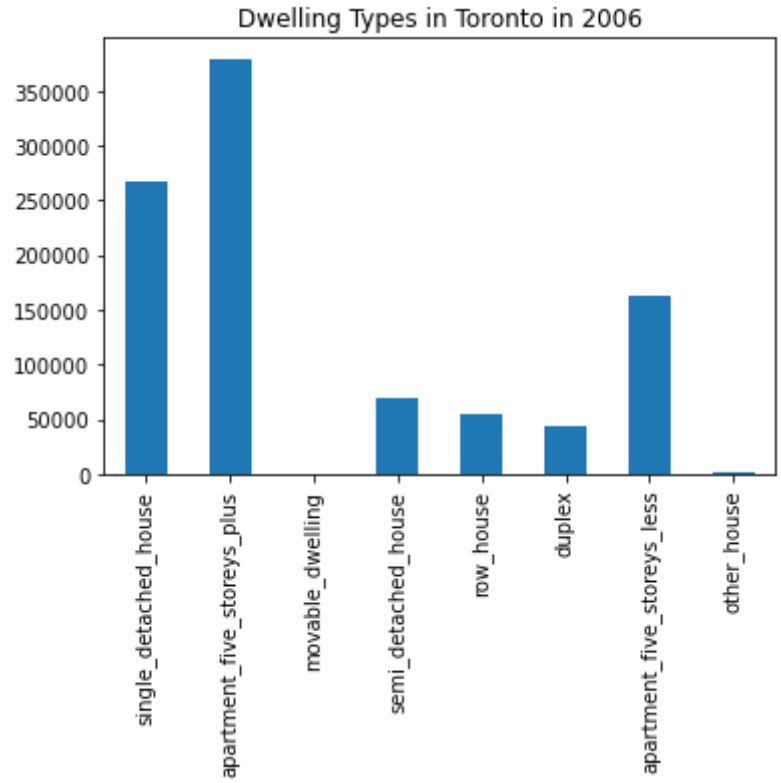


In [12]:

```
#Bar chart for 2006
dwellings_2006 = housing_units.iloc[1]
dwellings_2006.plot.bar(title = 'Dwelling Types in Toronto in 2006')
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1221be6d0>
```

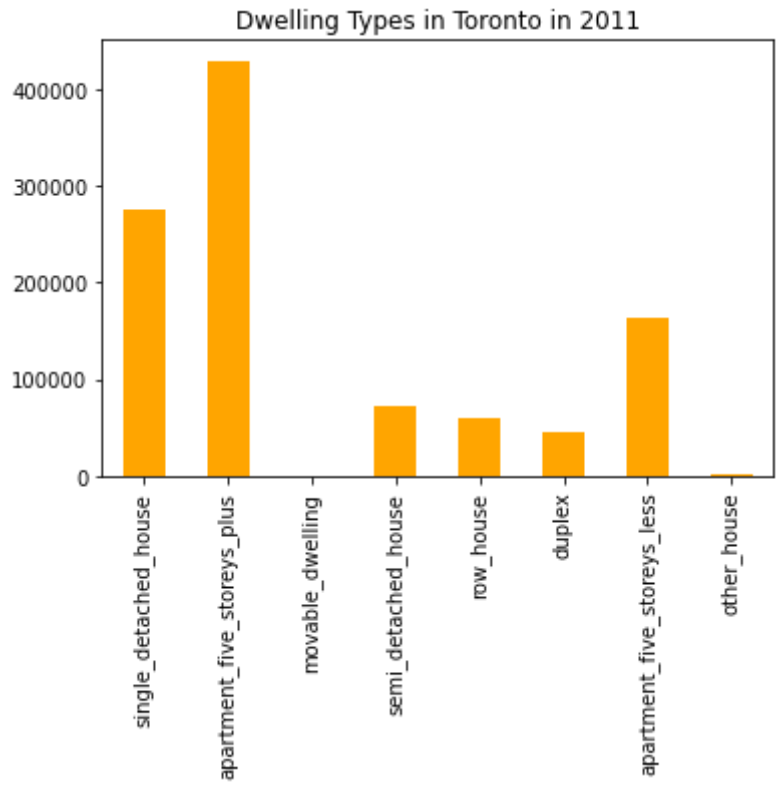


In [13]:

```
#Bar chart for 2011
dwellings_2011 = housing_units.iloc[2]
dwellings_2011.plot.bar(title = 'Dwelling Types in Toronto in 2011', color = 'orang
e')
```

Out[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x122327990>
```

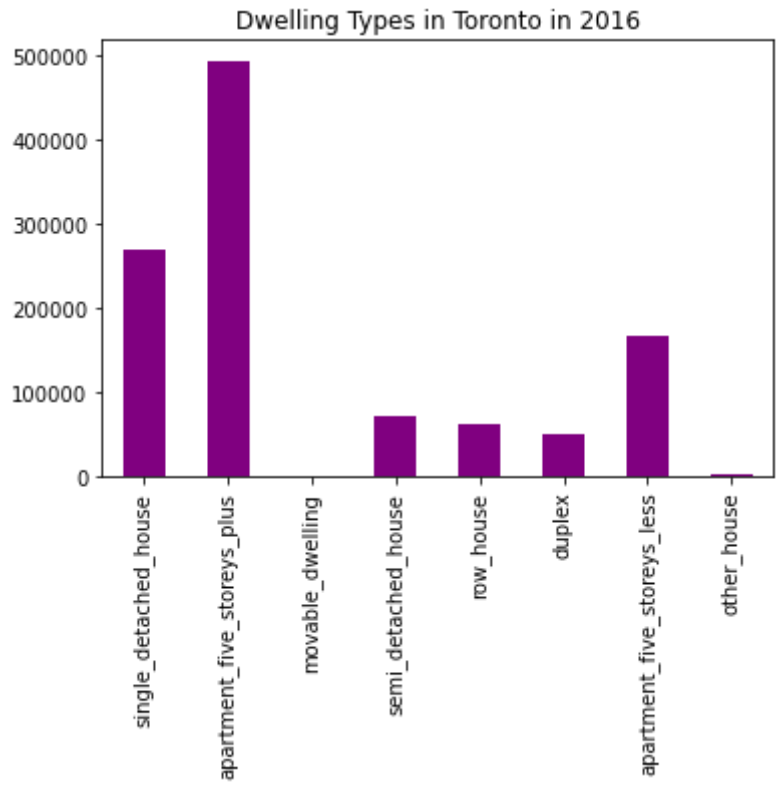


In [14]:

```
#Bar chart for 2016
dwellings_2016 = housing_units.iloc[3]
dwellings_2016.plot.bar(title = 'Dwelling Types in Toronto in 2016', color = 'purpl
e')
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1224a8dd0>
```



Average Monthly Shelter Costs in Toronto Per Year

In this section, you will calculate the average monthly shelter costs for owned and rented dwellings and the average house value for each year. Plot the results as a line chart.

Optional challenge: Plot each line chart in a different color.

In [28]:

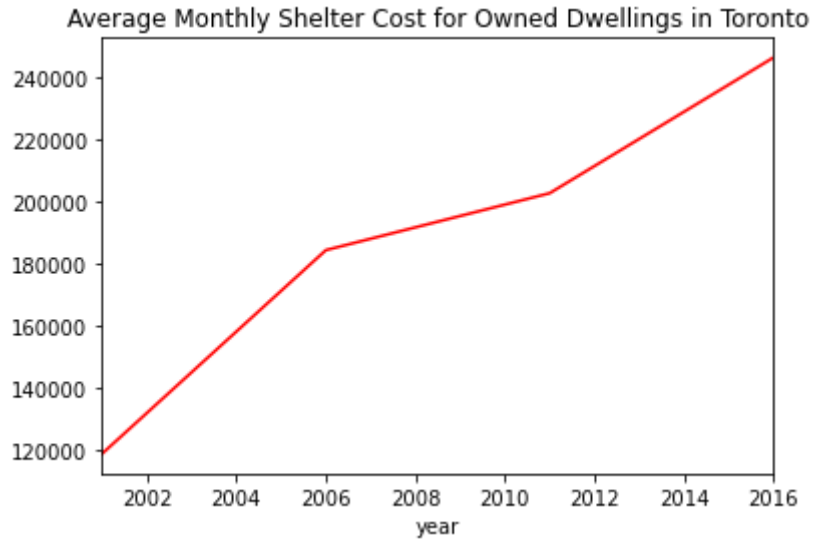
```
# Calculate the average monthly shelter costs for owned and rented dwellings
housing_units = to_data.groupby(['year']).sum()
housing_units.head()
shelter_cost_owned = housing_units.iloc[:,-2]
shelter_cost_rented = housing_units.iloc[:,-1]
```

In [40]:

```
def create_line_chart(title, data, xlabel, ylabel, color):
    fig = plt.figure()
    data.plot.line(title = title,
                               color = color)
    plt.show()
```

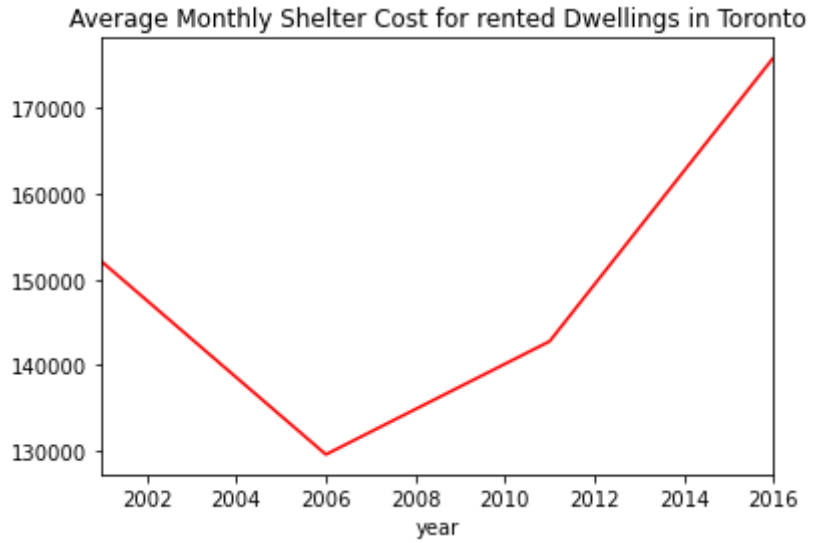
In [41]:

```
create_line_chart('Average Monthly Shelter Cost for Owned Dwellings in Toronto', sh
elter_cost_owned, '', '', color = 'red')
```



In [42]:

```
create_line_chart('Average Monthly Shelter Cost for rented Dwellings in Toronto', s
helter_cost_rented, '', '', color = 'red')
```

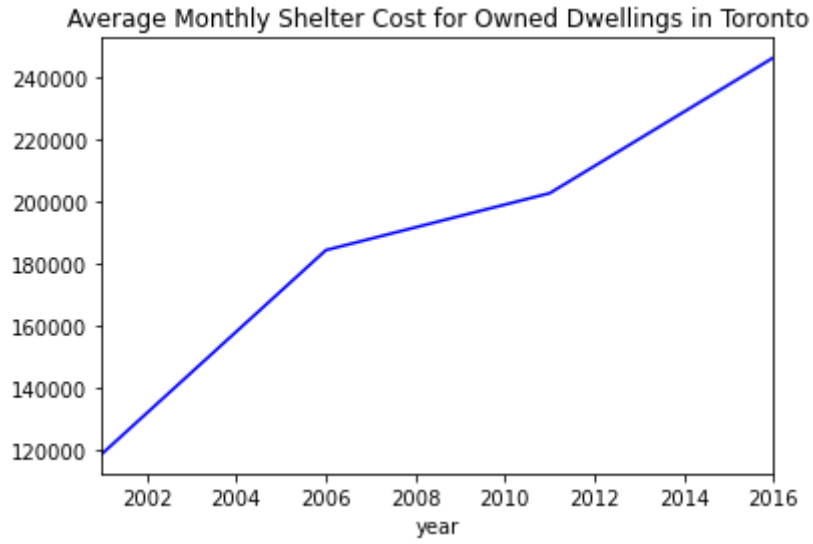


In [17]:

```
# Create two line charts, one to plot the monthly shelter costs for owned dwelleing
and other for rented dwellings per year
# Line chart for owned dwellings
shelter_cost_owned.plot.line(title = 'Average Monthly Shelter Cost for Owned Dwelli
ngs in Toronto',
                             color = 'blue')
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1226023d0>
```

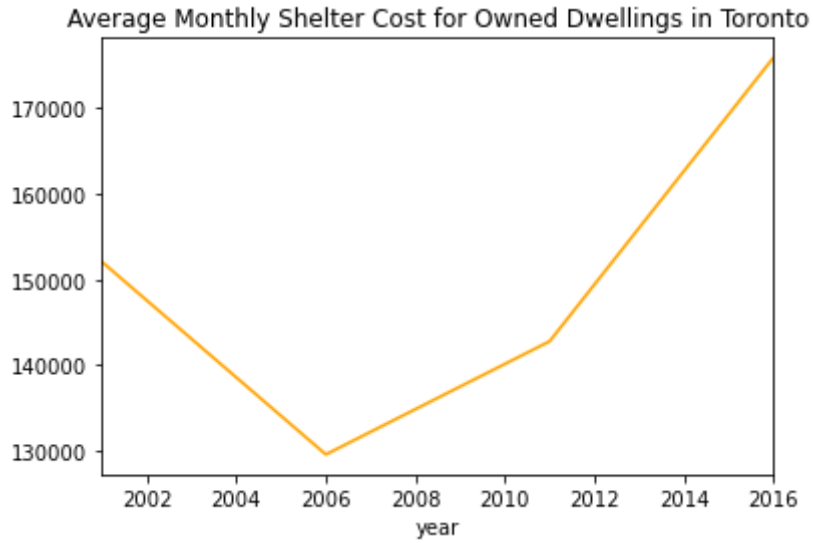


In [18]:

```
# Line chart for rented dwellings
shelter_cost_rented.plot.line(title = 'Average Monthly Shelter Cost for Owned Dwell
ings in Toronto',
                              color = 'orange')
```

Out[18]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1227cb890>
```



Average House Value per Year

In [19]:

```
# Calculate the average house value per
housing_units = to_data.groupby(['year']).mean()
avg_house_value = housing_units.iloc[:,-3]
avg_house_value
```

Out[19]:

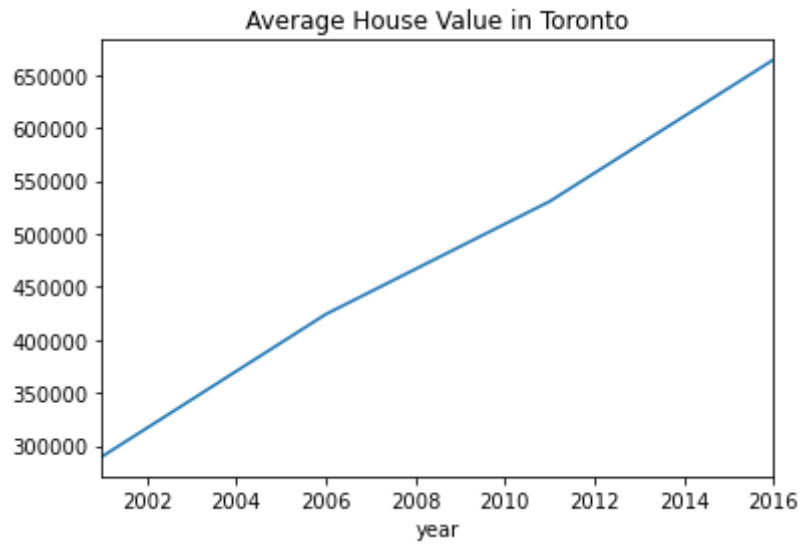
```
year
2001    289882.885714
2006    424059.664286
2011    530424.721429
2016    664068.328571
Name: average_house_value, dtype: float64
```

In [20]:

```
# Plot the average house value per year as a line chart
avg_house_value.plot.line(title = 'Average House Value in Toronto')
```

Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x122900710>
```



Average House Value by Neighbourhood

In this section, you will use `hvplot` to create an interactive visualization of the average house value with a dropdown selector for the neighbourhood.

Hint: It will be easier to create a new DataFrame from grouping the data and calculating the mean house values for each year and neighbourhood.

In [21]:

```
# Create a new DataFrame with the mean house values by neighbourhood per year
to_new_data = to_data.groupby([to_data.index, "neighbourhood"]).mean()
to_house_value = to_new_data["average_house_value"]
to_house_value = pd.DataFrame(to_house_value).reset_index()
to_house_value.head()
```

Out[21]:

	year	neighbourhood	average_house_value
0	2001	Agincourt North	200388
1	2001	Agincourt South-Malvern West	203047
2	2001	Alderwood	259998
3	2001	Annex	453850
4	2001	Banbury-Don Mills	371864

In [22]:

```
# Use hvplot to create an interactive line chart of the average house value per nei
ghbourhood
# The plot should have a dropdown selector for the neighbourhood

to_house_value.hvplot(
    x='year',
    xlabel = 'Year',
    y='average_house_value',
    ylabel = 'Avg. House Value',
    groupby = 'neighbourhood',
    kind = 'line')
```

Out[22]:

Number of Dwelling Types per Year

In this section, you will use `hvplot` to create an interactive visualization of the average number of dwelling types per year with a dropdown selector for the neighbourhood.

Hint: It will be easier to create a new DataFrame from grouping the data and calculating the mean number of dwelling types for each year and neighbourhood.

In [23]:

```
# Fetch the data of all dwelling types per year
to_data.head()
dwelling_type = to_data.groupby([to_data.index, "neighbourhood"]).mean()

dwelling_type.hvplot(
    x='year',
    xlabel = 'Year',
    y=['single_detached_house',
       'apartment_five_storeys_plus',
       'movable_dwelling',
       'semi_detached_house',
       'row_house',
       'duplex',
       'apartment_five_storeys_less',
       'other_house'
      ],
    ylabel = 'Avg. House Value',
    groupby = 'neighbourhood',
    kind = 'bar',
    stacked = False,
    rot = 90,
    height = 500)
```

Out[23]:

In [24]:

```
# Use hvplot to create an interactive bar chart of the number of dwelling types per neighbourhood
# The plot should have a dropdown selector for the neighbourhood
# YOUR CODE HERE!

dwelling_type.hvplot(
    x='year',
    xlabel = 'Year',
    y='average_house_value',
    ylabel = 'Avg. House Value',
    groupby = 'neighbourhood',
    kind = 'bar',
    stacked = False)
```

Out[24]:

The Top 10 Most Expensive Neighbourhoods

In this section, you will need to calculate the house value for each neighbourhood and then sort the values to obtain the top 10 most expensive neighbourhoods on average. Plot the results as a bar chart.

In [25]:

```
# Getting the data from the top 10 expensive neighbourhoods
top_10_most_expensive = to_data.sort_values(by='average_house_value', ascending=False).head(10)
top_10_most_expensive
```

Out[25]:

year	neighbourhood	single_detached_house	apartment_five_storeys_plus	movable_dwelling	sem
2016	Bridle Path-Sunnybrook-York Mills	2275	590	0	
2011	Bridle Path-Sunnybrook-York Mills	2285	480	0	
2016	Forest Hill South	1685	2025	0	
2016	Lawrence Park South	3420	925	0	
2016	Rosedale-Moore Park	2450	4990	0	
2016	St.Andrew-Windfields	3245	1745	0	
2016	Casa Loma	875	2680	0	
2006	Bridle Path-Sunnybrook-York Mills	2205	145	0	
2011	Forest Hill South	1730	1825	0	
2016	Bedford Park-Nortown	4820	1995	0	

In [26]:

```
# Plotting the data from the top 10 expensive neighbourhoods
top_10_most_expensive.hvplot.bar(
    x="neighbourhood",
    xlabel = "Year",
    y="average_house_value",
    ylabel = "Avg. House Value",
    title = "Top 10 Most Expensive Neighborhoods in Toronto",
    height=500,
    rot=46)
```

Out[26]:

In [27]:

```
## Cost Analysis

#In this section, you will use Plotly express to a couple of plots that investors c
an interactively filter and explore various factors related to the house value of t
he Toronto's #neighbourhoods.

### Create a bar chart row facet to plot the average house values for all Toronto's
neighbourhoods per year
```

In [28]:

```
to_data.reset_index(level=0, inplace=True)
fig = px.bar(
    to_data,
    x="neighbourhood",
    y=['single_detached_house',
       'apartment_five_storeys_plus',
       'movable_dwelling',
       'semi_detached_house',
       'row_house',
       'duplex',
       'apartment_five_storeys_less',
       'other_house'
      ],

    color="average_house_value",
    facet_row = 'year',
)
fig.show()
```

Create a sunburst chart to conduct a costs analysis of most expensive neighbourhoods in Toronto per year

In [29]:

```
# Fetch the data from all expensive neighbourhoods per year.
most_expensive_homes = to_data.drop([
    'single_detached_house',
    'apartment_five_storeys_plus',
    'movable_dwelling',
    'semi_detached_house',
    'row_house', 'duplex',
    'apartment_five_storeys_less',
    'other_house',
    'average_house_value'
], axis = 1)
```

In [30]:

```
# Create the sunburst chart
fig = px.sunburst(most_expensive_homes,path=['year','neighbourhood'],
                  values='shelter_costs_owned',
                  color = 'shelter_costs_owned',
                  hover_data=['shelter_costs_owned', 'shelter_costs_rented'],
                  color_continuous_scale='bupu',
                  )

fig.show()
```

Neighbourhood Map

In this section, you will read in neighbourhoods location data and build an interactive map with the average house value per neighbourhood. Use a `scatter_mapbox` from Plotly express to create the visualization. Remember, you will need your Mapbox API key for this.

Load Location Data

In [31]:

```
# Load neighbourhoods coordinates data
file_path = Path("Data/toronto_neighbourhoods_coordinates.csv")
df_neighbourhood_locations = pd.read_csv(file_path)
df_neighbourhood_locations.head()
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-31-a070331206ae> in <module>
      1 # Load neighbourhoods coordinates data
      2 file_path = Path("Data/toronto_neighbourhoods_coordinates.csv")
----> 3 df_neighbourhood_locations = pd.read_csv(file_path)
      4 df_neighbourhood_locations.head()

~/opt/anaconda3/envs/dev/lib/python3.7/site-packages/pandas/io/parsers.py in parser_f(filepath_or_buffer, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, iterator, chunksize, compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, dialect, tupleize_cols, error_bad_lines, warn_bad_lines, delim_whitespace, low_memory, memory_map, float_precision)
    700                     skip_blank_lines=skip_blank_lines)
    701
--> 702         return _read(filepath_or_buffer, kwds)
    703
    704     parser_f.__name__ = name

~/opt/anaconda3/envs/dev/lib/python3.7/site-packages/pandas/io/parsers.py in _read(filepath_or_buffer, kwds)
    427
    428     # Create the parser.
--> 429     parser = TextFileReader(filepath_or_buffer, **kwds)
    430
    431     if chunksize or iterator:

~/opt/anaconda3/envs/dev/lib/python3.7/site-packages/pandas/io/parsers.py in __init__(self, f, engine, **kwds)
    893             self.options['has_index_names'] = kwds['has_index_names']
    894
--> 895         self._make_engine(self.engine)
    896
    897     def close(self):

~/opt/anaconda3/envs/dev/lib/python3.7/site-packages/pandas/io/parsers.py in _make_engine(self, engine)
   1120     def _make_engine(self, engine='c'):
   1121         if engine == 'c':
-> 1122             self._engine = CParserWrapper(self.f, **self.options)
   1123         else:
   1124             if engine == 'python':

~/opt/anaconda3/envs/dev/lib/python3.7/site-packages/pandas/io/parsers.py in __init__(self, src, **kwds)
   1851         kwds['usecols'] = self.usecols
   1852
-> 1853         self._reader = parsers.TextReader(src, **kwds)
```

```
   1854         self.unnamed_cols = self._reader.unnamed_cols
   1855

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.__cinit__()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._setup_pars
er_source()

FileNotFoundError: [Errno 2] File b'Data/toronto_neighbourhoods_coordin
ates.csv' does not exist: b'Data/toronto_neighbourhoods_coordinates.cs
v'
```

Data Preparation

You will need to join the location data with the mean values per neighbourhood.

1. Calculate the mean values for each neighbourhood.
2. Join the average values with the neighbourhood locations.

In [ ]:

```
# Calculate the mean values for each neighborhood

df_neigbourhoods = to_data.groupby(by = "neighbourhood").mean()
df_neigbourhoods.reset_index(inplace=True)
df_neigbourhoods.head()
```

In [ ]:

```
# Join the average values with the neighbourhood locations

all_neighbourhoods = pd.merge(df_neighbourhood_locations, df_neigbourhoods, on = 'n
eighbourhood')
all_neighbourhoods = all_neighbourhoods.drop(all_neighbourhoods.iloc[:,3:11], axis
= 1)
#all_neighbourhoods = all_neighbourhoods.drop(all_neighbourhoods.iloc[:,4], axis =
 1)
all_neighbourhoods.head()
```

Mapbox Visualization

Plot the average values per neighbourhood using a Plotly express `scatter_mapbox` visualization.

In [ ]:

```
# Create a scatter mapbox to analyze neighbourhood info
px.set_mapbox_access_token(open(".env").read())
map_scatter = px.scatter_mapbox(all_neighbourhoods,
    lat='lat',
    lon='lon',
    size='average_house_value',
    color='average_house_value',
    color_continuous_scale = px.colors.cyclical.IceFire,
    size_max=15,
    zoom=11,
    hover_name='neighbourhood',
    title='Averange Value Per Neighbourhood In Toronto',
)
map_scatter.show()
```

In [ ]:

```

```