Scale

In [1]:

```
import pandas as pd
df = pd.DataFrame(['D', 'C-', 'C+', 'B-', 'B', 'B+', 'A-', 'A', 'A+'],
                  index = ['poor', 'ok', 'ok', 'good', 'good', 'good', 'excellent', 'excellent', 'exc
df.rename(columns = {0 : 'Grades'}, inplace = True)
df
```

Out[1]:

	Grades
poor	D
ok	C-
ok	C+
good	B-
good	B
good	B+
excellent	A-
excellent	A
excellent	A+

In [4]:

```
from pandas.api.types import CategoricalDtype

#Nominal Scale
grades = df['Grades'].astype(CategoricalDtype(categories = ['D', 'C-', 'C+', 'B-', 'B', 'B+', 'A-',
grades
```

Out[4]:

```
poor          D
ok           C-
ok           C+
good         B-
good          B
good         B+
excellent    A-
excellent     A
excellent    A+
Name: Grades, dtype: category
Categories (9, object): ['D', 'C-', 'C+', 'B-', ..., 'B+', 'A-', 'A', 'A+']
```

In [6]:

```
#Ordered Scale
grades = df['Grades'].astype(CategoricalDtype(categories = ['D', 'C-', 'C+', 'B-', 'B', 'B+', 'A-',
grades
```

Out[6]:

```
poor          D
ok           C-
ok           C+
good         B-
good          B
good         B+
excellent    A-
excellent     A
excellent    A+
Name: Grades, dtype: category
Categories (9, object): ['D' < 'C-' < 'C+' < 'B-' ... 'B+' < 'A-' < 'A' < 'A+']
```

In [7]:

```
grades > 'B'
```

Out[7]:

```
poor         False
ok           False
ok           False
good         False
good         False
good          True
excellent     True
excellent     True
excellent     True
Name: Grades, dtype: bool
```

In [10]:

```
#Categorization of ration scaled data
s = pd.Series([168, 180, 174, 190, 185, 179, 181, 170, 175, 169, 182, 177, 180, 171])
s.head()
```

Out[10]:

```
0    168
1    180
2    174
3    190
4    185
dtype: int64
```

In [11]:

```
pd.cut(s, 3)
```

Out[11]:

```
0     (167.978, 175.333]
1     (175.333, 182.667]
2     (167.978, 175.333]
3       (182.667, 190.0]
4       (182.667, 190.0]
5     (175.333, 182.667]
6     (175.333, 182.667]
7     (167.978, 175.333]
8     (167.978, 175.333]
9     (167.978, 175.333]
10    (175.333, 182.667]
11    (175.333, 182.667]
12    (175.333, 182.667]
13    (167.978, 175.333]
dtype: category
Categories (3, interval[float64]): [(167.978, 175.333] < (175.333, 182.667] < (182.6
67, 190.0]]
```

In [12]:

```
pd.cut(s, 3, labels = ['Small', 'Medium', 'Large'])
```

Out[12]:

```
0      Small
1     Medium
2      Small
3      Large
4      Large
5     Medium
6     Medium
7      Small
8      Small
9      Small
10    Medium
11    Medium
12    Medium
13     Small
dtype: category
Categories (3, object): ['Small' < 'Medium' < 'Large']
```

Basic Plot with matplotlib

In [13]:

```
%matplotlib notebook
import matplotlib as mpl
mpl.get_backend()
```

Out[13]:

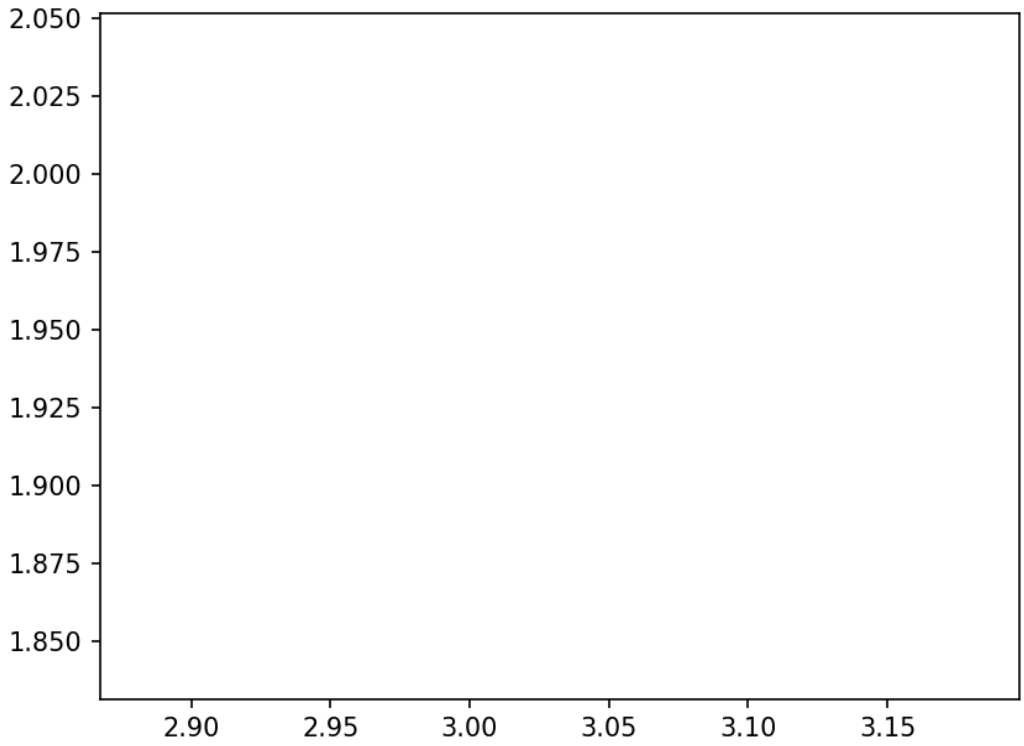
```
'nbAgg'
```

In [14]:

```
import matplotlib.pyplot as plt
plt.plot?
```

In [15]:

```
plt.plot(3, 2)
```

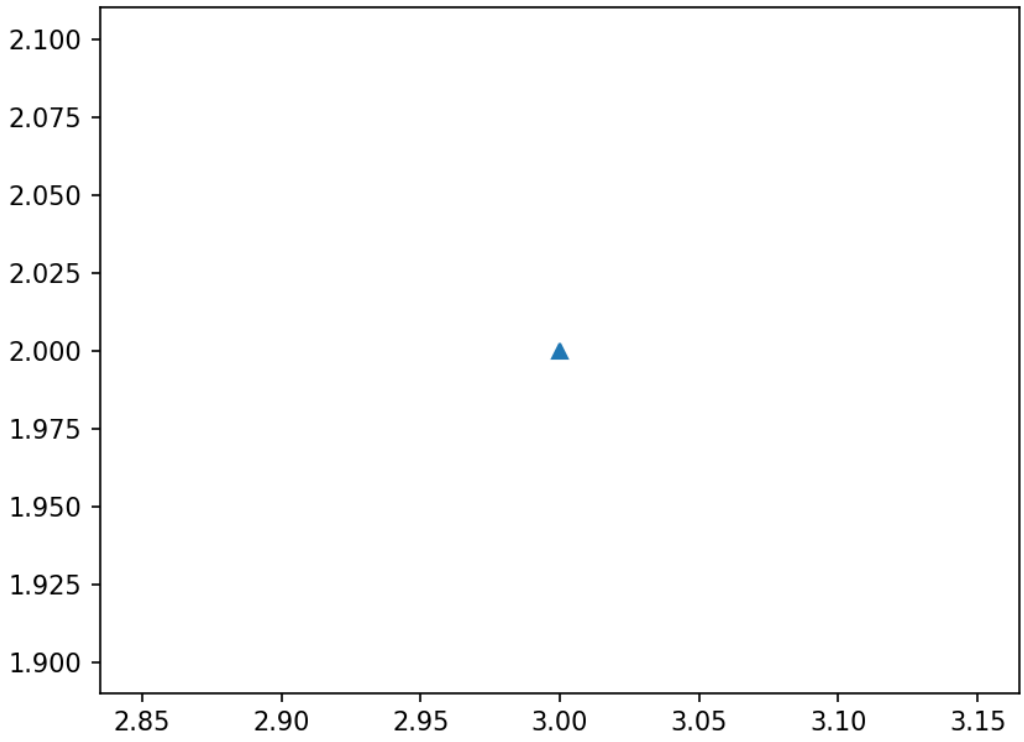


Out[15]:

```
[<matplotlib.lines.Line2D at 0x160308e6190>]
```

In [19]:

```
plt.plot(3, 2, '^')
```

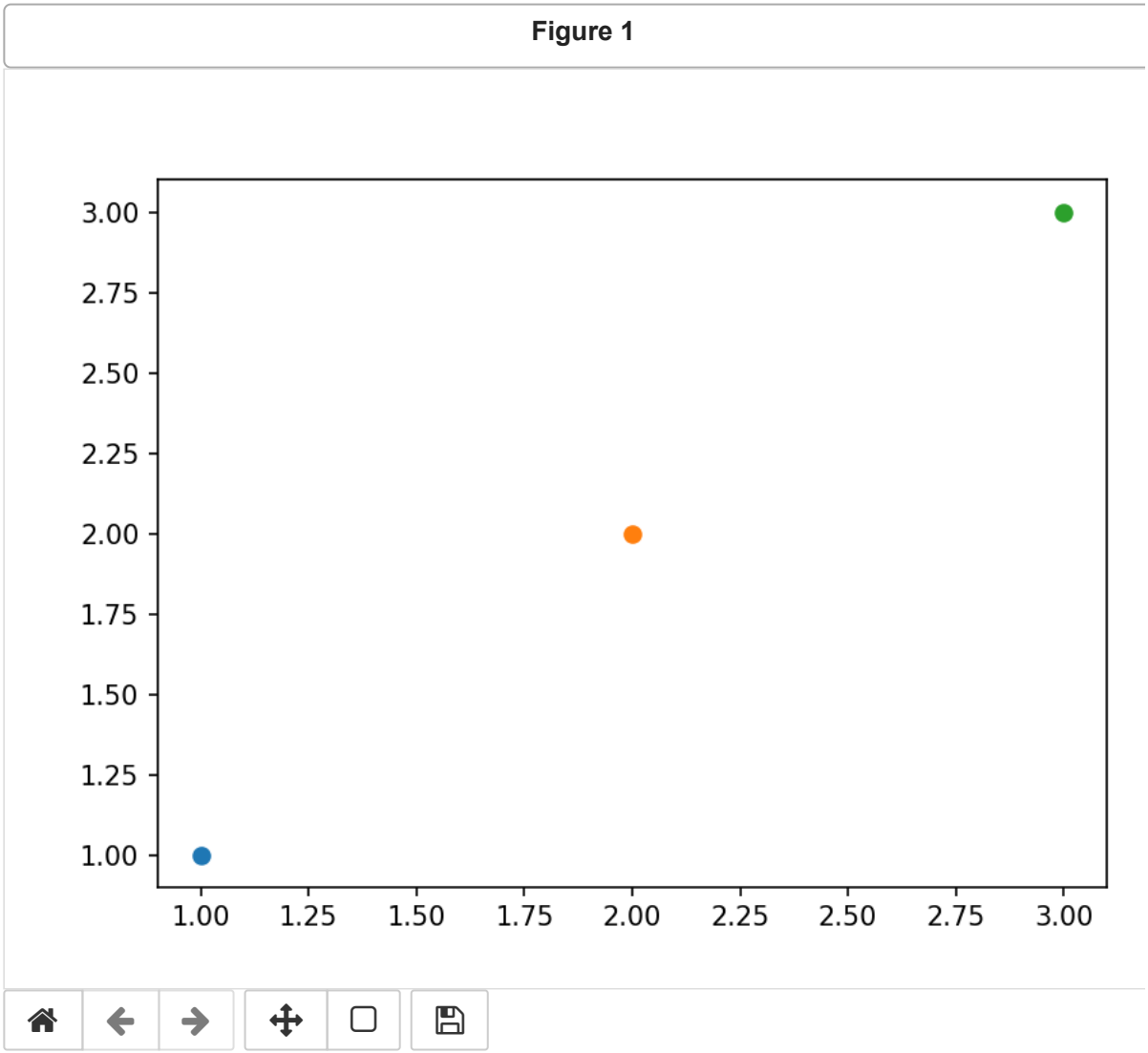


Out[19]:

```
[<matplotlib.lines.Line2D at 0x16031281f10>]
```

In [24]:

```
plt.figure()
plt.plot(1, 1, 'o')
plt.plot(2, 2, 'o')
plt.plot(3, 3, 'o')
```



Out[24]:

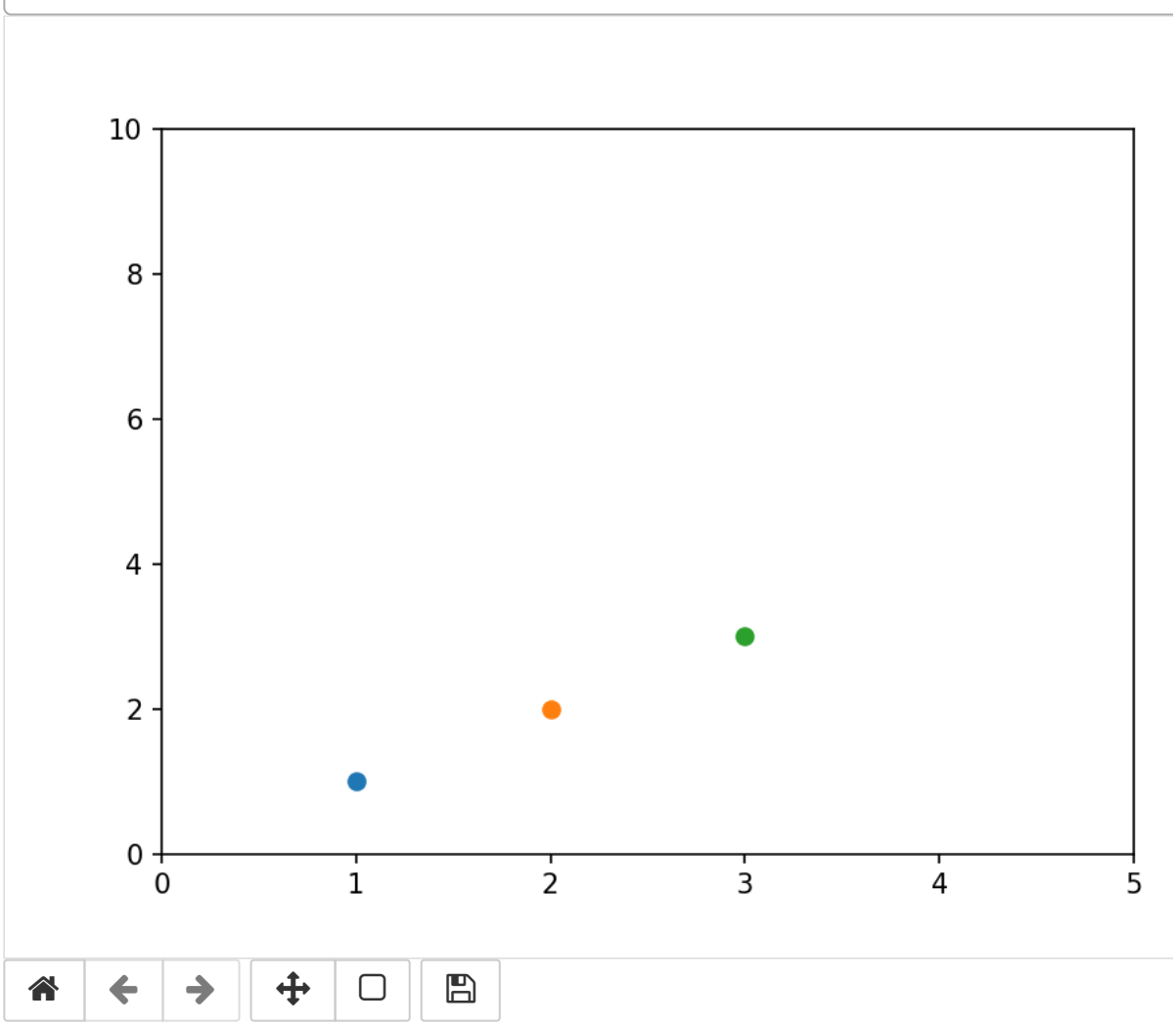
```
[<matplotlib.lines.Line2D at 0x160355b89a0>]
```

In [25]:

```
plt.figure()
plt.plot(1, 1, 'o')
plt.plot(2, 2, 'o')
plt.plot(3, 3, 'o')

#스케일(축) 바꾸는 법
ax = plt.gca()
ax.axis([0, 5, 0 ,10])
```

Figure 2



Out[25]:

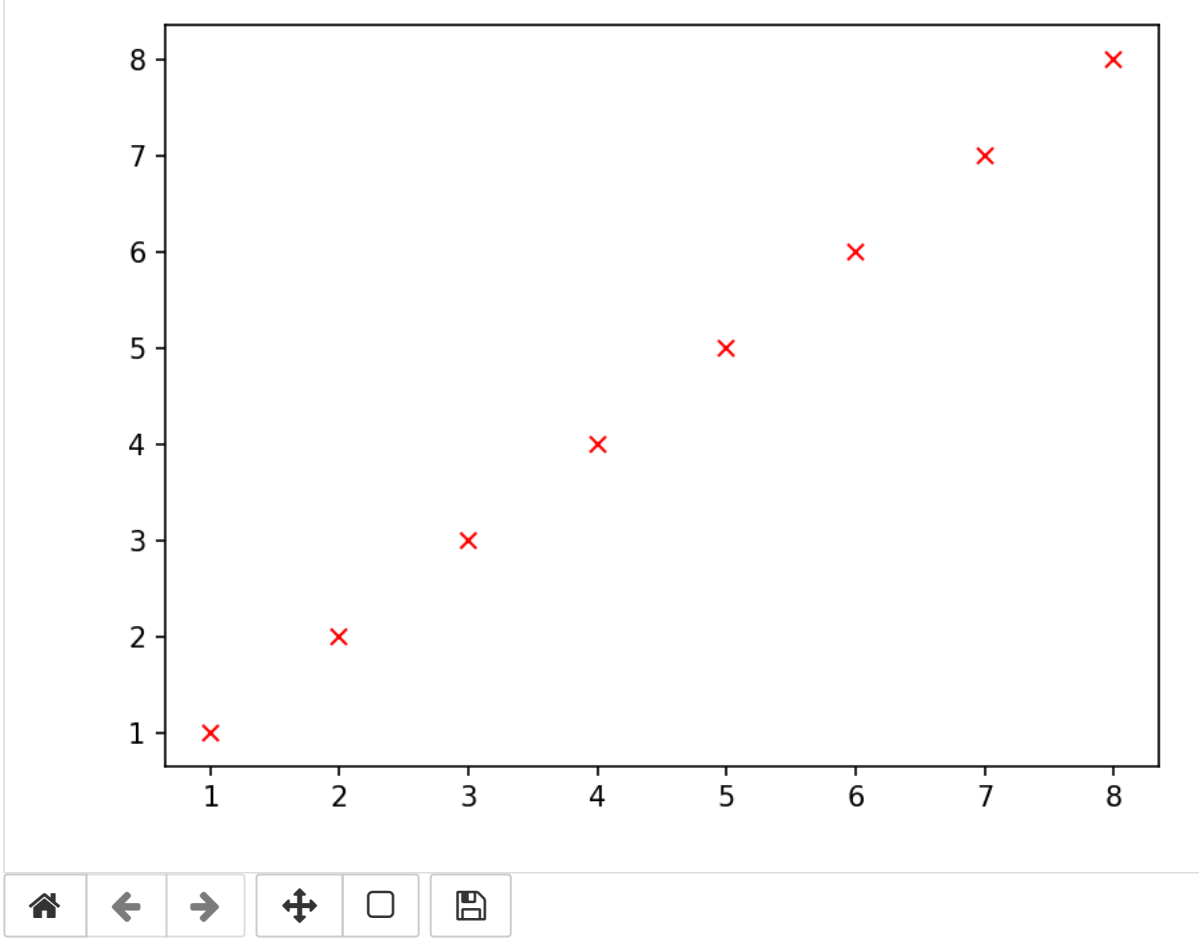
```
(0.0, 5.0, 0.0, 10.0)
```

In [30]:

```
import numpy as np
x = np.array([1, 2, 3, 4, 5, 6, 7, 8])
y = x

plt.figure()
plt.plot(x, y, 'xr')
```

Figure 3



Out[30]:

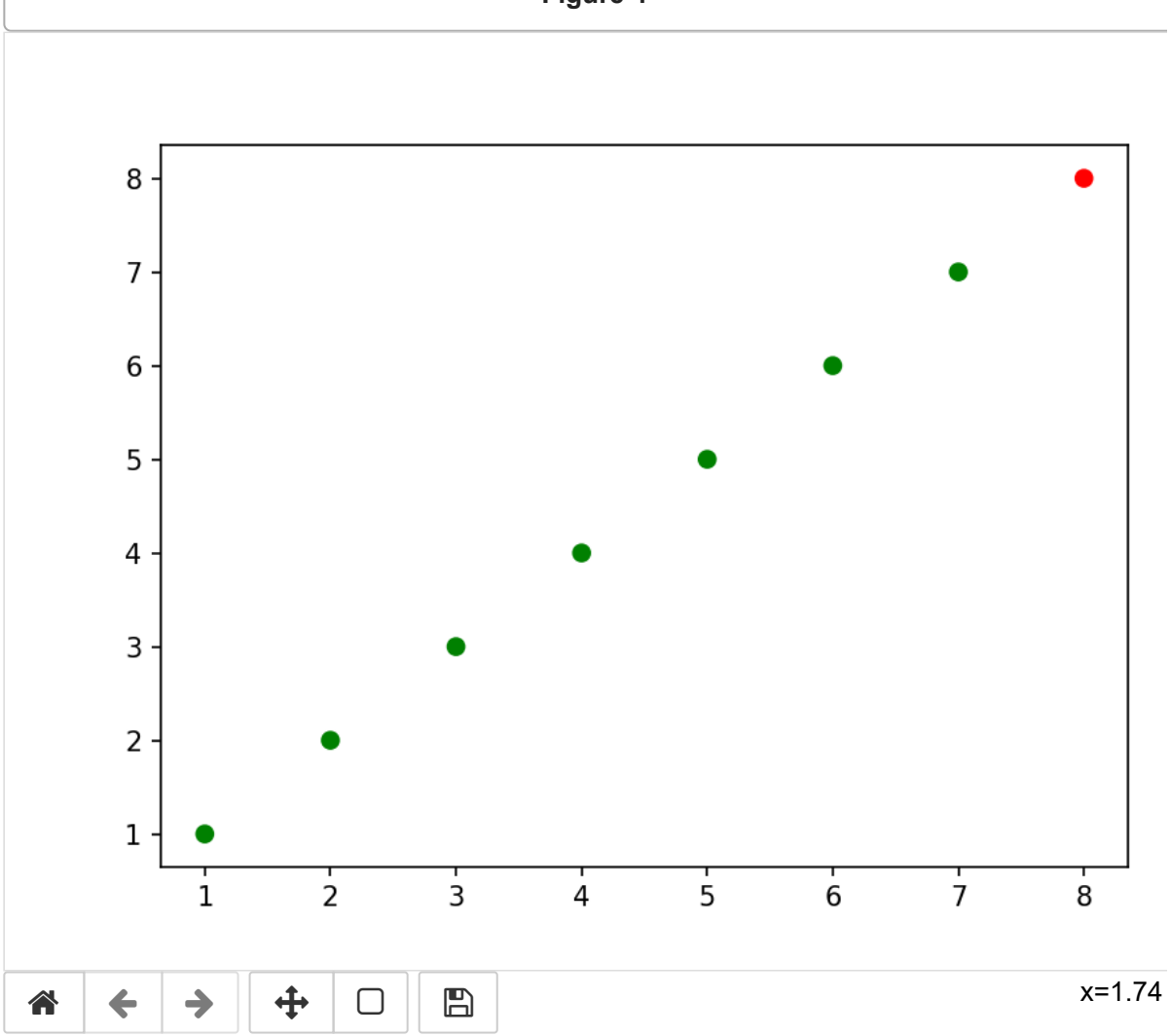
```
[<matplotlib.lines.Line2D at 0x16034a7a520>]
```

In [31]:

```
colors = ['green'] * (len(x) - 1)
colors.append('red')

plt.figure()
plt.scatter(x, y, c = colors)
```

Figure 4



Out[31]:

```
<matplotlib.collections.PathCollection at 0x16036db1280>
```

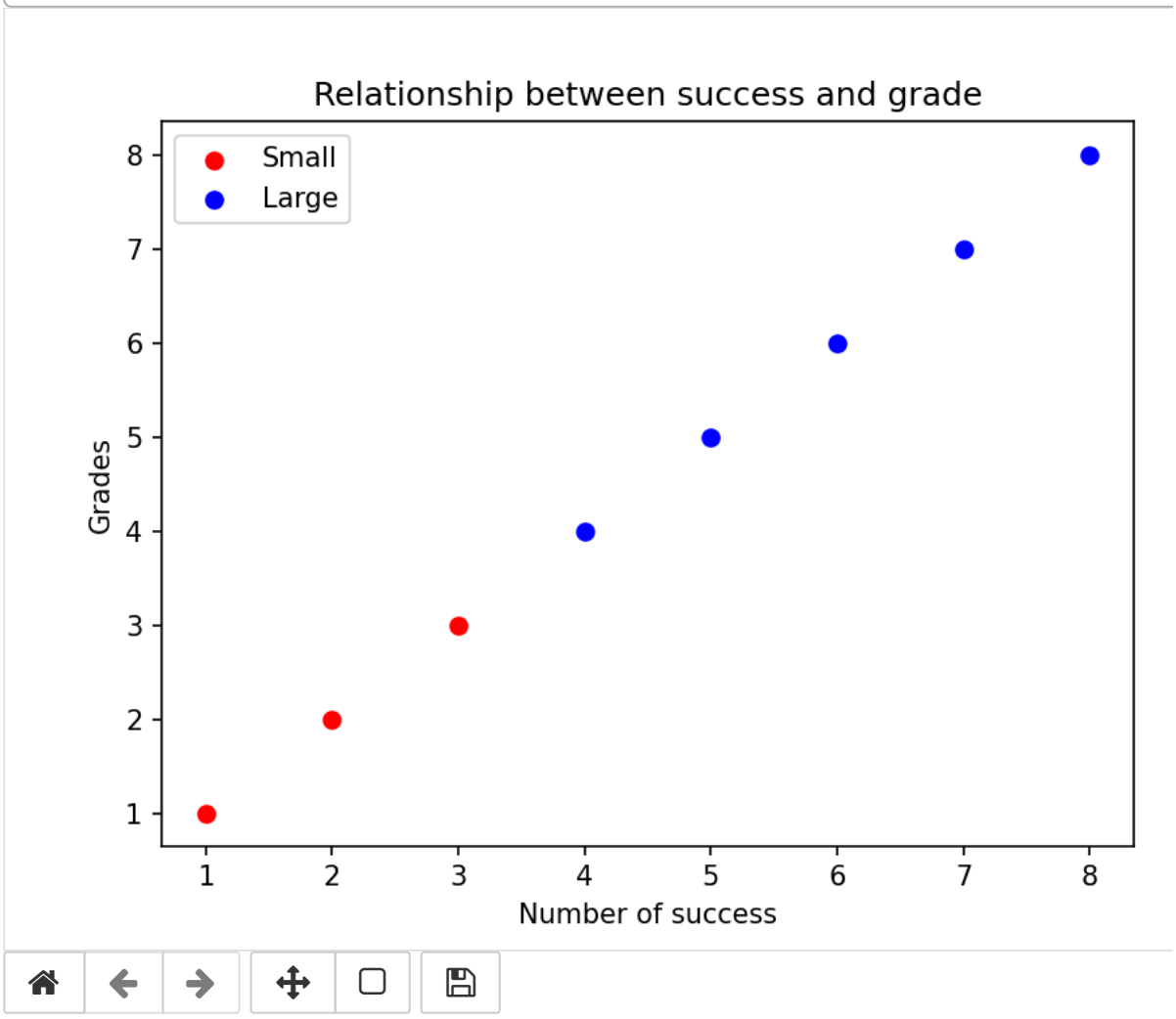
In [39]:

```
plt.figure()
plt.scatter(x[:3], y[:3], c = 'red', label = 'Small')
plt.scatter(x[3:], y[3:], c = 'blue', label = 'Large')

plt.ylabel('Grades')
plt.xlabel('Number of success')
plt.title('Relationship between success and grade')

#왼쪽 위에 표시되는 것
plt.legend()
```

Figure 5



Out[39]:

```
<matplotlib.legend.Legend at 0x1603983f100>
```

Line Plots

In [40]:

```
linear_data = x
linear_data
```

Out[40]:

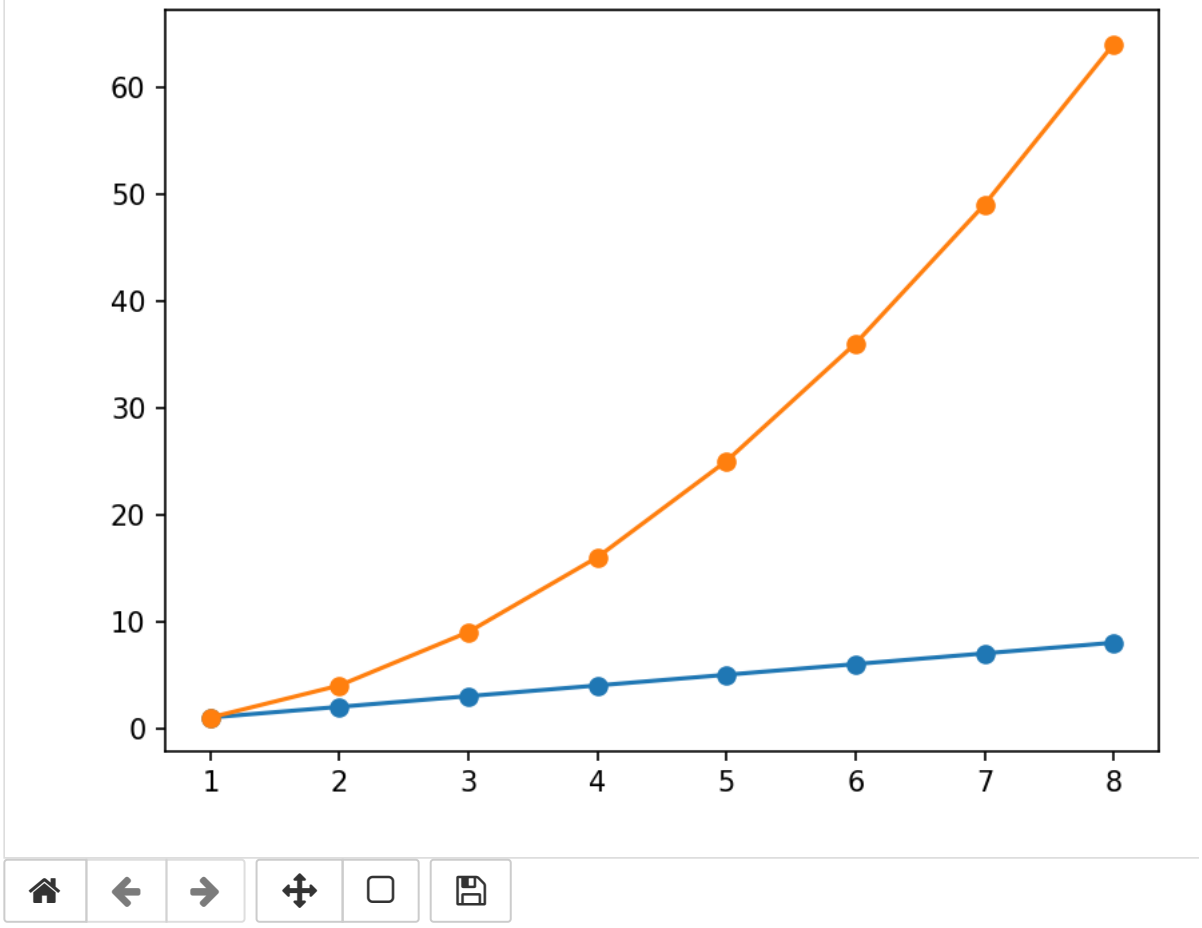
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

In [63]:

```
exponential_data = linear_data**2

plt.figure()
plt.plot(x, linear_data, '-o', x, exponential_data, '-o')
```

Figure 14



Out[63]:

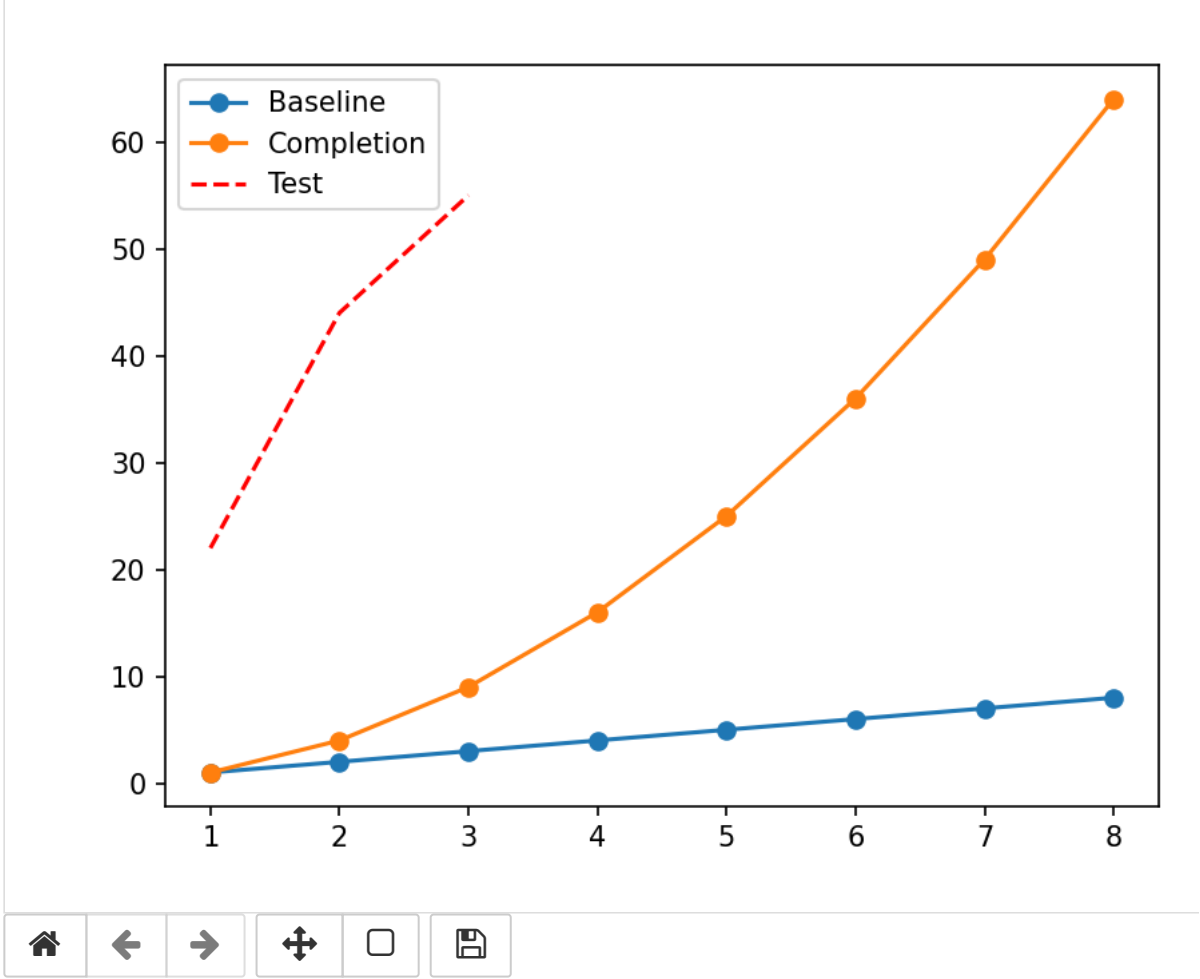
```
[<matplotlib.lines.Line2D at 0x1603e286670>,
 <matplotlib.lines.Line2D at 0x1603e2866a0>]
```

In [61]:

```
plt.figure()
plt.plot(x, linear_data, '-o', x, exponential_data, '-o')
plt.plot(x[:3], [22, 44, 55], '--r')

plt.legend(['Baseline', 'Completion', 'Test'])
```

Figure 13



Out[61]:

```
<matplotlib.legend.Legend at 0x1603e182640>
```

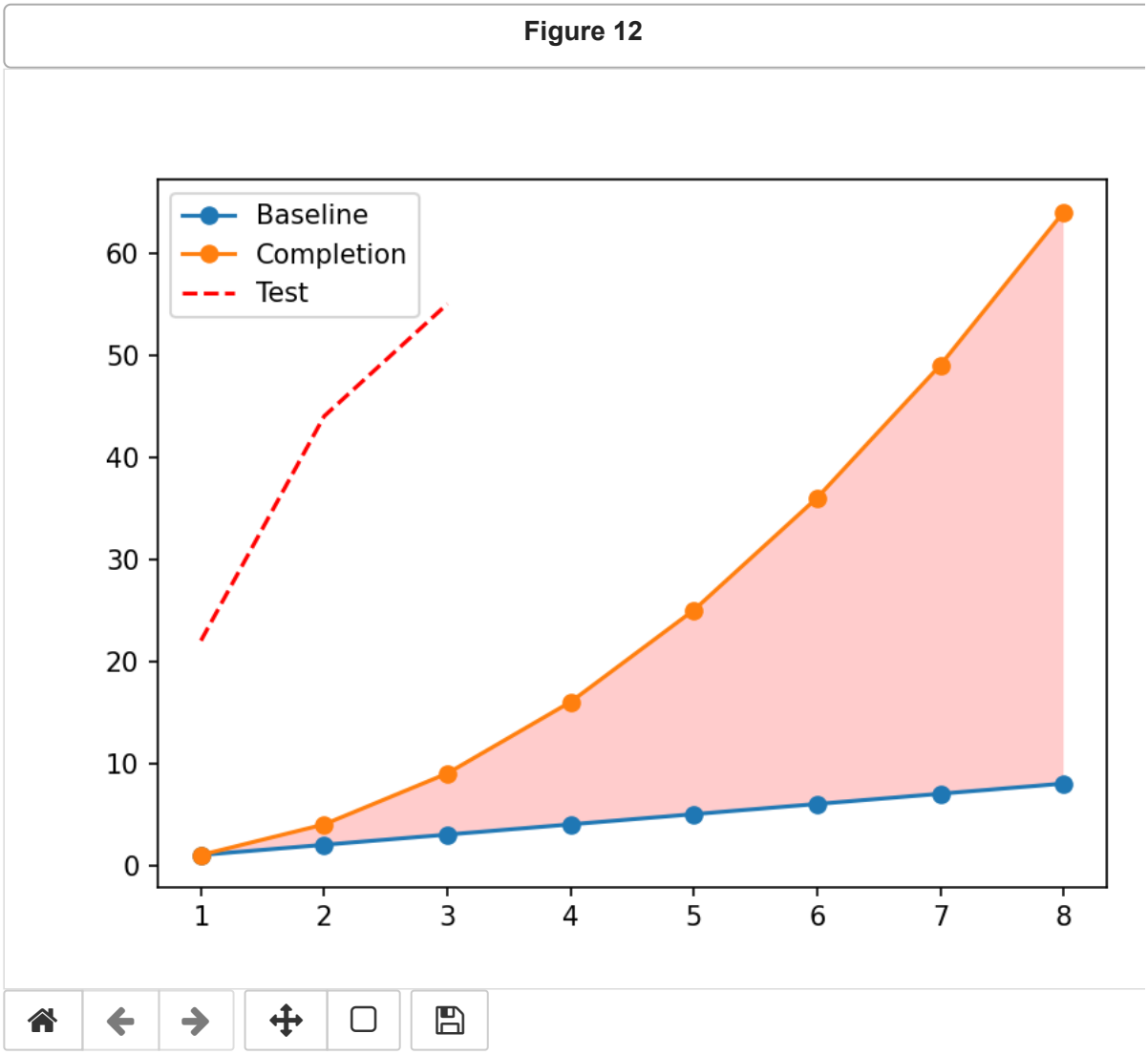
In [60]:

```
plt.figure()
plt.plot(x, linear_data, '-o', x, exponential_data, '-o')
plt.plot(x[:3], [22, 44, 55], '--r')

plt.legend(['Baseline', 'Completion', 'Test'])

plt.fill_between(x, linear_data, exponential_data, facecolor = 'red', alpha = 0.2)
```

Figure 12



Out[60]:

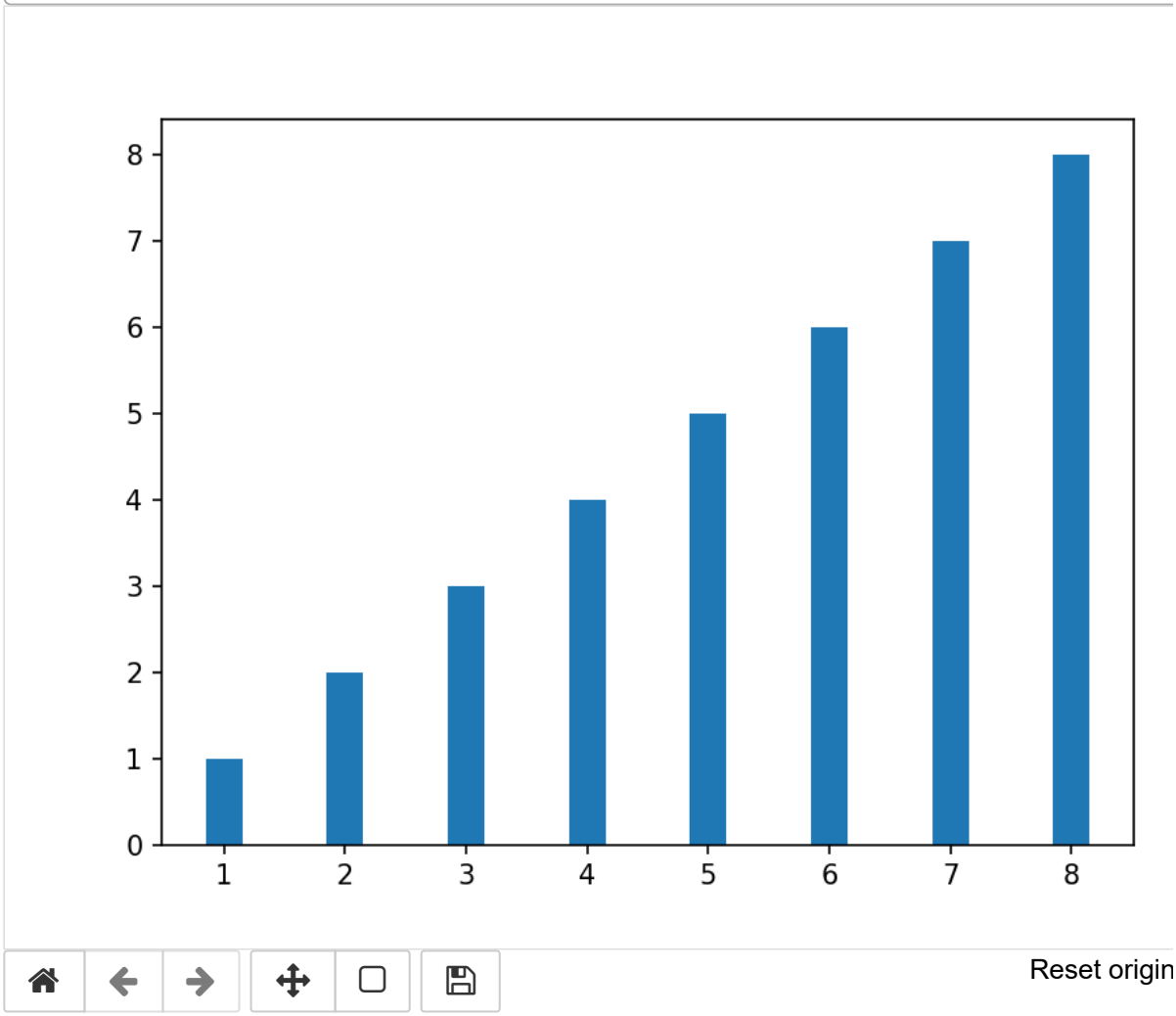
```
<matplotlib.collections.PolyCollection at 0x16042d156d0>
```

Bar Charts

In [55]:

```
plt.figure()
plt.bar(x, linear_data, width = 0.3)
```

Figure 9



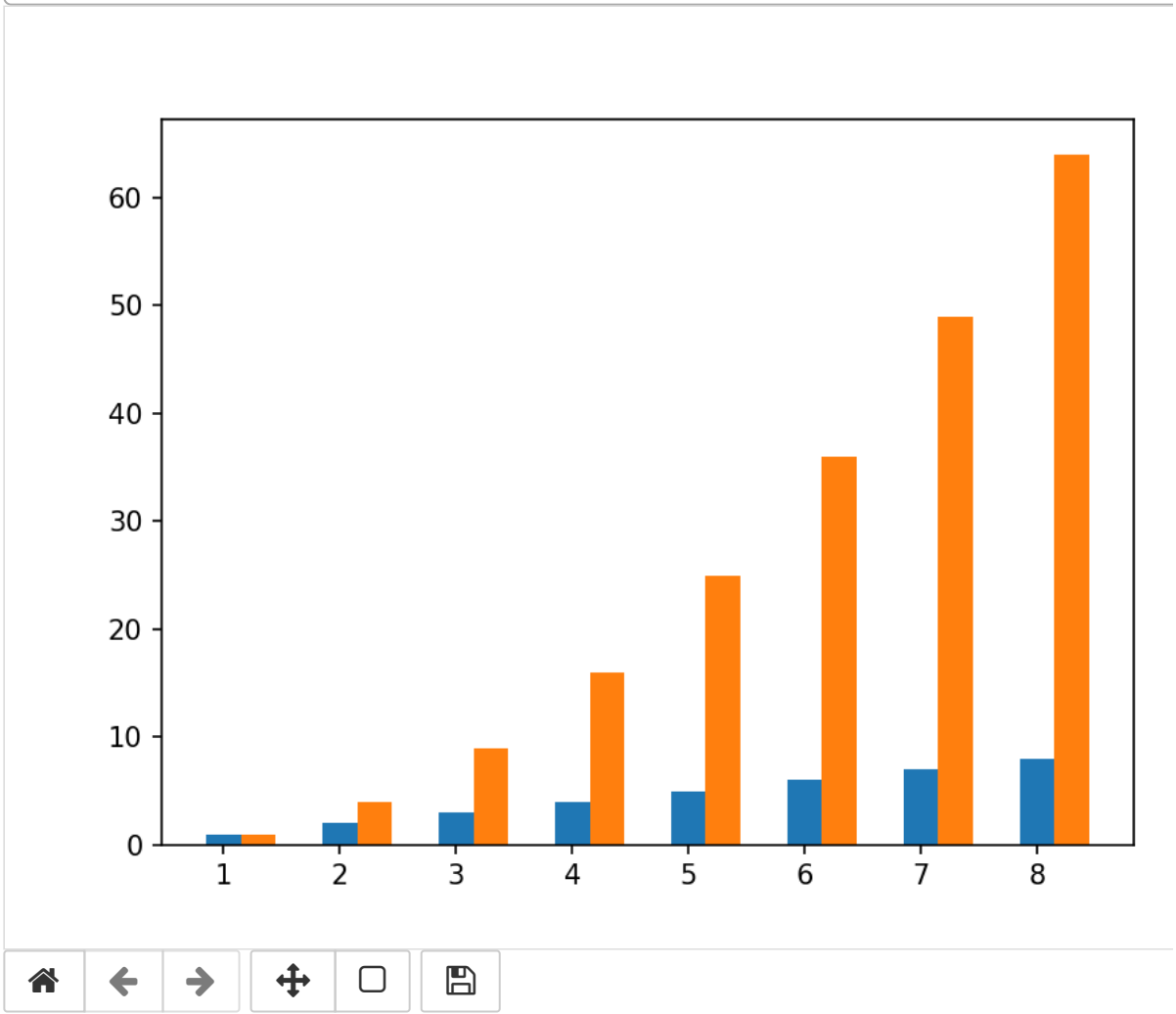
Out[55]:

```
<BarContainer object of 8 artists>
```

In [56]:

```
plt.figure()
plt.bar(x, linear_data, width = 0.3)
plt.bar(x+0.3, exponential_data, width = 0.3)
```

Figure 10



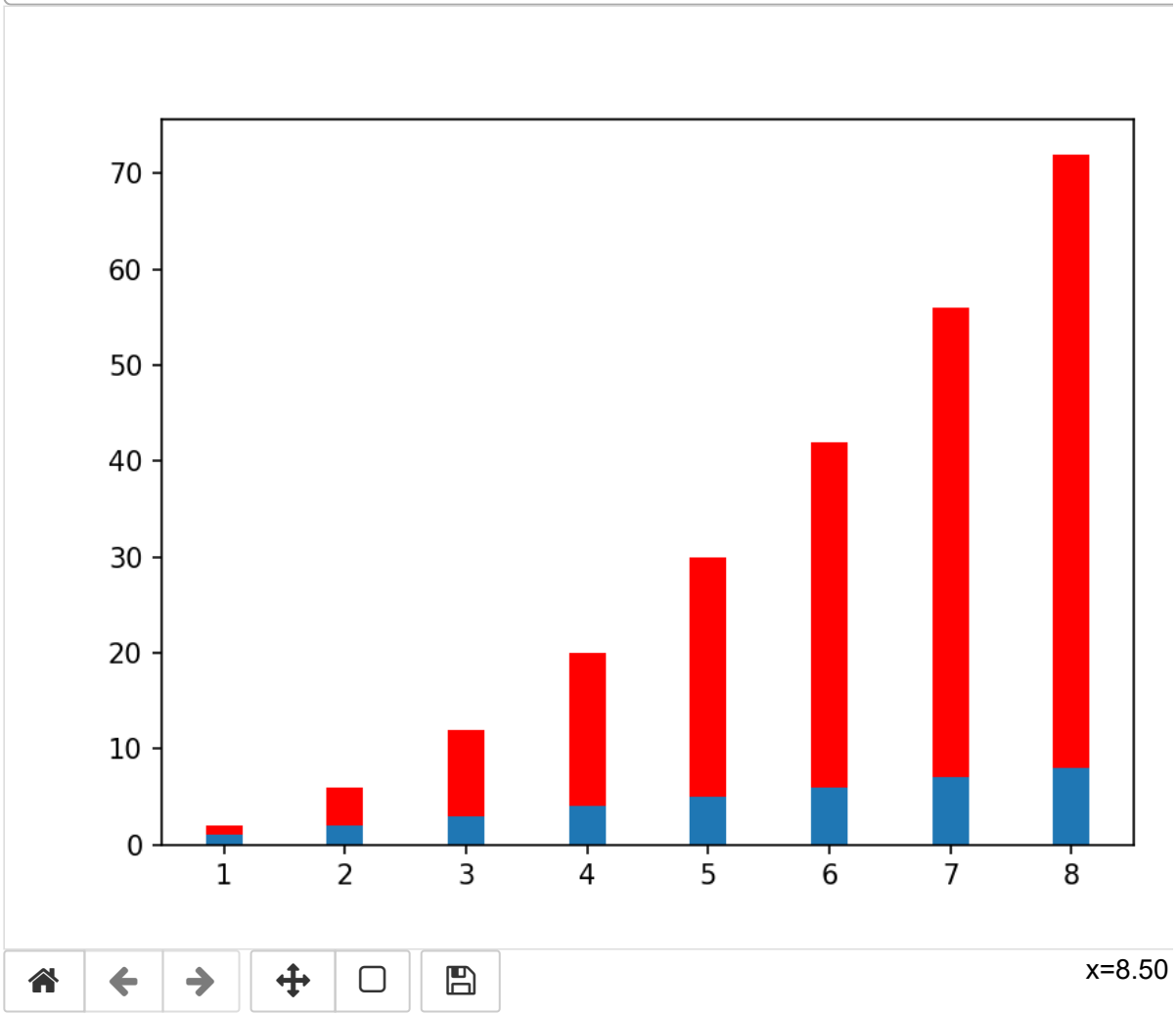
Out[56]:

```
<BarContainer object of 8 artists>
```

In [59]:

```
plt.figure()
plt.bar(x, linear_data, width = 0.3)
plt.bar(x, exponential_data, bottom = linear_data, width = 0.3, color = 'red')
```

Figure 11



Out[59]:

<BarContainer object of 8 artists>