```
The house price data of King County was collected from kaggle.
This dataset contains house sale prices for King County, which includes Seattle. It includes homes sold between May 2014 and May 2015.
It includes  'id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
          'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade',
          'sqft_above', 'sqft_basement', 'yr_built', 'yr_renovated', 'zipcode',
          'lat', 'long', 'sqft_living15', 'sqft_lot15' and other data columns.
Combined with the python analysis tool learned recently, the possible factors affecting house prices were analyzed.
For example, what is the relationship between house price and house size?
What is the relationship between house price and housing rating?
......
Taking price as dependent variable Y,  sqft_living and grade as independent Xs, I will try to find their direct correlation through data analysis.

```

In [61]:

```
#read dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
house_price=pd.read_csv('kc_house_data.csv')

#Show all data columns
house_price.columns


```

Out[61]:

```
Index(['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade',
       'sqft_above', 'sqft_basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'sqft_living15', 'sqft_lot15'],
      dtype='object')
```

In [31]:

```
#select the price, grade and condition
price_grade_size = house_price.loc[house_price.index[:], ['price', 'sqft_living','grade']]
price_grade_size
```

Out[31]:

	price	sqft_living	grade
0	221900.0	1180	7
1	538000.0	2570	7
2	180000.0	770	6
3	604000.0	1960	7
4	510000.0	1680	8
...	...	...	...
21608	360000.0	1530	8
21609	400000.0	2310	8
21610	402101.0	1020	7
21611	400000.0	1600	8
21612	325000.0	1020	7

In [39]:

```
#Judge whether there is missing data
price_grade_size.isnull().any(axis=0)
```

Out[39]:

```
price          False
sqft_living    False
grade          False
dtype: bool
```

In [33]:

```
#Descriptive statistics of the new table
price_grade_size.describe()
```

Out[33]:

	price	sqft_living	grade
count	2.161300e+04	21613.000000	21613.000000
mean	5.400881e+05	2079.899736	7.656873
std	3.671272e+05	918.440897	1.175459
min	7.500000e+04	290.000000	1.000000
25%	3.219500e+05	1427.000000	7.000000
50%	4.500000e+05	1910.000000	7.000000
75%	6.450000e+05	2550.000000	8.000000
max	7.700000e+06	13540.000000	13.000000

In [51]:

```
#Perform group by descriptive statistics on the new table
price_grade_size.groupby('grade').describe()
```

Out[51]:

	price							
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
grade								
1	1.0	1.420000e+05	NaN	142000.0	142000.0	142000.0	142000.0	14200(
3	3.0	2.056667e+05	1.135180e+05	75000.0	168500.0	262000.0	271000.0	28000(
4	29.0	2.143810e+05	9.430617e+04	80000.0	145000.0	205000.0	265000.0	43500(
5	242.0	2.485240e+05	1.181003e+05	78000.0	175000.0	228700.0	295750.0	79500(
6	2038.0	3.019196e+05	1.229703e+05	82000.0	215037.5	275276.5	366837.5	120000(
7	8981.0	4.025903e+05	1.558769e+05	90000.0	285000.0	375000.0	485000.0	205000(
8	6068.0	5.428528e+05	2.174734e+05	140000.0	390000.0	510000.0	640000.0	307000(
9	2615.0	7.735132e+05	3.161201e+05	230000.0	571500.0	720000.0	880000.0	270000(
10	1134.0	1.071771e+06	4.835451e+05	316000.0	768087.5	914327.0	1250000.0	360000(
11	399.0	1.496842e+06	7.050993e+05	420000.0	1036000.0	1284000.0	1700000.0	706250(
12	90.0	2.191222e+06	1.027819e+06	835000.0	1500000.0	1817500.0	2668500.0	535000(
13	13.0	3.709615e+06	1.859450e+06	1780000.0	2415000.0	2983000.0	3800000.0	770000(

In [52]:

```
#Perform group by descriptive statistics on the new table
price_grade_size.groupby('sqft_living').describe()
```

Out[52]:

	price							
	count	mean	std	min	25%	50%	75%	max
sqft_living								
290	1.0	142000.0	NaN	142000.0	142000.0	142000.0	142000.0	142000
370	1.0	276000.0	NaN	276000.0	276000.0	276000.0	276000.0	276000
380	1.0	245000.0	NaN	245000.0	245000.0	245000.0	245000.0	245000
384	1.0	265000.0	NaN	265000.0	265000.0	265000.0	265000.0	265000
390	2.0	236500.0	12020.81528	228000.0	232250.0	236500.0	240750.0	245000
...	...	...	...	...	...	...	...	
9640	1.0	4668000.0	NaN	4668000.0	4668000.0	4668000.0	4668000.0	4668000
9890	1.0	6885000.0	NaN	6885000.0	6885000.0	6885000.0	6885000.0	6885000
10040	1.0	7062500.0	NaN	7062500.0	7062500.0	7062500.0	7062500.0	7062500
12050	1.0	7700000.0	NaN	7700000.0	7700000.0	7700000.0	7700000.0	7700000
13540	1.0	2280000.0	NaN	2280000.0	2280000.0	2280000.0	2280000.0	2280000

1038 rows × 16 columns

In [60]:

```
#Group statistics on the relationship between price and grade/sqft_living
price_grade_size.groupby('grade').agg({'price':'mean'})
price_grade_size.groupby('sqft_living').agg({'price':'mean'})
```

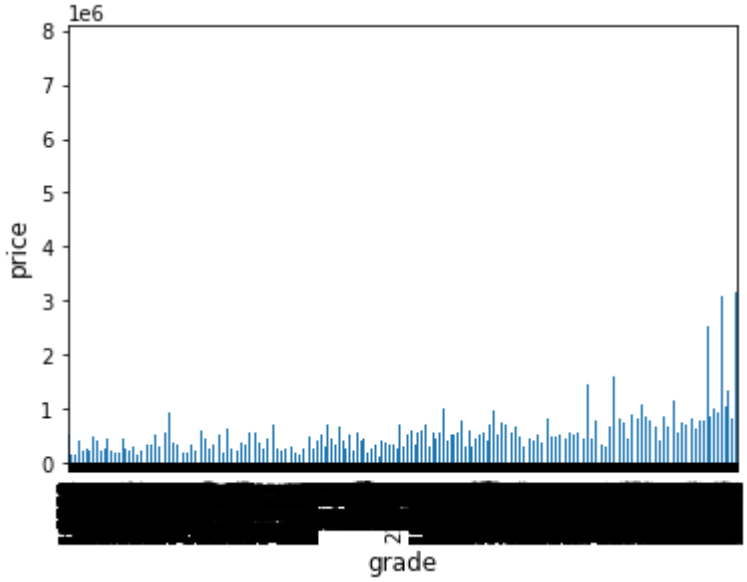
Out[60]:

	price
sqft_living	
290	142000.0
370	276000.0
380	245000.0
384	265000.0
390	236500.0
...	...
9640	4668000.0
9890	6885000.0
10040	7062500.0
12050	7700000.0
13540	2280000.0

1038 rows × 1 columns

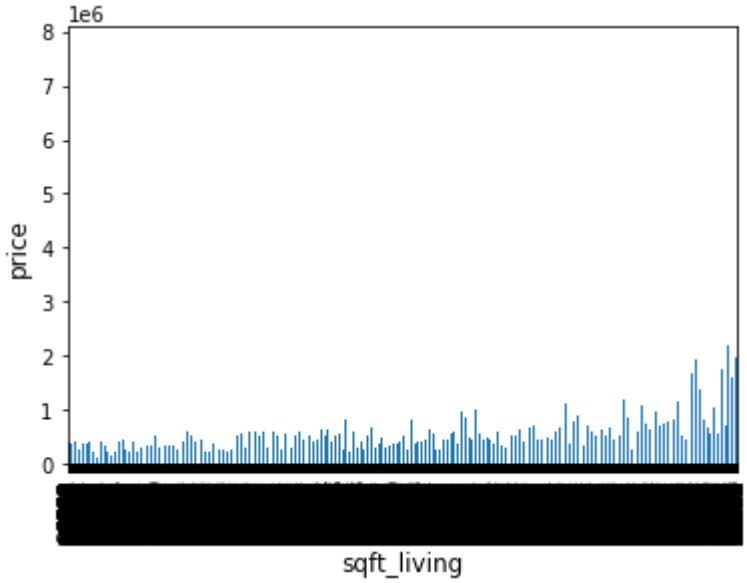
In [69]:

```
sorted_pgc = price_grade_size.sort_values(by='grade', ascending=True)
# sorted_pgc
ax = sorted_pgc['price'].plot(kind='bar')
ax.set_xlabel('grade', fontsize=12)
ax.set_ylabel('price', fontsize=12)
plt.show()
```



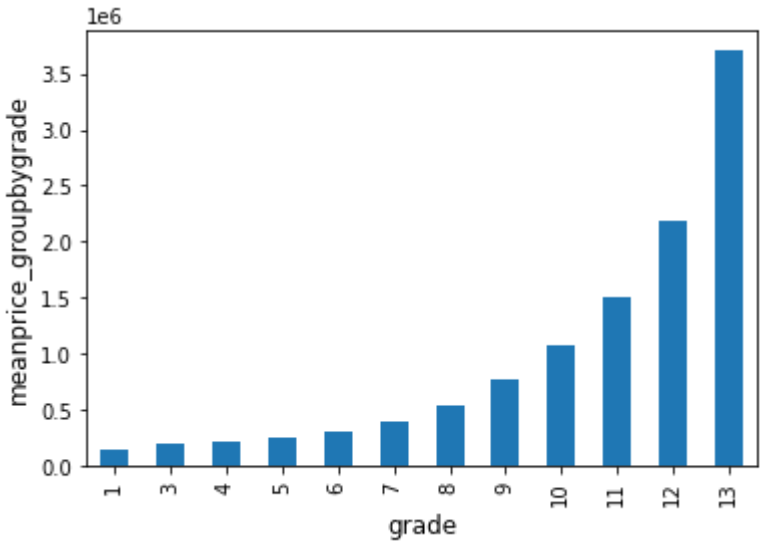
In [70]:

```
sorted_pgc1 = price_grade_size.sort_values(by='sqft_living', ascending=True)
# sorted_pgc1
ax = sorted_pgc1['price'].plot(kind='bar')
ax.set_xlabel('sqft_living', fontsize=12)
ax.set_ylabel('price', fontsize=12)
plt.show()
```



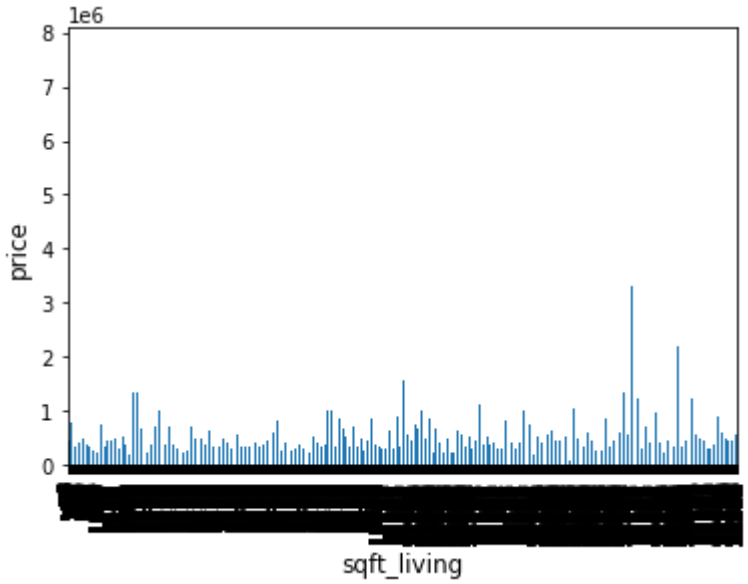
In [71]:

```
meanprice_groupbygrade = price_grade_size.groupby('grade').agg({'price':'mean'})
ax = meanprice_groupbygrade['price'].plot(kind='bar')
ax.set_xlabel('grade', fontsize=12)
ax.set_ylabel('meanprice_groupbygrade', fontsize=12)
plt.show()
```



In [68]:

```
meanprice_groupbylivingspace = price_grade_size.groupby('sqft_living').agg({'price':'mean'})
ax = price_grade_size['price'].plot(kind='bar')
ax.set_xlabel('sqft_living', fontsize=12)
ax.set_ylabel('price', fontsize=12)
plt.show()
```



In [ ]:

```
1
```