Исследование объявлений о продаже квартир

Исследуем данные сервиса Яндекс.Недвижимость — архив объявлений о продаже квартир в Санкт-Петербурге и соседних населённых пунктов за несколько лет. Нужно определить рыночную стоимость объектов недвижимости. Наша задача — установить параметры, которые позволят построить автоматизированную систему: она отследит аномалии и мошенническую деятельность.

По каждой квартире на продажу доступны два вида данных. Первые вписаны пользователем, вторые — получены автоматически на основе картографических данных. Например, расстояние до центра, аэропорта, ближайшего парка и водоёма.

Цель проекта: исследовать параметры объектов недвижимости и выявить зависимости, которые помогут в определении рыночной стоимости жилья

Изучение общей информации

Импортируем необходимые библиотеки и читаем данные, подобрав соответствующий разделитель.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
#  увеличим максимальное количество отображающихся столбцов командой
pd.set_option('display.max_columns', None)
```

In [2]:
```
try: data = pd.read_csv('https://code.s3.yandex.net/datasets/real_estate_data.csv', s
except: data = pd.read_csv(r'C:\\Users\\user\\Downloads\\real_estate_data.csv', sep=
```

Выведем общую информацию о размерах массива данных, названиях столбцов и типах данных в них.

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23699 entries, 0 to 23698
Data columns (total 22 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   total_images          23699 non-null  int64
 1   last_price            23699 non-null  float64
```

```
 2   total_area            23699 non-null  float64
 3   first_day_exposition  23699 non-null  object
 4   rooms                 23699 non-null  int64
 5   ceiling_height        14504 non-null  float64
 6   floors_total          23613 non-null  float64
 7   living_area           21796 non-null  float64
 8   floor                 23699 non-null  int64
 9   is_apartment          2775 non-null   object
 10  studio                23699 non-null  bool
 11  open_plan             23699 non-null  bool
 12  kitchen_area          21421 non-null  float64
 13  balcony               12180 non-null  float64
 14  locality_name         23650 non-null  object
 15  airports_nearest      18157 non-null  float64
 16  cityCenters_nearest   18180 non-null  float64
 17  parks_around3000      18181 non-null  float64
 18  parks_nearest         8079 non-null   float64
 19  ponds_around3000      18181 non-null  float64
 20  ponds_nearest         9110 non-null   float64
 21  days_exposition       20518 non-null  float64
dtypes: bool(2), float64(14), int64(3), object(3)
```

Оценим состав данных по первым строкам массива.

In [4]:
```
data.head(10)
```

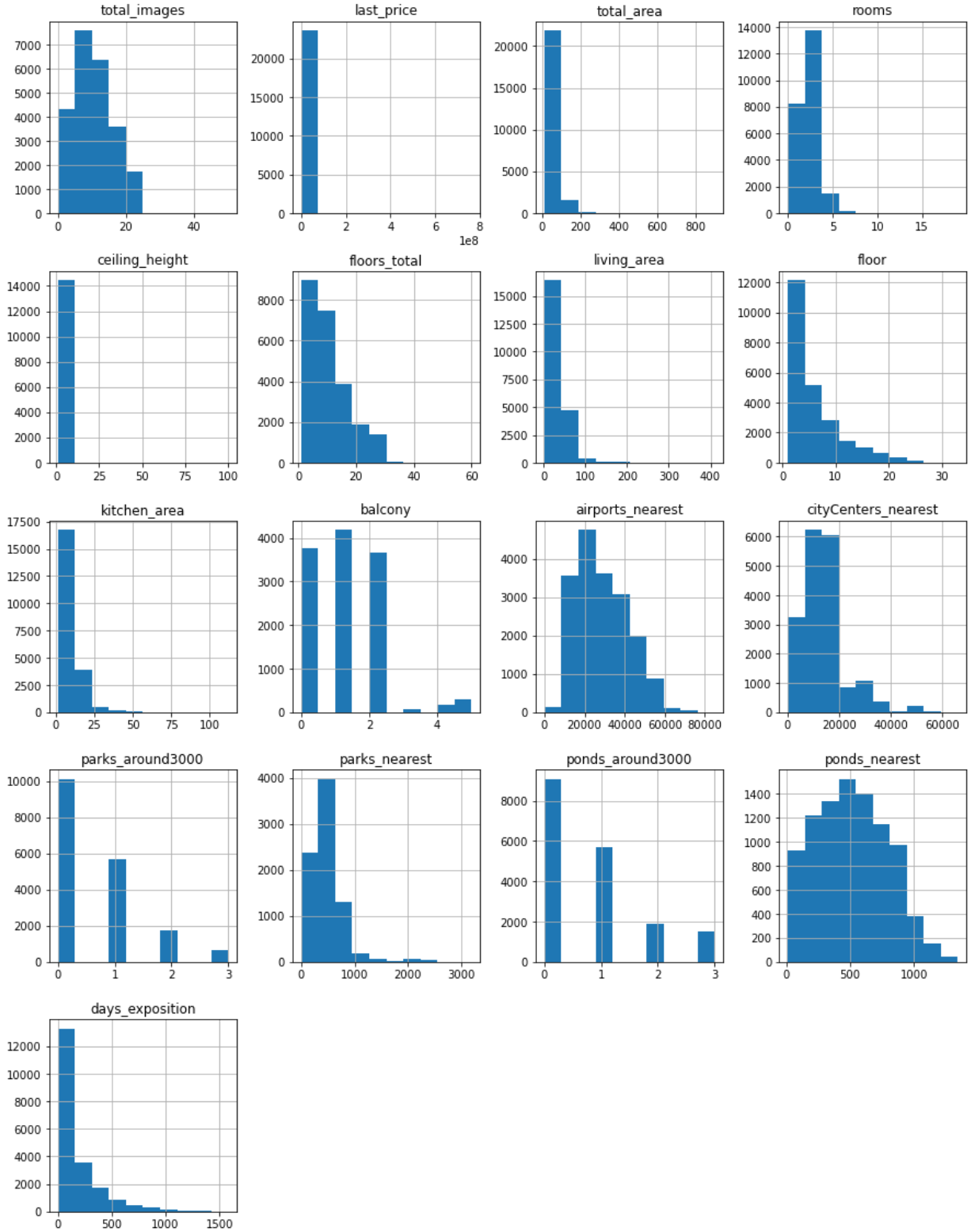
Out[4]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living
0	20	13000000.0	108.00	2019-03-07T00:00:00	3	2.70	16.0	
1	7	3350000.0	40.40	2018-12-04T00:00:00	1	NaN	11.0	
2	10	5196000.0	56.00	2015-08-20T00:00:00	2	NaN	5.0	
3	0	64900000.0	159.00	2015-07-24T00:00:00	3	NaN	14.0	
4	2	10000000.0	100.00	2018-06-19T00:00:00	2	3.03	14.0	
5	10	2890000.0	30.40	2018-09-10T00:00:00	1	NaN	12.0	
6	6	3700000.0	37.30	2017-11-02T00:00:00	1	NaN	26.0	
7	5	7915000.0	71.60	2019-04-18T00:00:00	2	NaN	24.0	
8	20	2900000.0	33.16	2018-05-23T00:00:00	1	NaN	27.0	
9	18	5400000.0	61.00	2017-02-26T00:00:00	3	2.50	9.0	

Построим общую гистограмму для всех столбцов таблицы.

In [5]:

```
data.hist(figsize=(15, 20))
plt.show()
```



Для уточнения информации, полученной в гистограммах, используем метод describe и округлим значения для удобства чтения числовых данных.

In [6]:
```
data.describe().round()
```

Out[6]:

	total_images	last_price	total_area	rooms	ceiling_height	floors_total	living_area	floor
count	23699.0	23699.0	23699.0	23699.0	14504.0	23613.0	21796.0	23699.0
mean	10.0	6541549.0	60.0	2.0	3.0	11.0	34.0	6.0
std	6.0	10887013.0	36.0	1.0	1.0	7.0	22.0	5.0
min	0.0	12190.0	12.0	0.0	1.0	1.0	2.0	1.0
25%	6.0	3400000.0	40.0	1.0	3.0	5.0	19.0	2.0
50%	9.0	4650000.0	52.0	2.0	3.0	9.0	30.0	4.0
75%	14.0	6800000.0	70.0	3.0	3.0	16.0	42.0	8.0
max	50.0	763000000.0	900.0	19.0	100.0	60.0	410.0	33.0

По гистограмме и основным характеристикам значений можно сделать следующие приблизительные выводы:

- в объявлениях встречается от 0 до 50 фото объекта, чаще 9-10;
- цена объектов колеблется в очень больших пределах, неправдоподобно велико различие минимума и максимума, огромное стандартное отклонение, разница среднего и медианы почти на 2 млн. В дальнейшем следует найти и устранить выбросы в данных;
- общая площадь продаваемых объектов чаще всего менее 100 кв.м. (50-60), но есть объекты и площадью 900 кв.м., которые так же помешают в дальнейшей обработке;
- количество комнат от 0 (свободная планировка) до 19 (видимо, это дворец с огромной ценой и площадью, необходимо проверить), чаще всего продают 2-ухкомнатные квартиры;
- высота потолков от 1 до 3 м, что объяснимо - мансарды, полуподвальные помещения - низкий потолок. Современные коттеджи с высокими потолками. Но максимум в 100 м говорит о необходимости проверки данных столбца;
- больше всего квартир продается в 9-11-этажных зданиях, чаще всего в домах от 5 до 16 этажей. Странное значение 60 этажей,помещение в Лахта-центре? Возможно, стоит удалить;
- жилая площадь в среднем около 30 кв.м, чаще колеблется от 20 до 42, минимум в 2 и максимум в 410 кв.м. следует проверить;
- много квартир продают на 4-6 этажах, что логично - зданий с подходящей этажностью в объявлениях большинство;
- площадь кухни невелика, чаще всего 9-11 кв.м., но есть некоторый выбор и среди объектов с площадью кухни от 10 до 20 кв.м.;
- больше всего квартир с 1 балконом, чуть меньше без балконов и с 2-мя, но встречаются и 3-5 балконов в квартире;
- расстояние до аэропорта в среднем в пределах от 18000 до 37000 (видимо, м), там - где

эти данные указаны;
- большинство квартир на расстоянии до 10-16 тыс.м от центра (для тех объектов, где эти данные указаны);
- парков поблизости чаще всего 1, но встречаются объекты с 2-3-мя парками рядом;
- если парки есть, то они в основном на расстоянии до 1000 м;
- прудов тоже чаще до 1, те что есть от 1 до 3-х чаще на расстоянии около 500 м;
- квартиры продаются в срок до 100 дней, в пределах от 45 до 230, есть рекорд в 1580 дней (больше 4 лет!).

Данные нуждаются в более подробном исследовании, изменении типов данных, замене или удалении пропусков.

Предобработка данных

Найдем количество пропущенных значений в столбцах массива данных. И проверим наличие строк-дубликатов.

In [7]:
```
data.isna().sum()
```

Out[7]:
```
total_images               0
last_price                 0
total_area                 0
first_day_exposition       0
rooms                      0
ceiling_height          9195
floors_total              86
living_area             1903
floor                      0
is_apartment           20924
studio                     0
open_plan                  0
kitchen_area            2278
balcony                11519
locality_name             49
airports_nearest        5542
cityCenters_nearest     5519
parks_around3000        5518
parks_nearest          15620
ponds_around3000        5518
ponds_nearest          14589
days_exposition         3181
dtype: int64
```

In [8]:
```
data.isna().mean()
```

Out[8]:
```
total_images           0.000000
last_price             0.000000
total_area             0.000000
first_day_exposition   0.000000
rooms                  0.000000
ceiling_height         0.387991
floors_total           0.003629
living_area            0.080299
floor                  0.000000
is_apartment           0.882906
```

```
studio                  0.000000
open_plan               0.000000
kitchen_area            0.096122
balcony                 0.486054
locality_name           0.002068
airports_nearest        0.233850
cityCenters_nearest     0.232879
parks_around3000        0.232837
parks_nearest           0.659100
ponds_around3000        0.232837
ponds_nearest           0.615596
days_exposition         0.134225
dtype: float64
```

In [9]:
```
data.duplicated().sum()
```

Out[9]: 0

Строк-дубликатов не выявлено. Исследуем пропуски и типы данных в остальных столбцах массива. Первые пять столбцов не вызывают нареканий по пропускам.
Сразу можно обратить внимание на столбец `days_exposition` . Тип данных `days_exposition` float, в то время, как количество дней целое число, но чтобы перевести тип данных в int, значения Nan нужно заменить нулями. А это повлияет на трактовку данных, получится, что часть объектов продана за ноль дней, такого быть не может. Поэтому оставим пропуски, как есть.

При ознакомлении со строками массива в общей информации мы убедились, что `first_day_exposition` - это дата в формате '%Y-%m-%dT%H:%M:%S'.
Заменим тип данных на datetime.

In [10]:
```
data['first_day_exposition'] = pd.to_datetime(data['first_day_exposition'], format='%
data.dtypes
```

Out[10]:
```
total_images                     int64
last_price                     float64
total_area                     float64
first_day_exposition    datetime64[ns]
rooms                            int64
ceiling_height                 float64
floors_total                   float64
living_area                    float64
floor                            int64
is_apartment                    object
studio                            bool
open_plan                         bool
kitchen_area                   float64
balcony                        float64
locality_name                   object
airports_nearest               float64
cityCenters_nearest            float64
parks_around3000               float64
parks_nearest                  float64
ponds_around3000               float64
ponds_nearest                  float64
```

```
days_exposition              float64
```

Рассмотрим первый параметр с пропусками - высота потолков. 9 195 пропусков, примерно 40%. Высота потолков зависит от типа квартир - времени их постройки ("сталинки" 3-3.5 м, "хрущевки" и "брежневки" примерно 2.5 м, панельные "9-этажки" 2.65 м, в "новостройках" от 2.5 до 3.2 м). Предполагаю, что часто в объявлении указан тип квартиры, по которому многие жители нашей страны могут предположить высоту потолка. Поэтому продавцы ее не уточняют. Скорее всего высоту потолка актуально указывать для новостроек и "сталинок" - там это может быть расценено как дополнительное преимущество. Оценим разброс значений высот потолков в наших данных:

In [11]:
```
data['ceiling_height'].unique()
```

Out[11]:
```
array([  2.7 ,    nan,   3.03,   2.5 ,   2.67,   2.56,   3.05,   2.75,
         2.6 ,   2.9 ,   2.8 ,   2.55,   3.  ,   2.65,   3.2 ,   2.61,
         3.25,   3.45,   2.77,   2.85,   2.64,   2.57,   4.15,   3.5 ,
         3.3 ,   2.71,   4.  ,   2.47,   2.73,   2.84,   3.1 ,   2.34,
         3.4 ,   3.06,   2.72,   2.54,   2.51,   2.78,   2.76,  25.  ,
         2.58,   3.7 ,   2.52,   5.2 ,   2.87,   2.66,   2.59,   2.  ,
         2.45,   3.6 ,   2.92,   3.11,   3.13,   3.8 ,   3.15,   3.55,
         3.16,   3.62,   3.12,   2.53,   2.74,   2.96,   2.46,   5.3 ,
         5.  ,   2.79,   2.95,   4.06,   2.94,   3.82,   3.54,   3.53,
         2.83,   4.7 ,   2.4 ,   3.38,   3.01,   5.6 ,   3.65,   3.9 ,
         3.18,   3.35,   2.3 ,   3.57,   2.48,   2.62,   2.82,   3.98,
         2.63,   3.83,   3.52,   3.95,   3.75,   2.88,   3.67,   3.87,
         3.66,   3.85,   3.86,   4.19,   3.24,   4.8 ,   4.5 ,   4.2 ,
         3.36,  32.  ,   3.08,   3.68,   3.07,   3.37,   3.09,   8.  ,
         3.26,   3.34,   2.81,   3.44,   2.97,   3.14,   4.37,   2.68,
         3.22,   3.27,  27.  ,   4.1 ,   2.93,   3.46,  24.  ,   3.47,
         3.33,   3.63,   3.32,  26.  ,   1.2 ,   8.3 ,   2.98,   2.86,
         3.17,   4.4 ,   3.28,   3.04,   4.45,   5.5 ,   3.84,   3.23,
         3.02,   3.21,   3.43,   3.78,   4.3 ,   3.39,   2.69,   3.31,
         4.65,   3.56,   2.2 ,   3.51,   3.93,   3.42,   2.99,   3.49,
        14.  ,   4.14,   2.91,   3.88,   1.75,   4.25,   3.29,  20.  ,
         2.25,   3.76,   3.69,   6.  ,  22.6 ,   2.89,   3.58,   5.8 ,
        27.5 ,   2.49,   4.9 ,   3.48,  10.3 ,   1.  , 100.  ,   3.59])
```

Высота потолка в частных домах может быть и 5, и 6 м. Предположим, что значения до 10 м относятся к таким строениям. В значениях 20, 25 и более явная ошибка. Измеряли не в метрах или разделитель поставили не туда. Приведем их к стандарту, разделив на 10.

In [12]:
```
data.loc[data['ceiling_height'] >= 20, 'ceiling_height'] = data.loc[data['ceiling_hei
data['ceiling_height'].unique()
```

Out[12]:
```
array([ 2.7 ,   nan,  3.03,  2.5 ,  2.67,  2.56,  3.05,  2.75,  2.6 ,
        2.9 ,  2.8 ,  2.55,  3.  ,  2.65,  3.2 ,  2.61,  3.25,  3.45,
        2.77,  2.85,  2.64,  2.57,  4.15,  3.5 ,  3.3 ,  2.71,  4.  ,
        2.47,  2.73,  2.84,  3.1 ,  2.34,  3.4 ,  3.06,  2.72,  2.54,
        2.51,  2.78,  2.76,  2.58,  3.7 ,  2.52,  5.2 ,  2.87,  2.66,
        2.59,  2.  ,  2.45,  3.6 ,  2.92,  3.11,  3.13,  3.8 ,  3.15,
        3.55,  3.16,  3.62,  3.12,  2.53,  2.74,  2.96,  2.46,  5.3 ,
        5.  ,  2.79,  2.95,  4.06,  2.94,  3.82,  3.54,  3.53,  2.83,
        4.7 ,  2.4 ,  3.38,  3.01,  5.6 ,  3.65,  3.9 ,  3.18,  3.35,
        2.3 ,  3.57,  2.48,  2.62,  2.82,  3.98,  2.63,  3.83,  3.52,
        3.95,  3.75,  2.88,  3.67,  3.87,  3.66,  3.85,  3.86,  4.19,
        3.24,  4.8 ,  4.5 ,  4.2 ,  3.36,  3.08,  3.68,  3.07,  3.37,
```

```
3.09,  8.  ,  3.26,  3.34,  2.81,  3.44,  2.97,  3.14,  4.37,
2.68,  3.22,  3.27,  4.1 ,  2.93,  3.46,  3.47,  3.33,  3.63,
3.32,  1.2 ,  8.3 ,  2.98,  2.86,  3.17,  4.4 ,  3.28,  3.04,
4.45,  5.5 ,  3.84,  3.23,  3.02,  3.21,  3.43,  3.78,  4.3 ,
3.39,  2.69,  3.31,  4.65,  3.56,  2.2 ,  3.51,  3.93,  3.42,
2.99,  3.49, 14.  ,  4.14,  2.91,  3.88,  1.75,  4.25,  3.29,
2.25,  3.76,  3.69,  6.  ,  2.26,  2.89,  3.58,  5.8 ,  2.49,
4.9 ,  3.48, 10.3 ,  1.  , 10.  ,  3.59])
```

Посмотрим на данные квартир с высотой потолка 2 м и меньше.

In [13]:
```
data.query('ceiling_height <= 2')
```

Out[13]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total
552	11	2999000.0	33.50	2017-03-30	1	2.00	15.0
2572	4	2400000.0	62.00	2018-12-27	3	2.00	5.0
4212	1	2120000.0	35.43	2017-12-18	1	2.00	18.0
5712	5	1500000.0	42.80	2017-08-14	2	1.20	2.0
5850	9	1650000.0	35.50	2018-05-29	1	2.00	5.0
11352	4	3300000.0	36.00	2017-06-18	1	2.00	17.0
16446	6	12300000.0	88.30	2019-03-12	3	2.00	5.0
16884	0	1500000.0	33.00	2018-10-03	1	2.00	5.0
16934	5	4100000.0	40.00	2017-10-17	1	1.75	37.0
17496	15	6700000.0	92.90	2019-02-19	3	2.00	17.0
19098	7	2700000.0	56.00	2018-02-04	3	2.00	5.0
19329	5	1700000.0	40.00	2018-07-15	2	2.00	9.0
20520	9	3600000.0	38.00	2018-07-05	1	2.00	12.0
22590	16	6000000.0	55.00	2018-10-31	2	1.00	12.0
22960	17	2300000.0	53.60	2018-10-18	2	2.00	2.0

Всего 15 строк, удалять их все не будем. Возможно, это подвальные помещения, чердаки, мансарды.

Найдем высоту потолков больше 10 м.

In [14]:
```
data.query('ceiling_height >= 10')
```

Out[14]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total
15061	19	5600000.0	56.4	2018-05-11	2	14.0	14.0
22309	20	5300000.0	45.0	2017-09-30	1	10.3	16.0
22869	0	15000000.0	25.0	2018-07-25	1	10.0	5.0

Ориентируясь на постановку задания отбросим значения меньше 1.2 и больше 10 м включительно. Хотя небольшое количество этих значений вряд ли серьезно повлияет на наши дальнейшие исследования.

In [15]:
```
drop_value = [1.00, 1.20, 10.0, 10.3, 14.0]
data = data.query('ceiling_height not in @drop_value')
data['ceiling_height'].unique()
```

Out[15]:
```
array([2.7 ,  nan, 3.03, 2.5 , 2.67, 2.56, 3.05, 2.75, 2.6 , 2.9 , 2.8 ,
       2.55, 3.  , 2.65, 3.2 , 2.61, 3.25, 3.45, 2.77, 2.85, 2.64, 2.57,
       4.15, 3.5 , 3.3 , 2.71, 4.  , 2.47, 2.73, 2.84, 3.1 , 2.34, 3.4 ,
       3.06, 2.72, 2.54, 2.51, 2.78, 2.76, 2.58, 3.7 , 2.52, 5.2 , 2.87,
       2.66, 2.59, 2.  , 2.45, 3.6 , 2.92, 3.11, 3.13, 3.8 , 3.15, 3.55,
       3.16, 3.62, 3.12, 2.53, 2.74, 2.96, 2.46, 5.3 , 5.  , 2.79, 2.95,
       4.06, 2.94, 3.82, 3.54, 3.53, 2.83, 4.7 , 2.4 , 3.38, 3.01, 5.6 ,
       3.65, 3.9 , 3.18, 3.35, 2.3 , 3.57, 2.48, 2.62, 2.82, 3.98, 2.63,
       3.83, 3.52, 3.95, 3.75, 2.88, 3.67, 3.87, 3.66, 3.85, 3.86, 4.19,
       3.24, 4.8 , 4.5 , 4.2 , 3.36, 3.08, 3.68, 3.07, 3.37, 3.09, 8.  ,
       3.26, 3.34, 2.81, 3.44, 2.97, 3.14, 4.37, 2.68, 3.22, 3.27, 4.1 ,
       2.93, 3.46, 3.47, 3.33, 3.63, 3.32, 8.3 , 2.98, 2.86, 3.17, 4.4 ,
       3.28, 3.04, 4.45, 5.5 , 3.84, 3.23, 3.02, 3.21, 3.43, 3.78, 4.3 ,
       3.39, 2.69, 3.31, 4.65, 3.56, 2.2 , 3.51, 3.93, 3.42, 2.99, 3.49,
       4.14, 2.91, 3.88, 1.75, 4.25, 3.29, 2.25, 3.76, 3.69, 6.  , 2.26,
       2.89, 3.58, 5.8 , 2.49, 4.9 , 3.48, 3.59])
```

Проверим, что первоначальное количество строк уменьшилось на 5 (число строк с откинутыми значениями высоты потолков). Первоначальное количество строк 23699.

In [16]:
```
data.shape
```

Out[16]:
```
(23694, 22)
```

Заполним пропущенные значения медианным значением высоты потолков в массиве, рассчитанным после обработки данных и устранения аномалий.

In [17]:
```
m = data['ceiling_height'].median()
data['ceiling_height'] = data['ceiling_height'].fillna(m)
data['ceiling_height'].isna().sum()
```

Out[17]:
```
0
```

Переходим к общему количеству этажей в здании. Посмотрим, в каких строках массива есть пропуски этого значения. Из общей информации знаем, что строк с пропусками общей "этажности" - 86.

In [18]:
```
data.loc[data['floors_total'].isna()].head(10)
```

Out[18]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	li
186	12	11640000.0	65.20	2018-10-02	2	2.65	NaN	
237	4	2438033.0	28.10	2016-11-23	1	2.65	NaN	
457	4	9788348.0	70.80	2015-08-01	2	2.65	NaN	
671	4	6051191.0	93.60	2017-04-06	3	2.65	NaN	
1757	5	3600000.0	39.00	2017-04-22	1	2.65	NaN	
1930	1	9750000.0	66.77	2016-05-13	2	2.65	NaN	
2392	7	5395770.0	53.00	2017-06-29	2	2.65	NaN	
2846	4	2325000.0	27.80	2016-09-21	1	2.65	NaN	
2952	10	6500000.0	54.30	2019-04-23	1	2.65	NaN	
3031	9	6400000.0	56.70	2018-03-27	2	2.65	NaN	

На основе этих данных предполагаем, что отсутствующие значения "этажности" зданий можно заменить на этаж квартиры, продаваемой в этом здании. т.к. обычно не указывают этаж для одноэтажных строений или для последних этажей. Заодно поменяем тип данных на целочисленный.

In [19]:
```
data['floors_total'] = data['floors_total'].fillna(data['floor']).astype('int')
```

Проверим общее количество пропусков по этому параметру и перестановку значений в конкретной строке (возьмем строку 186, ориентируясь на выведенные выше данные). Убедимся, что количество этажей равно указанному этажу.

In [20]:
```
data['floors_total'].isna().sum()
```

Out[20]:
```
0
```

In [21]:
```
data.loc[186]
```

Out[21]:
```
total_images                          12
last_price                     11640000.0
total_area                           65.2
first_day_exposition    2018-10-02 00:00:00
```

```
rooms                                 2
ceiling_height                     2.65
floors_total                          4
living_area                        30.8
floor                                 4
is_apartment                        NaN
studio                            False
open_plan                         False
kitchen_area                       12.0
balcony                             NaN
locality_name          Санкт-Петербург
airports_nearest                39197.0
cityCenters_nearest             12373.0
parks_around3000                    1.0
parks_nearest                     123.0
ponds_around3000                    0.0
ponds_nearest                       NaN
days_exposition                    49.0
Name: 186, dtype: object
```

Оценим количество вариантов по площадям: жилой и кухонной.

In [22]:
```
data['living_area'].value_counts(ascending=False)
```

Out[22]:
```
18.00    882
17.00    675
30.00    598
16.00    486
20.00    481
        ...
96.40      1
50.26      1
43.81      1
54.43      1
16.74      1
Name: living_area, Length: 1782, dtype: int64
```

In [23]:
```
data['kitchen_area'].value_counts(ascending=False)
```

Out[23]:
```
6.00     1300
10.00    1261
8.00     1110
9.00     1101
7.00     1062
         ...
7.12        1
13.73       1
14.74       1
53.10       1
16.74       1
Name: kitchen_area, Length: 971, dtype: int64
```

Видим, что эти значения весьма разнообразны. Пропусков в них приблизительно 8 и 10% от общей выборки. Можно было бы заполнить их медианными значениями. Но эти площади зависят, например, от типа здания, который нам неизвестен, или типа квартиры: апартаменты, студия, открытая планировка, в которых пропусков гораздо больше, чем данных. При замене медианными значениями нет общего признака, на который можно корректно опираться. Поэтому пропуски считаю нужным оставить, не заменяя другими

значениями.
Большее количество пропусков по кухонной площади объясняется наличием типов квартир: студия и с открытой планировкой, где кухня не может быть выделена из жилой или общей площади.
Разберемся с типами квартир. Пропуски есть только в данных по апартаментам. Тип данных квартир-студий и свободной планировки bool. А апартаментов object. Выведем уникальные значения для апартаментов.

In [24]:
```
data['is_apartment'].unique()
```

Out[24]:
```
array([nan, False, True], dtype=object)
```

Заменим пропуски на значение False, считая, что пропуски как раз и обозначают, что квартиру нельзя считать апартаментами.

In [25]:
```
data['is_apartment'] = data['is_apartment'].fillna(False)
data['is_apartment'].unique()
```

Out[25]:
```
array([False,  True])
```

Теперь можно объединить данные по типам квартир в один столбец `area_type` :

In [26]:
```
def area_type(row):

    if row['is_apartment'] == True:
        return 'is_apartment'

    if row['studio'] == True:
        return 'studio'

    if row['open_plan'] == True:
        return 'open_plan'

    return 'no info'

data['area_type'] = data.apply(area_type, axis=1)
data.groupby('area_type')['area_type'].count()
```

Out[26]:
```
area_type
is_apartment       49
no info         23429
open_plan          67
studio            149
Name: area_type, dtype: int64
```

In [27]:
```
data['area_type'].astype('str').head(5)
```

Out[27]:
```
0    no info
1    no info
2    no info
3    no info
4    no info
```

Name: area_type, dtype: object

Удалим не нужные теперь столбцы по каждому типу квартиры.

In [28]:
```
data = data.drop(columns=['is_apartment', 'studio', 'open_plan'])
data.columns
```

Out[28]:
```
Index(['total_images', 'last_price', 'total_area', 'first_day_exposition',
       'rooms', 'ceiling_height', 'floors_total', 'living_area', 'floor',
       'kitchen_area', 'balcony', 'locality_name', 'airports_nearest',
       'cityCenters_nearest', 'parks_around3000', 'parks_nearest',
       'ponds_around3000', 'ponds_nearest', 'days_exposition', 'area_type'],
      dtype='object')
```

Почти 50% пропусков в данных по количеству балконов. С большой долей вероятности эти данные не заполняют в случае отсутствия балкона. Заменим пропуски нулями и сразу изменим тип данных на целочисленный. Убедимся в корректном отображении данных.

In [29]:
```
data['balcony'] = data['balcony'].fillna(0).astype('int')
data.head()
```

Out[29]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	

Исследуем параметр `locality_name` . Подсчитаем количество уникальных названий населенных пунктов и выведем их для ознакомления.

In [30]:
```
data['locality_name'].value_counts()
```

Out[30]:
```
Санкт-Петербург                    15717
посёлок Мурино                       522
посёлок Шушары                       440
Всеволожск                           398
Пушкин                               369
                                   ...
посёлок Платформа 69-й километр        1
поселок Коробицыно                     1
посёлок Белоостров                     1
посёлок Шугозеро                       1
поселок Гончарово                      1
Name: locality_name, Length: 364, dtype: int64
```

In [31]:
```
data['locality_name'].unique()
```

Out[31]:
```
array(['Санкт-Петербург', 'посёлок Шушары', 'городской посёлок Янино-1',
       'посёлок Парголово', 'посёлок Мурино', 'Ломоносов', 'Сертолово',
       'Петергоф', 'Пушкин', 'деревня Кудрово', 'Коммунар', 'Колпино',
       'поселок городского типа Красный Бор', 'Гатчина', 'поселок Мурино',
       'деревня Фёдоровское', 'Выборг', 'Кронштадт', 'Кировск',
       'деревня Новое Девяткино', 'посёлок Металлострой',
       'посёлок городского типа Лебяжье',
       'посёлок городского типа Сиверский', 'поселок Молодцово',
       'поселок городского типа Кузьмоловский',
       'садовое товарищество Новая Ропша', 'Павловск',
       'деревня Пикколово', 'Всеволожск', 'Волхов', 'Кингисепп',
       'Приозерск', 'Сестрорецк', 'деревня Куттузи', 'посёлок Аннино',
       'поселок городского типа Ефимовский', 'посёлок Плодовое',
       'деревня Заклинье', 'поселок Торковичи', 'поселок Первомайское',
       'Красное Село', 'посёлок Понтонный', 'Сясьстрой', 'деревня Старая',
       'деревня Лесколово', 'посёлок Новый Свет', 'Сланцы',
       'село Путилово', 'Ивангород', 'Мурино', 'Шлиссельбург',
       'Никольское', 'Зеленогорск', 'Сосновый Бор', 'поселок Новый Свет',
       'деревня Оржицы', 'деревня Кальтино', 'Кудрово',
       'поселок Романовка', 'посёлок Бугры', 'поселок Бугры',
       'поселок городского типа Рощино', 'Кириши', 'Луга', 'Волосово',
       'Отрадное', 'село Павлово', 'поселок Оредеж', 'село Копорье',
       'посёлок городского типа Красный Бор', 'посёлок Молодёжное',
       'Тихвин', 'посёлок Победа', 'деревня Нурма',
       'поселок городского типа Синявино', 'Тосно',
       'посёлок городского типа Кузьмоловский', 'посёлок Стрельна',
       'Бокситогорск', 'посёлок Александровская', 'деревня Лопухинка',
       'Пикалёво', 'поселок Терволово',
       'поселок городского типа Советский', 'Подпорожье',
       'посёлок Петровское', 'посёлок городского типа Токсово',
       'поселок Сельцо', 'посёлок городского типа Вырица',
       'деревня Кипень', 'деревня Келози', 'деревня Вартемяги',
       'посёлок Тельмана', 'поселок Севастьяново',
       'городской поселок Большая Ижора', nan,
       'городской посёлок Павлово', 'деревня Агалатово',
       'посёлок Новогорелово', 'городской посёлок Лесогорский',
       'деревня Лаголово', 'поселок Цвелодубово',
       'поселок городского типа Рахья', 'поселок городского типа Вырица',
       'деревня Белогорка', 'поселок Заводской',
       'городской посёлок Новоселье', 'деревня Большие Колпаны',
       'деревня Горбунки', 'деревня Батово', 'деревня Заневка',
       'деревня Иссад', 'Приморск', 'городской посёлок Фёдоровское',
       'деревня Мистолово', 'Новая Ладога', 'поселок Зимитицы',
       'поселок Барышево', 'деревня Разметелево',
       'поселок городского типа имени Свердлова', 'деревня Пеники',
       'поселок Рябово', 'деревня Пудомяги', 'поселок станции Корнево',
       'деревня Низино', 'деревня Бегуницы', 'посёлок Поляны',
       'городской посёлок Мга', 'поселок Елизаветино',
       'посёлок городского типа Кузнечное', 'деревня Колтуши',
       'поселок Запорожское', 'посёлок городского типа Рощино',
       'деревня Гостилицы', 'деревня Малое Карлино',
       'посёлок Мичуринское', 'посёлок городского типа имени Морозова',
       'посёлок Песочный', 'посёлок Сосново', 'деревня Аро',
       'поселок Ильичёво', 'посёлок городского типа Тайцы',
       'деревня Малое Верево', 'деревня Извара', 'поселок станции Вещево',
       'село Паша', 'деревня Калитино',
       'посёлок городского типа Ульяновка', 'деревня Чудской Бор',
       'поселок городского типа Дубровка', 'деревня Мины',
```

```
'поселок Войсковицы', 'посёлок городского типа имени Свердлова',
'деревня Коркино', 'посёлок Ропша',
'поселок городского типа Приладожский', 'посёлок Щеглово',
'посёлок Гаврилово', 'Лодейное Поле', 'деревня Рабитицы',
'поселок городского типа Никольский', 'деревня Кузьмолово',
'деревня Малые Колпаны', 'поселок Тельмана',
'посёлок Петро-Славянка', 'городской посёлок Назия',
'посёлок Репино', 'посёлок Ильичёво', 'поселок Углово',
'поселок Старая Малукса', 'садовое товарищество Рахья',
'поселок Аннино', 'поселок Победа', 'деревня Меньково',
'деревня Старые Бегуницы', 'посёлок Сапёрный', 'поселок Семрино',
'поселок Гаврилово', 'поселок Глажево', 'поселок Кобринское',
'деревня Гарболово', 'деревня Юкки',
'поселок станции Приветнинское', 'деревня Мануйлово',
'деревня Пчева', 'поселок Поляны', 'поселок Цвылёво',
'поселок Мельниково', 'посёлок Пудость', 'посёлок Усть-Луга',
'Светогорск', 'Любань', 'поселок Селезнёво',
'поселок городского типа Рябово', 'Каменногорск', 'деревня Кривко',
'поселок Глебычево', 'деревня Парицы', 'поселок Жилпосёлок',
'посёлок городского типа Мга', 'городской поселок Янино-1',
'посёлок Войскорово', 'село Никольское', 'посёлок Терволово',
'поселок Стеклянный', 'посёлок городского типа Важины',
'посёлок Мыза-Ивановка', 'село Русско-Высоцкое',
'поселок городского типа Лебяжье',
'поселок городского типа Форносово', 'село Старая Ладога',
'поселок Житково', 'городской посёлок Виллози', 'деревня Лампово',
'деревня Шпаньково', 'деревня Лаврики', 'посёлок Сумино',
'посёлок Возрождение', 'деревня Старосиверская',
'посёлок Кикерино', 'поселок Возрождение',
'деревня Старое Хинколово', 'посёлок Пригородный',
'посёлок Торфяное', 'городской посёлок Будогощь',
'поселок Суходолье', 'поселок Красная Долина', 'деревня Хапо-Ое',
'поселок городского типа Дружная Горка', 'поселок Лисий Нос',
'деревня Яльгелево', 'посёлок Стеклянный', 'село Рождествено',
'деревня Старополье', 'посёлок Левашово', 'деревня Сяськелево',
'деревня Камышовка',
'садоводческое некоммерческое товарищество Лесная Поляна',
'деревня Хязельки', 'поселок Жилгородок',
'посёлок городского типа Павлово', 'деревня Ялгино',
'поселок Новый Учхоз', 'городской посёлок Рощино',
'поселок Гончарово', 'поселок Почап', 'посёлок Сапёрное',
'посёлок Платформа 69-й километр', 'поселок Каложицы',
'деревня Фалилеево', 'деревня Пельгора',
'поселок городского типа Лесогорский', 'деревня Торошковичи',
'посёлок Белоостров', 'посёлок Алексеевка', 'поселок Серебрянский',
'поселок Лукаши', 'поселок Петровское', 'деревня Щеглово',
'поселок Мичуринское', 'деревня Тарасово', 'поселок Кингисеппский',
'посёлок при железнодорожной станции Вещево', 'поселок Ушаки',
'деревня Котлы', 'деревня Сижно', 'деревня Торосово',
'посёлок Форт Красная Горка', 'поселок городского типа Токсово',
'деревня Новолисино', 'посёлок станции Громово', 'деревня Глинка',
'посёлок Мельниково', 'поселок городского типа Назия',
'деревня Старая Пустошь', 'поселок Коммунары', 'поселок Починок',
'посёлок городского типа Вознесенье', 'деревня Разбегаево',
'посёлок городского типа Рябово', 'поселок Гладкое',
'посёлок при железнодорожной станции Приветнинское',
'поселок Тёсово-4', 'посёлок Жилгородок', 'деревня Бор',
'посёлок Коробицыно', 'деревня Большая Вруда', 'деревня Курковицы',
'посёлок Лисий Нос', 'городской посёлок Советский',
'посёлок Кобралово', 'деревня Суоранда', 'поселок Кобралово',
'поселок городского типа Кондратьево',
'коттеджный поселок Счастье', 'поселок Любань', 'деревня Реброво',
```

```
'деревня Зимитицы', 'деревня Тойворово', 'поселок Семиозерье',
'поселок Лесное', 'поселок Совхозный', 'поселок Усть-Луга',
'посёлок Ленинское', 'посёлок Суйда',
'посёлок городского типа Форносово', 'деревня Нижние Осельки',
'посёлок станции Свирь', 'поселок Перово', 'Высоцк',
'поселок Гарболово', 'село Шум', 'поселок Котельский',
'поселок станции Лужайка', 'деревня Большая Пустомержа',
'поселок Красносельское', 'деревня Вахнова Кара', 'деревня Пижма',
'коттеджный поселок Кивеннапа Север', 'поселок Коробицыно',
'поселок Ромашки', 'посёлок Перово', 'деревня Каськово',
'деревня Куровицы', 'посёлок Плоское', 'поселок Сумино',
'поселок городского типа Большая Ижора', 'поселок Кирпичное',
'деревня Ям-Тесово', 'деревня Раздолье', 'деревня Терпилицы',
'посёлок Шугозеро', 'деревня Ваганово', 'поселок Пушное',
'садовое товарищество Садко', 'посёлок Усть-Ижора',
'деревня Выскатка', 'городской посёлок Свирьстрой',
'поселок Громово', 'деревня Кисельня', 'посёлок Старая Малукса',
'деревня Трубников Бор', 'поселок Калитино',
'посёлок Высокоключевой', 'садовое товарищество Приладожский',
'посёлок Пансионат Зелёный Бор', 'деревня Ненимяки',
'поселок Пансионат Зелёный Бор', 'деревня Снегирёвка',
'деревня Рапполово', 'деревня Пустынка', 'поселок Рабитицы',
'деревня Большой Сабск', 'деревня Русско', 'деревня Лупполово',
'деревня Большое Рейзино', 'деревня Малая Романовка',
'поселок Дружноселье', 'поселок Пчевжа', 'поселок Володарское',
'деревня Нижняя', 'коттеджный посёлок Лесное', 'деревня Тихковицы'
```

364 названия, в которых встречаются неявные дубликаты: варианты написания "посёлок" и "поселок", а также "городской поселок", "поселок городского типа", "поселок станции". Избавимся сначала от буквы "ё", а потом назовем все типы поселков - "поселок".

In [32]:
```
data['locality_name'] = data['locality_name'].str.replace('ё', 'е')
data['locality_name'] = data['locality_name'].str.replace('городской поселок', 'посел
data['locality_name'] = data['locality_name'].str.replace('поселок городского типа',
data['locality_name'] = data['locality_name'].str.replace('коттеджный поселок', 'посе
data['locality_name'] = data['locality_name'].str.replace('поселок станции', 'поселок
data['locality_name'].value_counts()
```

Out[32]:
```
Санкт-Петербург            15717
поселок Мурино               556
поселок Шушары               440
Всеволожск                   398
Пушкин                       369
                           ...
поселок Красносельское         1
деревня Тойворово              1
деревня Чудской Бор            1
деревня Шпаньково              1
поселок Жилпоселок             1
Name: locality_name, Length: 320, dtype: int64
```

Количество уникальных значений уменьшилось до 320.

Остались пропуски в данных, заполненных автоматически:

- airports_nearest 0.233850
- cityCenters_nearest 0.232879
- parks_around3000 0.232837
- parks_nearest 0.659100

- ponds_around3000 0.232837
- ponds_nearest 0.615596

Похоже для расчета взяли не слишком подробную карту, где не хватает координат мелких населенных пунктов, не тот масштаб. Что подтверждается более 60 % пропусков для ближайших парков и прудов, в деревнях и селах они могут быть не отмечены на крупномасштабной карте.

Выведем данные по пропускам значений ближайшего аэропорта. Понимаем, что для большинства населенных пунктов ближайший аэропорт находится в Санкт-Петербурге.

In [33]:
```
data.loc[data['airports_nearest'].isna()].head(10)
```

Out[33]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living
5	10	2890000.0	30.40	2018-09-10	1	2.65	12	
8	20	2900000.0	33.16	2018-05-23	1	2.65	27	
12	10	3890000.0	54.00	2016-06-30	2	2.65	5	
22	20	5000000.0	58.00	2017-04-24	2	2.75	25	
30	12	2200000.0	32.80	2018-02-19	1	2.65	9	
37	10	1990000.0	45.80	2017-10-28	2	2.50	5	
38	10	3150000.0	40.00	2018-03-29	1	2.75	18	
47	17	3600000.0	56.10	2018-10-18	3	2.65	4	
60	3	2740000.0	35.00	2018-01-01	1	2.65	12	
62	0	4800000.0	78.60	2017-09-17	3	2.80	9	

Большинство данных представлены для Санкт-Петербурга, расстояния до прудов, парков и аэропорта там существенно отличаются в зависимости от района города. А для мелких населенных пунктов этих данных может вообще не быть. Поэтому пропуски в картографических данных оставим, как есть.

Обработаем выбросы данных и аномальные значения. Помним про существенный разброс цен наших объектов. Попробуем нарезать массив на 10 интервалов равного размера. На основании этих данных и данных метода describe (4 интервала - границы квартилей), приведенных выше, исследуем значения 1-го и последнего интервалов.

In [34]:
```
pd.qcut(data['last_price'], q=10)
```

Out[34]:
```
0        (10908680.0, 763000000.0]
```

```
1           (3190000.0, 3650000.0]
2           (4650000.0, 5290000.0]
3        (10908680.0, 763000000.0]
4          (7585000.0, 10908680.0]
                    ...
23694      (7585000.0, 10908680.0]
23695       (2500000.0, 3190000.0]
23696       (12189.999, 2500000.0]
23697    (10908680.0, 763000000.0]
23698       (12189.999, 2500000.0]
Name: last_price, Length: 23694, dtype: category
Categories (10, interval[float64]): [(12189.999, 2500000.0] < (2500000.0, 3190000.0]
< (3190000.0, 3650000.0] < (3650000.0, 4101884.8] ... (5290000.0, 6190000.0] < (61900
```

In [35]:
```
data.query('last_price < 500000')
```

Out[35]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	li
5698	7	450000.0	42.0	2017-07-31	2	2.65	1	
6859	6	490000.0	40.0	2017-12-11	1	2.65	5	
8793	7	12190.0	109.0	2019-03-20	2	2.75	25	
9581	7	450000.0	43.4	2018-08-31	2	2.65	5	
10782	3	490000.0	29.1	2016-05-05	1	2.50	5	
14911	5	430000.0	54.0	2018-06-26	2	2.65	3	
16032	8	480000.0	32.0	2019-01-06	1	2.65	2	
16219	14	450000.0	38.5	2018-07-11	2	2.65	2	
16274	18	440000.0	40.0	2018-07-10	1	2.65	5	
17456	7	430000.0	30.4	2019-04-22	1	2.65	2	
17676	0	450000.0	36.5	2018-02-01	1	2.65	5	
18867	1	470000.0	41.0	2018-02-11	1	2.65	5	
21912	0	470000.0	37.0	2018-02-18	1	2.65	3	
23484	11	470000.0	44.5	2018-07-02	2	2.65	2	

In [36]:
```
data.query('last_price > 20000000')
```

Out[36]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total
3	0	64900000.0	159.0	2015-07-24	3	2.65	14
42	13	22000000.0	161.8	2015-07-08	4	2.80	4
51	7	45000000.0	161.0	2017-10-17	3	3.20	8
121	20	33000000.0	180.0	2017-02-17	3	2.90	9
147	10	27700000.0	179.6	2017-07-06	5	2.85	25
...	...	...	...	...	...	...	...
23491	20	21800000.0	250.0	2017-09-16	3	2.65	12
23516	3	22529250.0	139.5	2018-07-04	4	3.30	16
23574	14	64990000.0	139.0	2015-11-24	3	3.00	8
23590	0	21187872.0	123.3	2017-04-25	3	2.65	18
23684	20	21400000.0	145.0	2018-11-02	4	3.00	26

Видим, что в довольно малых населенных пунктах цены на объекты начинаются с 430000. Исключение - цена 12190 на квартиру площадью больше 100 квадратов в СПб. Это явная ошибка.
В то же время цены на квартиры в СПб хоть и находятся в большинстве своем в пределах до 7 млн, но дальше (выше 3 квартиля) растут примерно в 100 раз. Попробуем отбросить самые высокие значения. Квартир со стоимостью выше 10 млн - 2720, более 20 млн - 707 квартир. Для этой выборки предлагаю отбросить квартиры со стоимостью выше 20 млн. и убрать ошибку в 12 тыс. И перевести тип данных в int. В дальнейшем корректнее было бы сравнивать цены СПб между собой или хотя бы с крупными городами области, а не со всеми поселками и деревнями.

In [37]:
```
data['last_price'] = data['last_price'].astype('int')
data = data.query('last_price < 20000000 and last_price != 12190')
data.shape
```

Out[37]: (22970, 20)

Отсекать слишком большие значения данных по общей площади и количеству комнат не будем, большая часть ушла вместе с ценами более 20 млн. Остальные данные адекватны ценам и площадям продаваемых в регионах объектов.
Для дальнейших расчетов изменим тип данных в столбце с общей площадью на целочисленный.

In [38]:
```
data['total_area'] = data['total_area'].astype('int')
data.head()
```

Out[38]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_
0	20	13000000	108	2019-03-07	3	2.70	16	
1	7	3350000	40	2018-12-04	1	2.65	11	
2	10	5196000	56	2015-08-20	2	2.65	5	
4	2	10000000	100	2018-06-19	2	3.03	14	
5	10	2890000	30	2018-09-10	1	2.65	12	

Так как общая площадь прямо пропорциональна жилой, проверим данные по слишком маленьким жилым площадям и выведем их совместно с другими параметрами.

In [39]:
```
data.query('living_area < 10')
```

Out[39]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	li
114	5	4000000	21	2019-02-07	1	2.90	8	
680	14	7200000	43	2017-10-31	1	2.65	8	
1326	8	8100000	52	2017-01-29	2	2.70	25	
2309	10	4200000	62	2017-06-15	2	2.60	9	
3242	7	4440000	41	2017-07-02	1	2.65	17	
4100	17	5300000	34	2017-06-19	1	2.70	23	
4542	12	3300000	18	2018-11-07	1	3.50	5	
7312	8	3400000	27	2018-02-21	2	2.50	15	
8325	9	4800000	52	2017-10-25	2	2.65	5	
13915	20	6350000	52	2018-02-06	2	3.00	6	
15833	20	4600000	33	2017-01-01	1	2.70	22	

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	li
16431	13	3799000	31	2018-04-12	1	2.60	5	
17248	20	5300000	33	2017-04-14	1	2.70	22	
17582	11	2680000	22	2018-08-11	0	2.65	25	
19251	19	4050000	33	2018-10-15	1	2.50	22	
19620	10	4300000	33	2018-02-01	1	2.65	5	
20994	7	8900000	50	2018-10-22	2	2.50	7	
21505	9	4100000	35	2018-01-10	1	2.75	27	
21758	0	2330000	23	2018-01-01	0	2.65	24	
21908	9	5300000	46	2018-03-20	1	2.75	7	
21943	15	6100000	77	2019-02-28	4	2.50	9	
22252	4	3340000	37	2018-02-08	1	2.80	8	
22473	0	3490304	33	2015-12-22	2	2.65	13	
23208	12	4800000	37	2016-02-24	1	2.65	14	

Учетная нормативная площадь жилого помещения на 1 человека в коммуналке — от 6 до 15 кв.м.

Откинем данные по жилым площадям меньшим 6 кв.м. (минимальный размер комнаты в питерском общежитии).

In [40]:
```
data = data.query('living_area > 5')
data.shape
```

Out[40]: (21158, 20)

Так как ранее мы удаляли строки, нужно обновить индексы датафрейма и убедиться, что они обновились корректно. в этом нам поможет вторая строка выводимой информации.

In [41]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 21158 entries, 0 to 23698
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
```

```
---  ------                --------------  -----
 0   total_images          21158 non-null  int64
 1   last_price            21158 non-null  int32
 2   total_area            21158 non-null  int32
 3   first_day_exposition  21158 non-null  datetime64[ns]
 4   rooms                 21158 non-null  int64
 5   ceiling_height        21158 non-null  float64
 6   floors_total          21158 non-null  int32
 7   living_area           21158 non-null  float64
 8   floor                 21158 non-null  int64
 9   kitchen_area          20363 non-null  float64
 10  balcony               21158 non-null  int32
 11  locality_name         21115 non-null  object
 12  airports_nearest      16258 non-null  float64
 13  cityCenters_nearest   16274 non-null  float64
 14  parks_around3000      16275 non-null  float64
 15  parks_nearest         7042 non-null   float64
 16  ponds_around3000      16275 non-null  float64
 17  ponds_nearest         7947 non-null   float64
 18  days_exposition       18347 non-null  float64
 19  area_type             21158 non-null  object
dtypes: datetime64[ns](1), float64(10), int32(4), int64(3), object(2)
```

In [42]:

```
data = data.reset_index(drop=True)
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21158 entries, 0 to 21157
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   total_images          21158 non-null  int64
 1   last_price            21158 non-null  int32
 2   total_area            21158 non-null  int32
 3   first_day_exposition  21158 non-null  datetime64[ns]
 4   rooms                 21158 non-null  int64
 5   ceiling_height        21158 non-null  float64
 6   floors_total          21158 non-null  int32
 7   living_area           21158 non-null  float64
 8   floor                 21158 non-null  int64
 9   kitchen_area          20363 non-null  float64
 10  balcony               21158 non-null  int32
 11  locality_name         21115 non-null  object
 12  airports_nearest      16258 non-null  float64
 13  cityCenters_nearest   16274 non-null  float64
 14  parks_around3000      16275 non-null  float64
 15  parks_nearest         7042 non-null   float64
 16  ponds_around3000      16275 non-null  float64
 17  ponds_nearest         7947 non-null   float64
 18  days_exposition       18347 non-null  float64
 19  area_type             21158 non-null  object
dtypes: datetime64[ns](1), float64(10), int32(4), int64(3), object(2)
memory usage: 2.7+ MB
```

Вывод:

Данные обработаны и готовы к проведению исследования. В ходе предобработки:

- изменен тип данных на datetime для столбца с датой публикации объявления;
- устранены ошибки в данных по высоте потолков и откинуты значения выше 10 м и

ниже 1.2 м, заменены пропуски по высоте потолков на медианное значение после обработки данных;
- пробелы в общем количестве этажей заполнены значениями этажа, на основании гипотезы, что пропущены значения для первых и последних этажей;
- создан единый столбец с типами квартир и удалены отдельные столбцы по каждому типу;
- пропуски в количестве балконов заменены на нулевые значения;
- устранены неявные дубликаты в названиях населенных пунктов;
- цены на объекты ограничены значениями в 430000 и 20 млн. руб.;
- утранены из выборки объекты с жилой площадью менее 6 кв.м;

Добавление в таблицу новых столбцов для расчетов

Для дальнейшего анализа необходимо добавить в таблицу ряд расчетных значений. Рассчитаем цену одного квадратного метра. Добавим в массив новый столбец с этими данными по каждому объекту.

In [43]:
```
data['price_area'] = (data['last_price'] / data['total_area']).astype('int')
data['price_area'].head()
```

Out[43]:
```
0    120370
1     83750
2     92785
3    100000
4     96333
Name: price_area, dtype: int32
```

Определим день недели публикации объявления. Добавим данные в новый столбец.

In [44]:
```
data['weekday'] = data['first_day_exposition'].dt.weekday
data['weekday'].head()
```

Out[44]:
```
0    3
1    1
2    3
3    1
4    0
Name: weekday, dtype: int64
```

Таким же образом создадим отдельные столбцы для месяца и года публикации объявления.

In [45]:
```
data['month'] = data['first_day_exposition'].dt.month
data['month'].head()
```

Out[45]:
```
0     3
1    12
2     8
3     6
4     9
Name: month, dtype: int64
```

In [46]:
```
data['year'] = data['first_day_exposition'].dt.year
data['year'].head()
```

Out[46]:
```
0    2019
1    2018
2    2015
3    2018
4    2018
Name: year, dtype: int64
```

Создадим функцию, которая определяет тип этажа квартиры (значения — «первый», «последний», «другой»). Добавим эти значения в новый столбец `floor_type` .

In [47]:
```
def floor_type(row):

    if row['floor'] == 1:
        return 'первый'

    if row['floor'] == row['floors_total']:
        return 'последний'

    return 'другой'

data['floor_type'] = data.apply(floor_type, axis=1)
data.groupby('floor_type')['floor_type'].count()
```

Out[47]:
```
floor_type
другой       15532
первый        2626
последний     3000
Name: floor_type, dtype: int64
```

In [48]:
```
data['floor_type'].astype('str')
```

Out[48]:
```
0        другой
1        первый
2        другой
3        другой
4        другой
          ...
21153    другой
21154    другой
21155    другой
21156    первый
21157    первый
Name: floor_type, Length: 21158, dtype: object
```

В новом столбце рассчитаем расстояние до центра города в километрах (переведем из м в км и округлим до целых значений).

In [49]:
```
data['cityCenters_km'] = (data['cityCenters_nearest'] / 1000).round()
data.head(10)
```

Out[49]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_
0	20	13000000	108	2019-03-07	3	2.70	16	5
1	7	3350000	40	2018-12-04	1	2.65	11	1
2	10	5196000	56	2015-08-20	2	2.65	5	3
3	2	10000000	100	2018-06-19	2	3.03	14	3
4	10	2890000	30	2018-09-10	1	2.65	12	1
5	6	3700000	37	2017-11-02	1	2.65	26	1
6	20	2900000	33	2018-05-23	1	2.65	27	1
7	18	5400000	61	2017-02-26	3	2.50	9	4
8	5	5050000	39	2017-11-16	1	2.67	12	2

Вывод:

В соответствии с целями исследования в массив добавлены следующие данные:

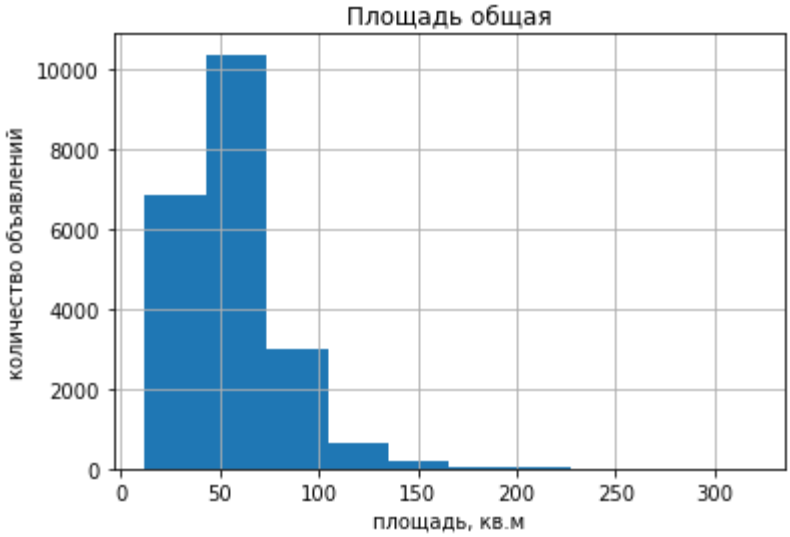
- цена одного квадратного метра;
- день недели публикации объявления;
- месяц публикации объявления;
- год публикации объявления;
- тип этажа квартиры;
- расстояние до центра города в километрах.

Исследовательский анализ данных

Изучим следующие параметры объектов:

Общая площадь

In [50]:
```
data['total_area'].hist()
plt.title('Площадь общая')
plt.xlabel('площадь, кв.м')
plt.ylabel('количество объявлений')
plt.show()
```



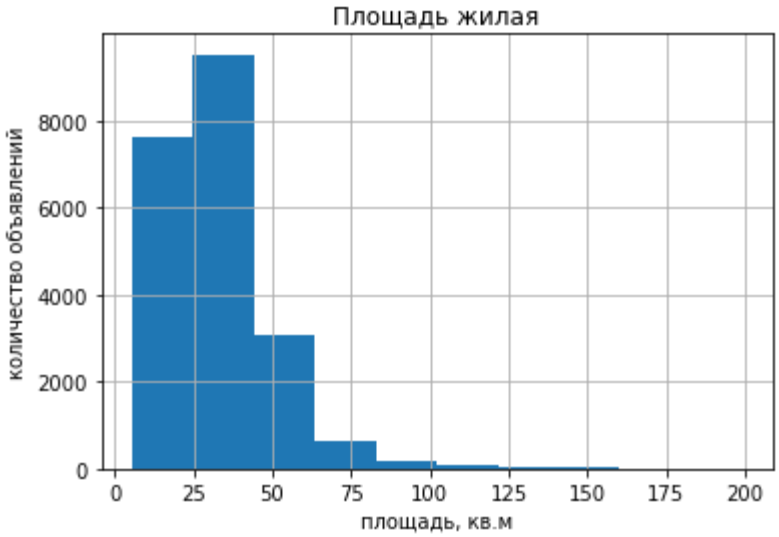
In [51]:
```
data['total_area'].describe()
```

Out[51]:
```
count    21158.000000
mean        56.433359
std         24.529912
min         12.000000
25%         39.000000
50%         51.000000
75%         67.000000
max        320.000000
Name: total_area, dtype: float64
```

Распределение данных в гистограмме тяготеет к нормальному. Видим, что большая часть продаваемых квартир имеет площадь около 50 кв.м. Основной разброс значений в пределах от 15 до 100 кв.м. Квартир с площадью меньше 50 квадратов продается больше, чем с площадью от 50 до 100 кв.м. Квартиры меньшей площади скорее всего относятся к небольшим населенным пунктам. От 50 кв. м - это СПб и более менее крупные города области. Есть варианты с площадью больше 150 кв.м и более 200, но таких гораздо меньше (видимо, элитные объекты в СПб).

Жилая площадь

In [52]:
```
data['living_area'].hist()
plt.title('Площадь жилая')
plt.xlabel('площадь, кв.м')
plt.ylabel('количество объявлений')
plt.show()
```



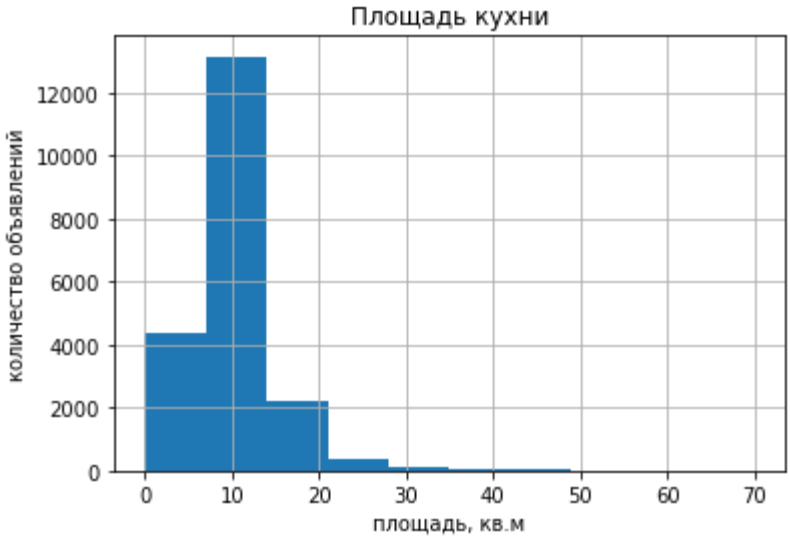
In [53]:
```
data['living_area'].describe()
```

Out[53]:
```
count    21158.000000
mean        32.540819
std         16.677331
min          5.400000
25%         18.500000
50%         30.000000
75%         41.300000
max        199.000000
Name: living_area, dtype: float64
```

Распределение данных в этой гистограмме практически копия распределения по общей площади, только в меньшем масштабе. Что логично - жилая площадь меньше общей. Среднее значение жилой площади около 30 кв.м. Довольно много малогабаритных квартир, комнат в общежитиях с жилой площадью около 15-20 кв.м. Есть выбор и среди объектов с жилой площадью 40-60 кв.м.

Площадь кухни

In [54]:
```
data['kitchen_area'].hist(range=(0, 70))
plt.title('Площадь кухни')
plt.xlabel('площадь, кв.м')
plt.ylabel('количество объявлений')
plt.show()
```



In [55]:
```
data['kitchen_area'].describe()
```

Out[55]:
```
count    20363.000000
mean        10.075960
std          4.684155
min          1.300000
25%          7.000000
50%          9.000000
75%         11.500000
max        100.700000
Name: kitchen_area, dtype: float64
```

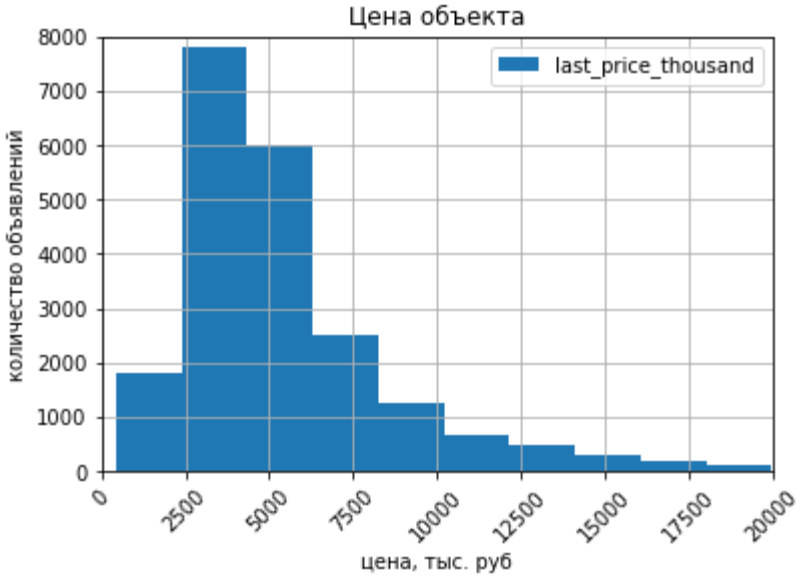
Ограничим верхний диапазон измерений 70 кв.м. для более четкого отображения интервалов данных. При этом знаем, что есть редкие объекты с площадью кухни 100 кв.м и близкие к ним.

В среднем площадь кухни в продаваемых объектах от 8 до 11 кв.м. Довольно много объектов с площадью от 1 до 9 кв.м, что объясняется наличием малогабаритных квартир в старом жилом фонде, возможно, продажей комнат в общежитиях (коммуналках). В 2 раза реже, чем кухни меньше 9 кв.м, встречаются кухни от 12 до 20 кв.м. Похоже кухни в новостройках тоже невелики, или вторичного жилья в данном массиве больше, чем нового.

Цена объекта

Для более наглядного отображения цен заведем новый столбец, где разделим столбец с ценами на 1000. И построим гистограмму с помощью plot с поворотом значений по оси X.

In [56]:
```
data['last_price_thousand'] = data['last_price'] / 1000
data.plot(y='last_price_thousand', kind='hist', rot=45, grid=True)
plt.xlim(0, 20000)
plt.ylim(0, 8000)
plt.title('Цена объекта')
plt.xlabel('цена, тыс. руб')
plt.ylabel('количество объявлений')
plt.show()
```



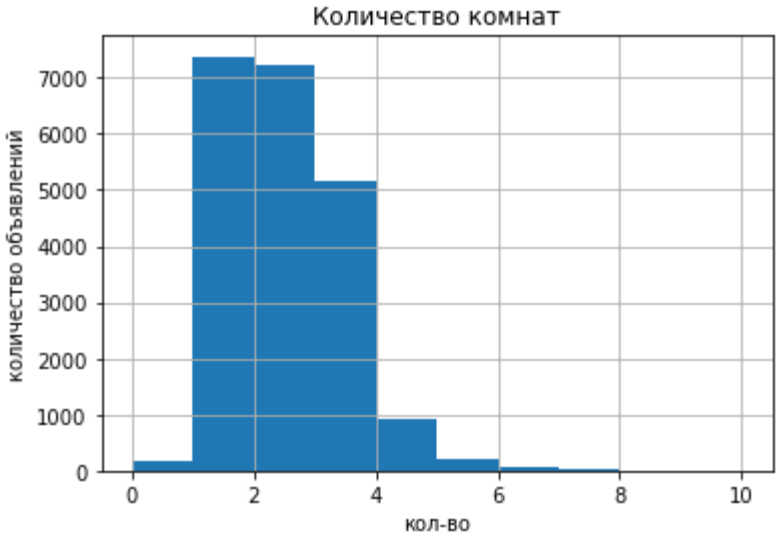
In [57]:
```
data['last_price'].describe().round()
```

Out[57]:
```
count       21158.0
mean      5426053.0
std       3171657.0
min        430000.0
25%       3400000.0
50%       4580000.0
75%       6500000.0
max      19999000.0
Name: last_price, dtype: float64
```

Большая часть объектов находится в ценовом диапазоне примерно от 2.5 млн. руб. до 6.5 млн. руб. Средняя (по медиане) цена объекта 4.5 млн. руб. Квартиры в ценовом диапозоне до 2.5 млн объясняются низкими ценами в отдаленных и мелких населенных пунктах, а также наличием квартир малой площади в самом Санкт-Петербурге(цены начинаются от 430 тыс. руб.). Максимальную цену мы ранее ограничили 20 млн. руб.

Количество комнат

In [58]:
```
data['rooms'].hist(range=(0, 10))
plt.title('Количество комнат')
plt.xlabel('кол-во')
plt.ylabel('количество объявлений')
plt.show()
```



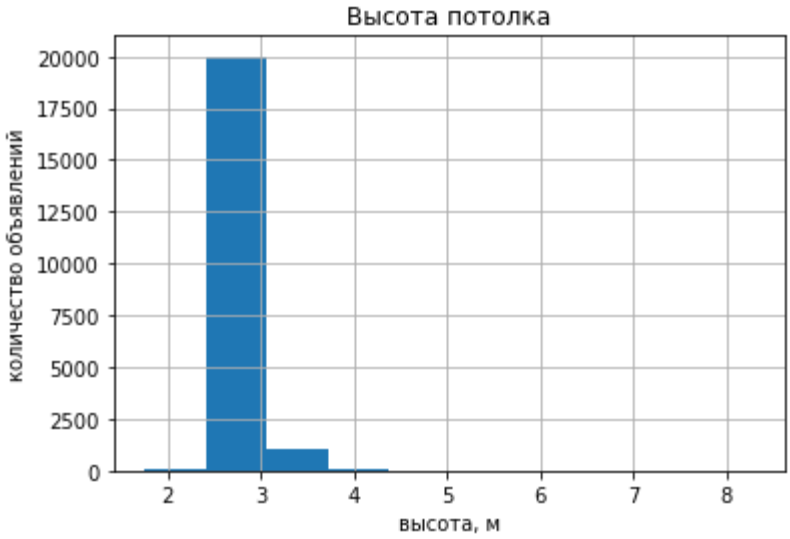
In [59]:
```
data['rooms'].describe()
```

Out[59]:
```
count    21158.000000
mean         2.012147
std          0.985356
min          0.000000
25%          1.000000
50%          2.000000
75%          3.000000
max         11.000000
Name: rooms, dtype: float64
```

Установим диапазон от нуля до 10 в соответствии с количеством корзин по умолчанию. Знаем, что у нас есть и редкие объекты с большим количеством комнат (большинство из них мы убрали из массива ранее).
Чаще всего на рынке предлагаются двухкомнатные квартиры, большой выбор в диапозоне от 1 до 4 комнат. Наличие объектов с 0 комнат объясняется присутствием жилья свободной планировки и, возможно, студий.

Высота потолков

In [60]:
```
data['ceiling_height'].hist()
plt.title('Высота потолка')
plt.xlabel('высота, м')
plt.ylabel('количество объявлений')
plt.show()
```



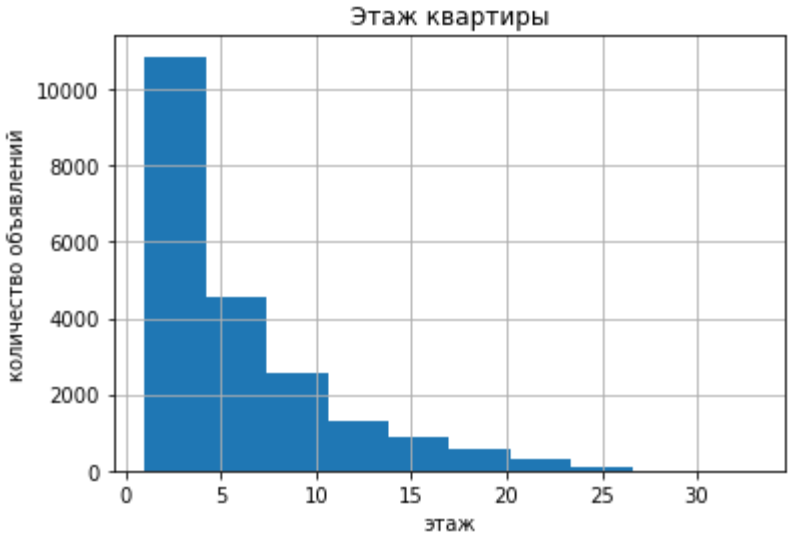
In [61]:
```
data['ceiling_height'].describe()
```

Out[61]:
```
count    21158.000000
mean         2.688387
std          0.216748
min          1.750000
25%          2.600000
50%          2.650000
75%          2.700000
max          8.300000
Name: ceiling_height, dtype: float64
```

Подавляющее большинство объектов имеет высоту потолков примерно от 2.6 до 2.7 метров. Так как большинство объектов из СПб, это может объяснятся высокими потолками в старом жилом фонде, наличием новостроек и коттеджей среди объектов. Потолки выше 3 метров - мы ранее отнесли к домам и коттеджам, продаваемым в области. Потолки 2 м и ниже могут встречаться в старом жилом фонде: мансарды, полуподвалы, цокольные этажи.

Этаж квартиры

In [62]:
```
data['floor'].hist()
plt.title('Этаж квартиры')
plt.xlabel('этаж')
plt.ylabel('количество объявлений')
plt.show()
```



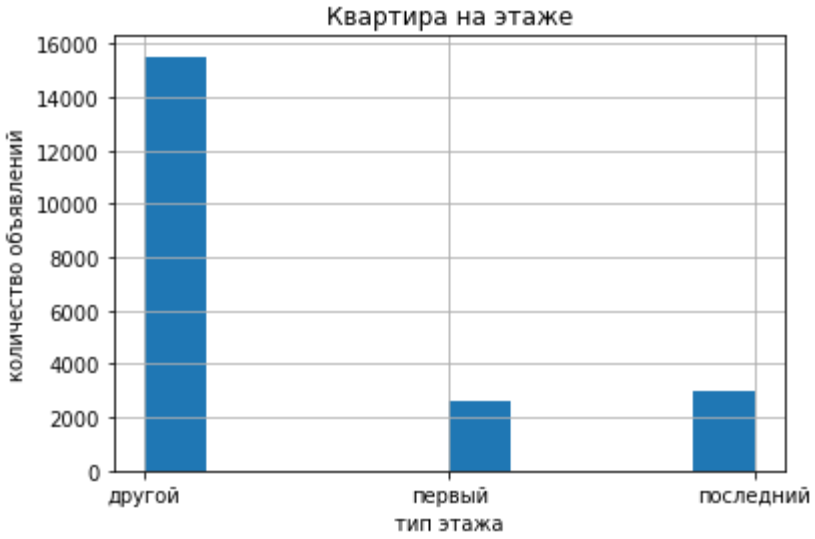
In [63]:
```
data['floor'].describe()
```

Out[63]:
```
count    21158.000000
mean         5.892807
std          4.881131
min          1.000000
25%          2.000000
50%          4.000000
75%          8.000000
max         33.000000
Name: floor, dtype: float64
```

Большая часть объектов находится в интервале от 2 до 8 этажей. Помним, что есть объект и в 30 этажей. Но это редкость. Малоэтажного жилья продается больше (1-5 этаж), начиная с 5 этажа: чем выше этажность зданий, тем меньше продаваемых в них квартир.

Тип этажа квартиры («первый», «последний», «другой»)

In [64]:
```
data['floor_type'].hist()
plt.title('Квартира на этаже')
plt.xlabel('тип этажа')
plt.ylabel('количество объявлений')
plt.show()
```



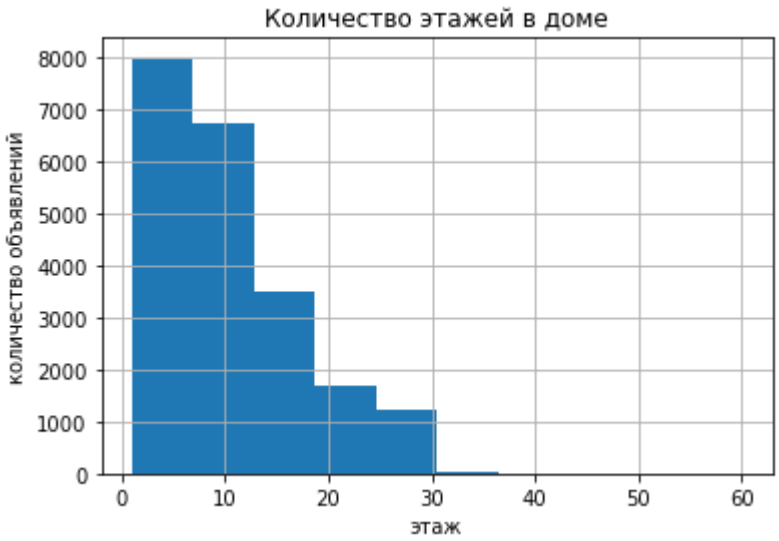
In [65]:
```
data['floor_type'].describe()
```

Out[65]:
```
count      21158
unique         3
top       другой
freq       15532
Name: floor_type, dtype: object
```

Из 21 158 объектов, оставленных в массиве после предобработки данных, 15 532 приходится на этажи с типом "другой", то есть не первый и не последний. Значит большая часть продаваемых квартир находится в зданиях с этажностью выше 3-х этажей. Квартир на последних этажах продается несколько больше, чем на первом.

Общее количество этажей в доме

In [66]:
```
data['floors_total'].hist()
plt.title('Количество этажей в доме')
plt.xlabel('этаж')
plt.ylabel('количество объявлений')
plt.show()
```



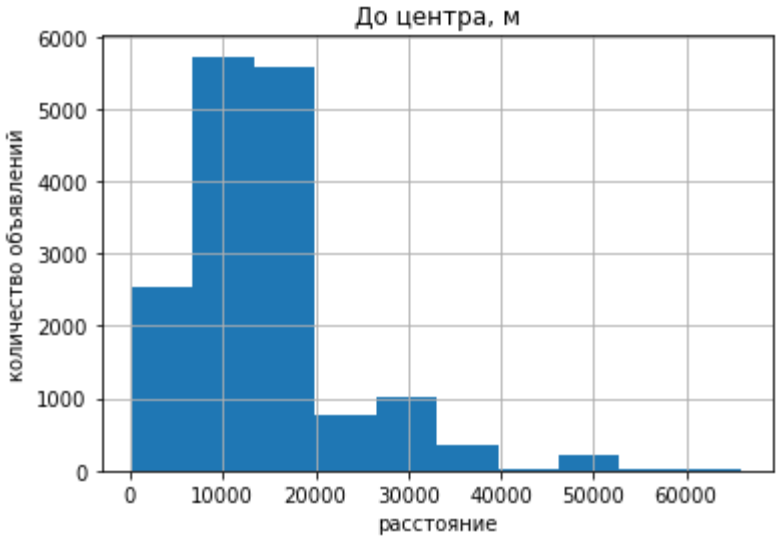
In [67]:
```
data['floors_total'].describe()
```

Out[67]:
```
count    21158.000000
mean        10.710086
std          6.598814
min          1.000000
25%          5.000000
50%          9.000000
75%         16.000000
max         60.000000
Name: floors_total, dtype: float64
```

Больше всего квартир продается 9-этажных домах, что объясняется наличием в выборке большого количества объектов из СПб. Много квартир продают в 5-этажках. Начиная с 11-12 этажа количество продаваемых квартир обратно пропорционально этажности зданий.

Расстояние до центра города в метрах

In [68]:
```
data['cityCenters_nearest'].hist()
plt.title('До центра, м')
plt.xlabel('расстояние')
plt.ylabel('количество объявлений')
plt.show()
```



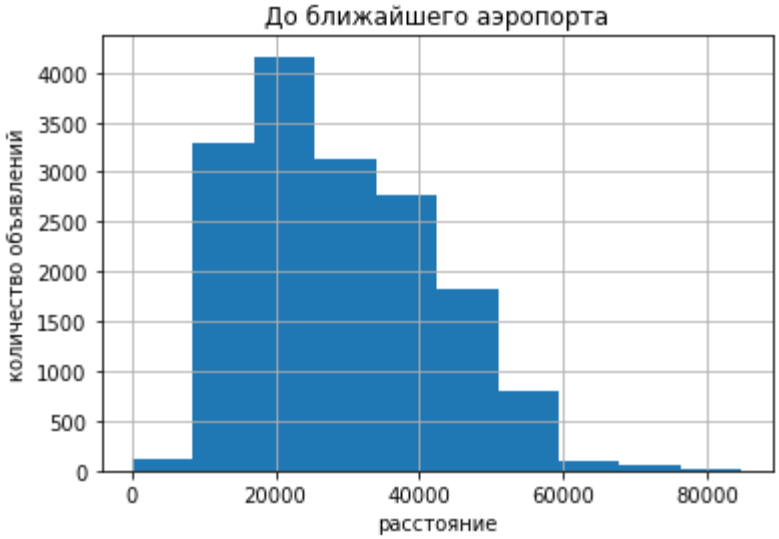
In [69]:
```
data['cityCenters_nearest'].describe()
```

Out[69]:
```
count    16274.000000
mean     14531.581971
std       8636.515173
min        208.000000
25%       9860.000000
50%      13277.000000
75%      16447.000000
max      65968.000000
Name: cityCenters_nearest, dtype: float64
```

Большая часть объектов находится на расстоянии от 10 до 17 тыс.м от центра. Самый ближний к центру объект на расстоянии 208 метров. 25% объектов на расстоянии от 200 до 10000 м. Самый удаленный от центра объект на расстоянии около 66000 м. Протяжённость Петербурга в административных границах: с севера на юг в пределах КАД — 32 км (за пределами КАД — 52 км), с северо-запада на юго-восток за пределами КАД, — около 90 км. Таким образом, географический центр Санкт-Петербурга находится в Финском заливе. Таким образом, можно предположить, что этот объект скорее всего находится в СПб на берегу Финского залива к северо-западу от центра города. Большинство объектов, для которых указаны расстояния в исследуемом массиве данных, находятся в СПб.

Расстояние до ближайшего аэропорта

In [70]:
```
data['airports_nearest'].hist()
plt.title('До ближайшего аэропорта')
plt.xlabel('расстояние')
plt.ylabel('количество объявлений')
plt.show()
```



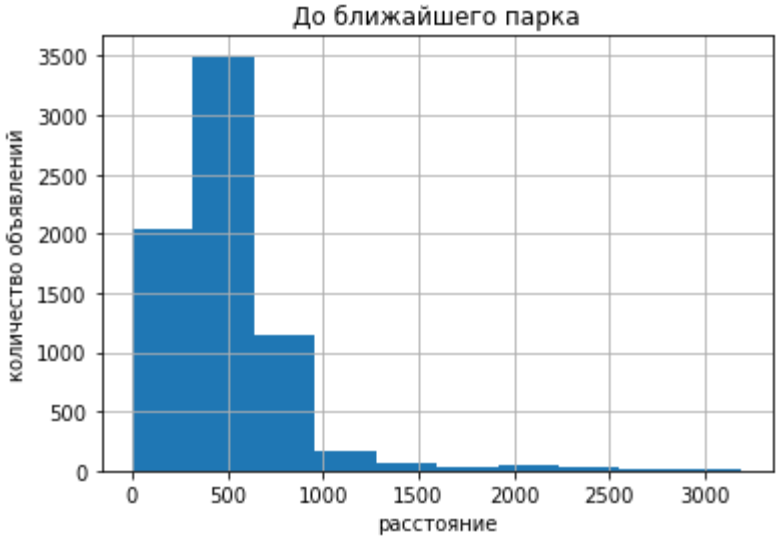
In [71]:
```
data['airports_nearest'].describe()
```

Out[71]:
```
count    16258.000000
mean     28827.114590
std      12803.415627
min          0.000000
25%      18398.000000
50%      26786.000000
75%      37412.000000
max      84869.000000
Name: airports_nearest, dtype: float64
```

Большая часть объектов находится на расстояни 18 - 40 тыс. м от аэропорта. Крупный гражданский аэропорт один и находится в Санкт-Петербурге - это Пулково. Большинство объектов в СПб и ближайших крупных населенных пунктах располагаются как раз на таком расстоянии от Пулково. Для более мелких и удаленных населенных пунктов в выборке данные не указаны.

Расстояние до ближайшего парка

In [72]:
```
data['parks_nearest'].hist()
plt.title('До ближайшего парка')
plt.xlabel('расстояние')
plt.ylabel('количество объявлений')
plt.show()
```



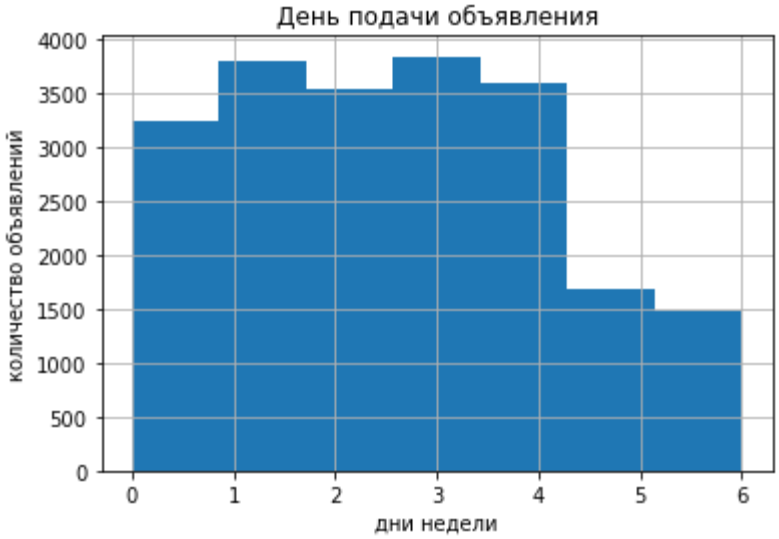
In [73]:
```
data['parks_nearest'].describe()
```

Out[73]:
```
count    7042.000000
mean      491.081369
std       337.086215
min         1.000000
25%       289.000000
50%       456.000000
75%       612.000000
max      3190.000000
Name: parks_nearest, dtype: float64
```

Расстояние до парка указано меньше, чем для половины объектов. Большинство из них находятся в Санкт-Петербурге и ближайших к нему городах, где парков довольно много. Поэтому среднее расстояние до парка около 500 метров. А минимальное вполне может быть равно 1 м.

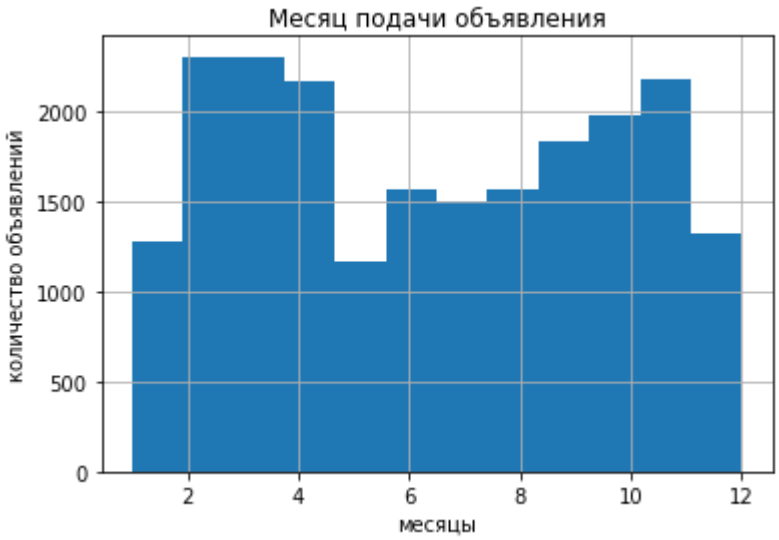
День и месяц публикации объявления

In [74]:
```
data['weekday'].hist(bins=7)
plt.title('День подачи объявления')
plt.xlabel('дни недели')
plt.ylabel('количество объявлений')
plt.show()
```



Видим, что чаще всего объявления о продаже подают по вторникам и четвергам, чуть реже по средам и пятницам, а меньше всего по суббота и воскресеньям.

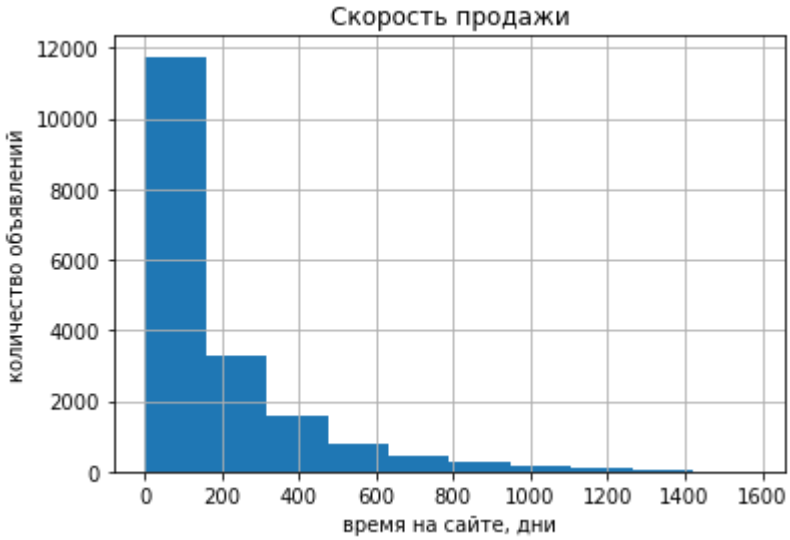
In [75]:
```
data['month'].hist(bins=12)
plt.title('Месяц подачи объявления')
plt.xlabel('месяцы')
plt.ylabel('количество объявлений')
plt.show()
```



По этой столбчатой диаграмме видно, что реже объявления подают в декабре и январе, что, видимо, связано с подготовкой к праздникам и выходными. Самый провальный месяц для публикаций - май, много выходных, праздников, начало дачных сезонов. А вот с февраля по апрель и в октябре-ноябре активность подачи объявлений заметно повышается.

Как быстро продавались квартиры

In [76]:
```
data['days_exposition'].hist()
plt.title('Скорость продажи')
plt.xlabel('время на сайте, дни')
plt.ylabel('количество объявлений')
plt.show()
```

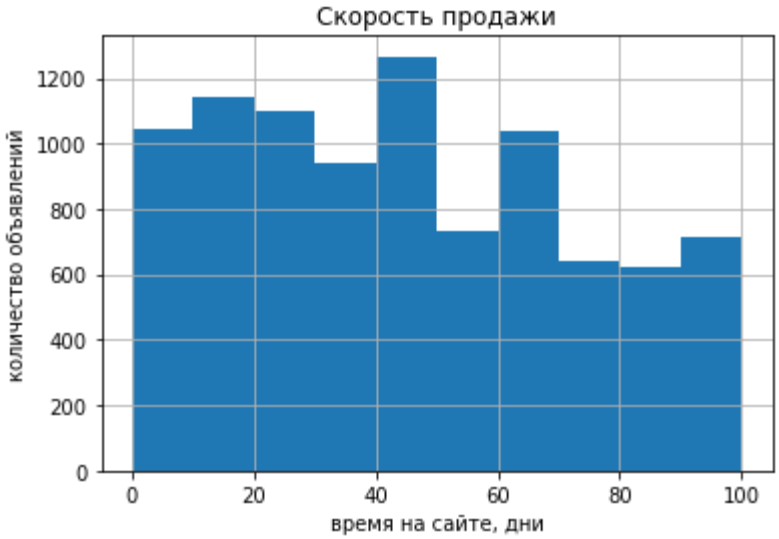


In [77]:
```
data['days_exposition'].describe().round()
```

Out[77]:
```
count    18347.0
mean       182.0
std        218.0
min          1.0
25%         44.0
50%         99.0
75%        234.0
max       1580.0
Name: days_exposition, dtype: float64
```

Большая часть объектов продается в провежутке от 44 до 234 дней с момента подачи объявления. Самые быстрые продажи занимают от 1 дня до полутора месяцев. Самые медленные тянутся от года до 4-х с чем-то лет. Разница между средним = 182 дня и медианой = 99 дней почти в 2 раза, так как очень велик максимум - 1580 дней.

In [78]:
```
data['days_exposition'].hist(range=(0, 100))
plt.title('Скорость продажи')
plt.xlabel('время на сайте, дни')
plt.ylabel('количество объявлений')
plt.show()
```

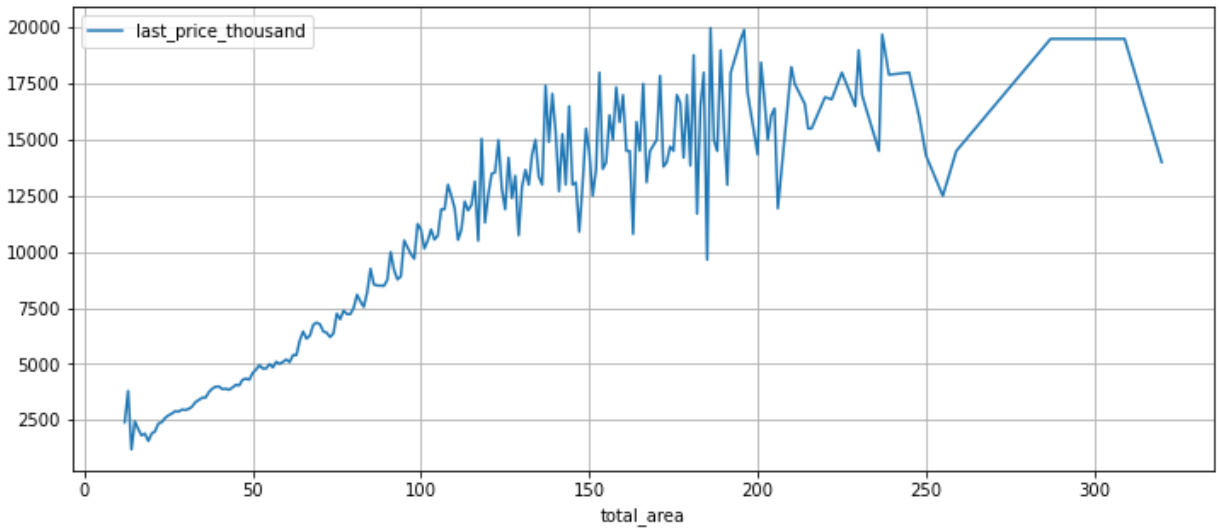


Пики на графике можно объяснить сроками размещения объявлений для СПб на сайте Яндекс Недвижимость в зависимости от вида объекта и типа сделки: продажа до 4.5 млн - 40 дней, от 4.5 до 10 млн - 60 дней, выше - 90 дней. Информация из источника: https://yandex.ru/support/realty/owner/home/add-ads-housing.html

Зависимость цены от различных факторов

Зависимость цены от общей площади

In [79]:
```
data.pivot_table(index='total_area', values='last_price_thousand', aggfunc='median').
plt.show()
```



In [80]:
```
data['last_price_thousand'].corr(data['total_area'])
```

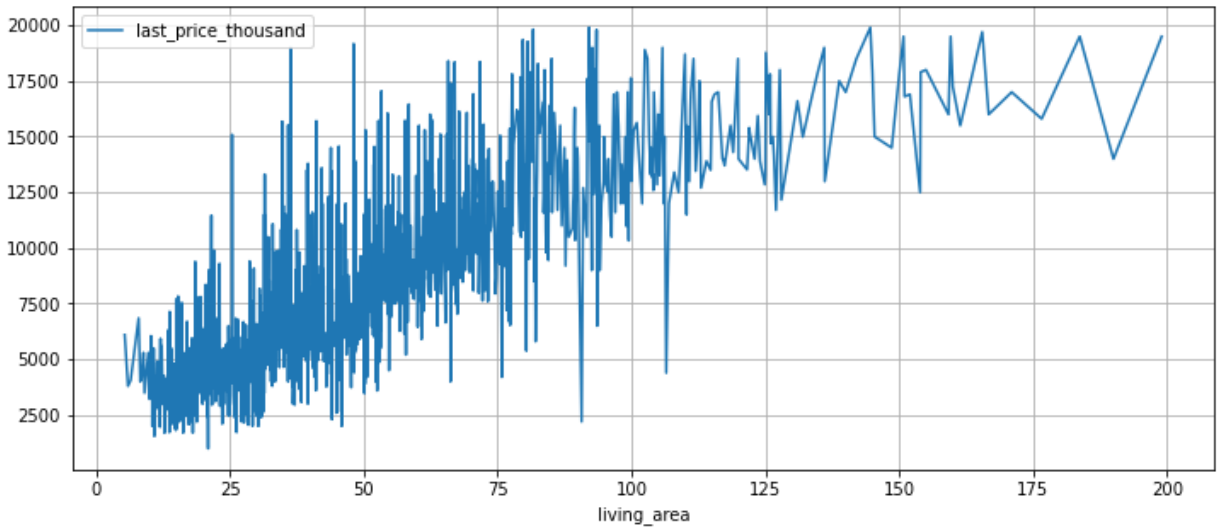
Out[80]: 0.7775878401796692

Обнаружили, что цена растет прямо пропорционально общей площади. Довольно большие пики на графике могут объясняться разницей цен на объекты равной площади в Санкт-Петербурге и малых населенных пунктах области (квартира 100 кв.м и дом в деревне

такой же площади).

Зависимость цены от жилой площади

In [81]:
```
data.pivot_table(index='living_area', values='last_price_thousand', aggfunc='median')
plt.show()
```



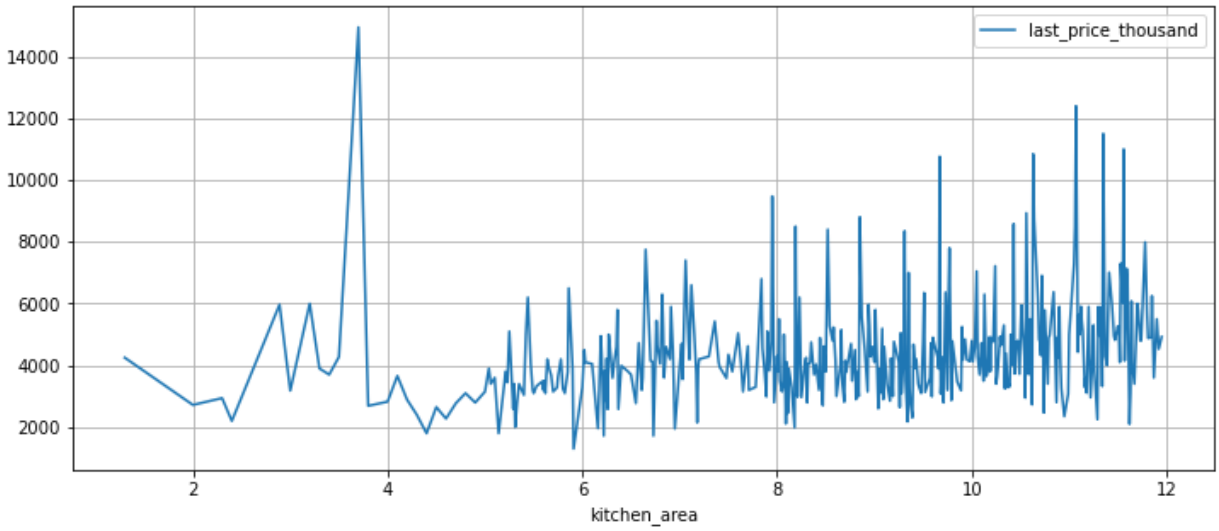
In [82]:
```
data['last_price_thousand'].corr(data['living_area'])
```

Out[82]:
```
0.6688630170579436
```

Рост цены в зависимости от жилой площади заметен, но выражен меньше, что доказывает и коэффициент корреляции. Большой разброс значений может объясняться разницей в общих площадях объектов с одинаковой жилой площадью. Наличием квартир студий и квартир свободной планировки. Цена объекта все же чаще рассчитывается по общей площади.

Зависимость цены от площади кухни

In [83]:
```
data.query('kitchen_area < 12').pivot_table(index='kitchen_area', values='last_price_
plt.show()
```



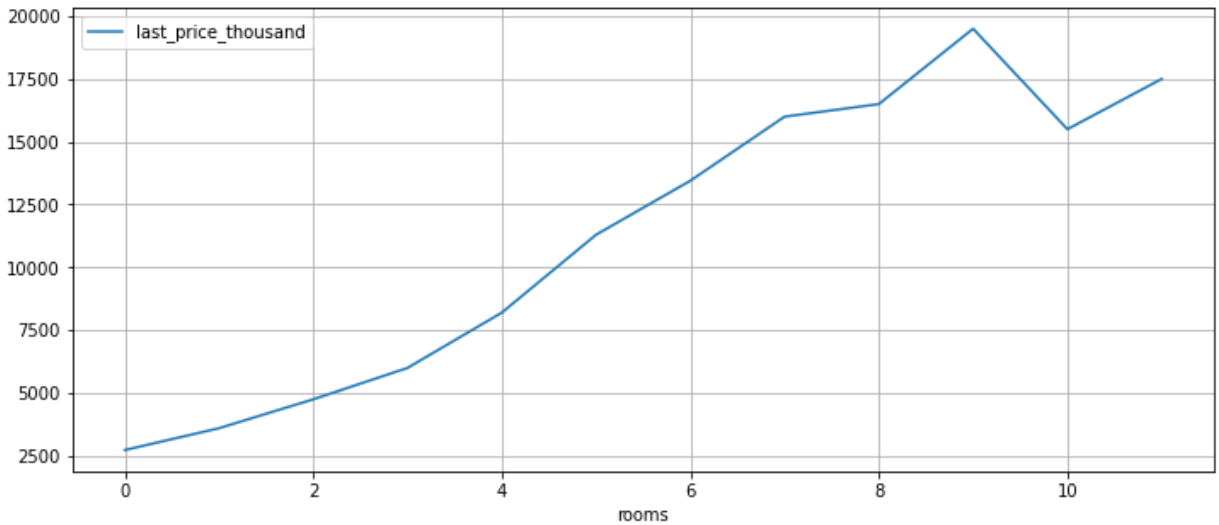
In [84]:
```
data['last_price_thousand'].corr(data['kitchen_area'])
```

Out[84]: 0.5763590475243308

Зависимость стоимости объекта от площади кухни выражена еще меньше, хотя все еще прямо пропорциональна. Для большей наглядности график строим для большинства объектов (75%) с площадью кухни до 12 кв.м.

Зависимость цены от количества комнат

In [85]:
```
data.pivot_table(index='rooms', values='last_price_thousand', aggfunc='median').plot(
plt.show()
```



In [86]:
```
data['last_price_thousand'].corr(data['rooms'])
```

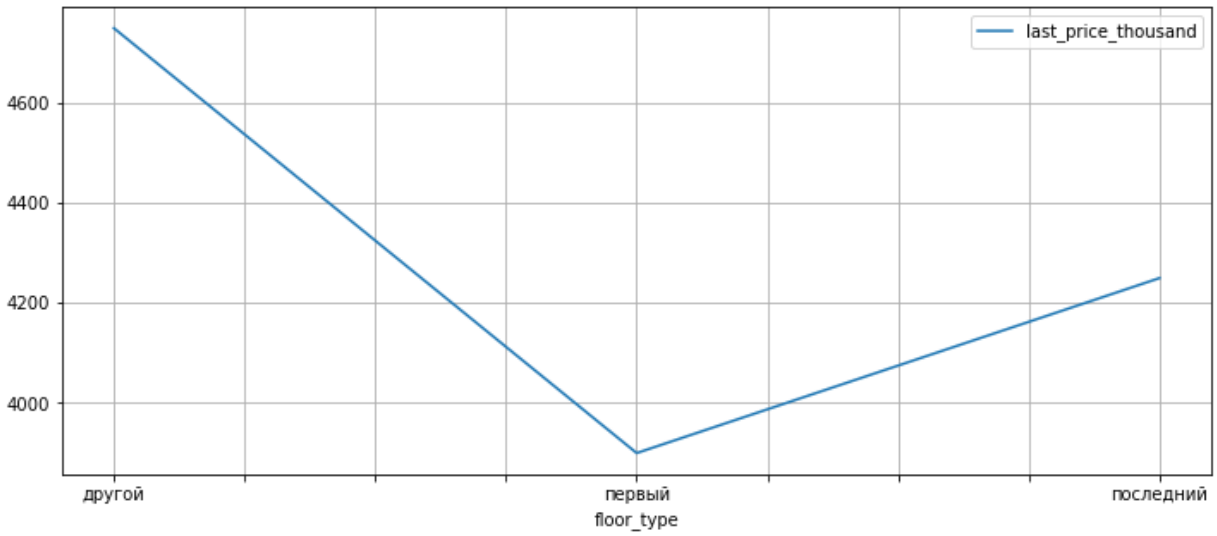
Out[86]: 0.5219201277123496

Зависимость цены от количества комнат растет прямо пропорционально, хотя наблюдается некоторый провал после 9 комнат. Возможно, количество комнат не всегда говорит о

большей площади, ведь есть очень маленькие трехкомнатные квартиры и довольно просторные студии такой же общей площади. Коэффициент корреляции подтверждает не самую сильную зависмость.

Зависимость цены от этажа, на котором расположена квартира (первый, последний, другой)

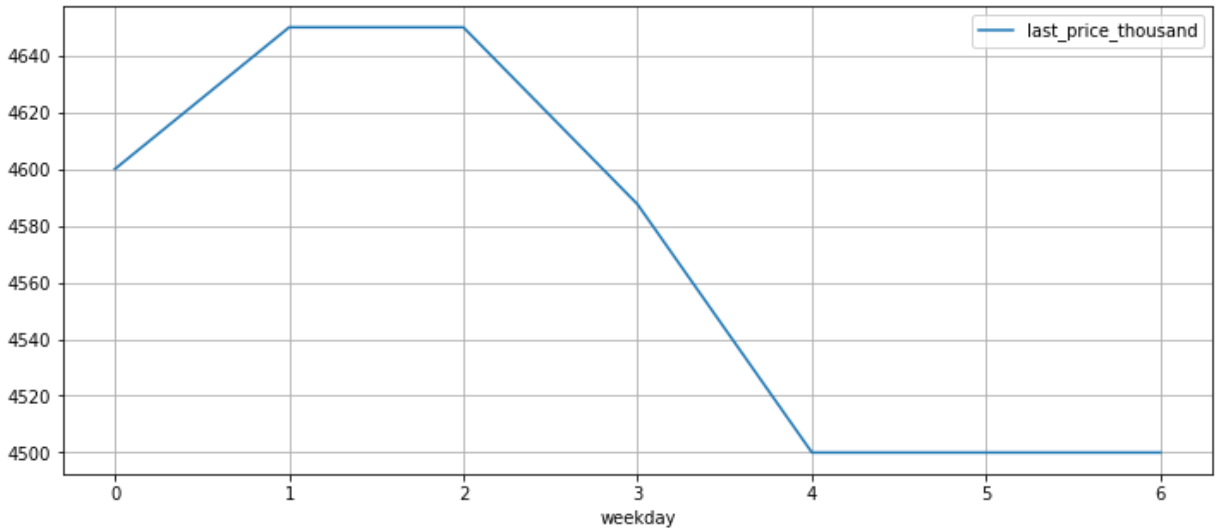
In [87]:
```
data.pivot_table(index='floor_type', values='last_price_thousand', aggfunc='median').
plt.show()
```



По графику видим провал цен на квартиры на 1-х этажах и явное предпочтение квартир не на последних этажах. Как известно, квартиры на последних этажах стоят дешевле. Но опять же, тут влияет наличие технического этажа, чердака. Как и квартира на 1 этаже в новом доме с цокольным этажом не будет такой же дешевой, как квартира на 1 этаже "брежневки". Опять же такое сравнение не очень корректно, т.к. в массиве есть данные коттеджных поселков, где большинство домов могут быть одноэтажными. И на стоимость дома это вряд ли повлияет. Этим, видимо, объясняется невысокий коэффициент корреляции.

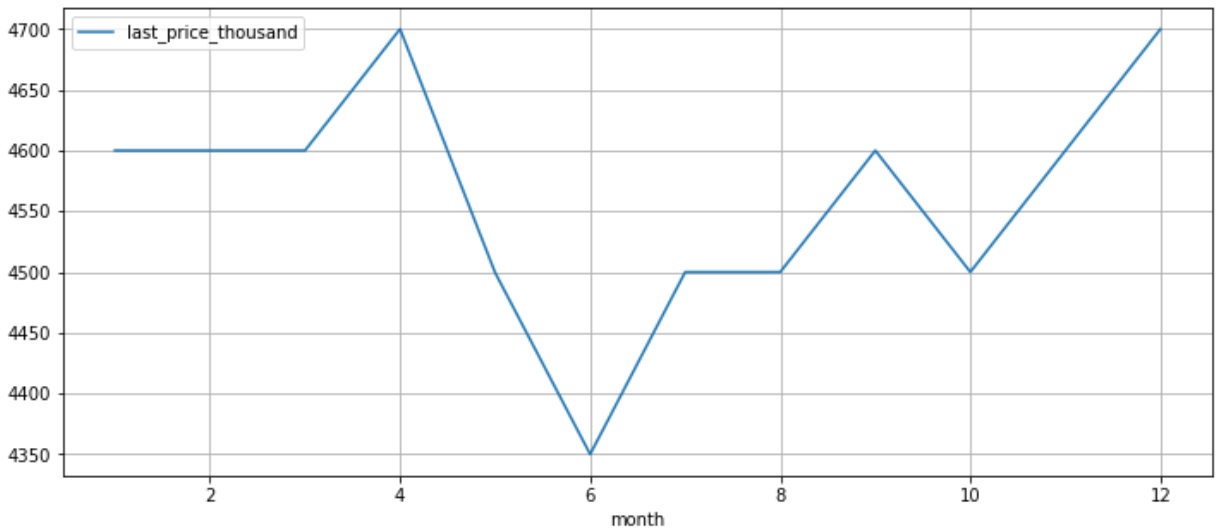
Зависимость цены от даты размещения (день недели, месяц, год)

In [88]:
```
data.pivot_table(index='weekday', values='last_price_thousand', aggfunc='median').plo
plt.show()
```



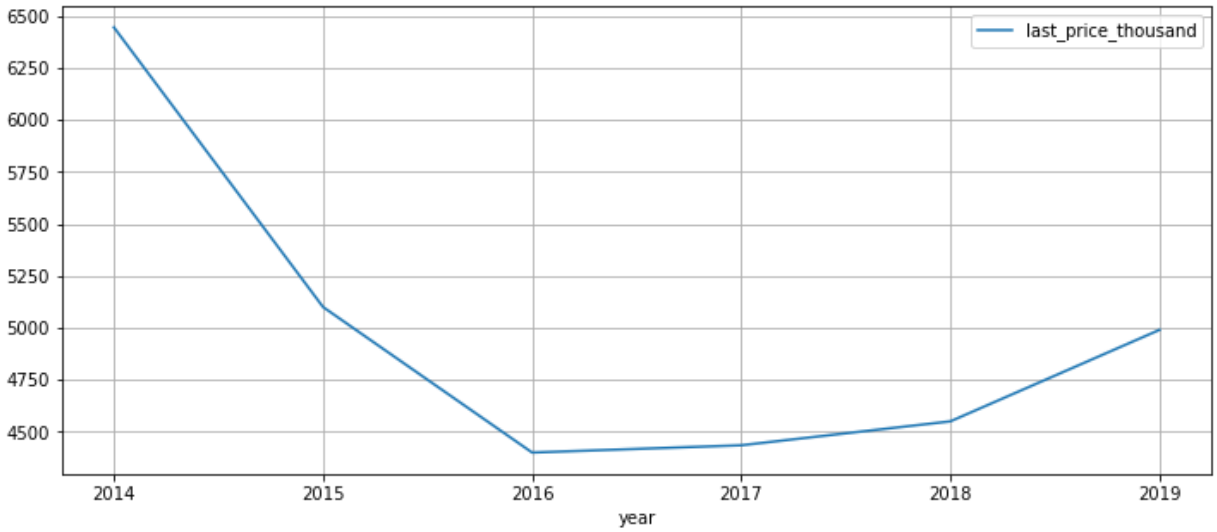
Пик цен на графике связан, видимо с тем, что наибольшее количество объявлений подается в середине рабочей недели вторник-среда.

In [89]:
```
data.pivot_table(index='month', values='last_price_thousand', aggfunc='median').plot(
plt.show()
```



Пики на графике связаны с количеством объявлений, подаваемых в каждом месяце. В конце весны - начале лета квартир продают меньше, чем в конце осени-начале весны. Провалы в месяцы начала дачного сезона - май, июнь. Рост к концу года и в апреле.

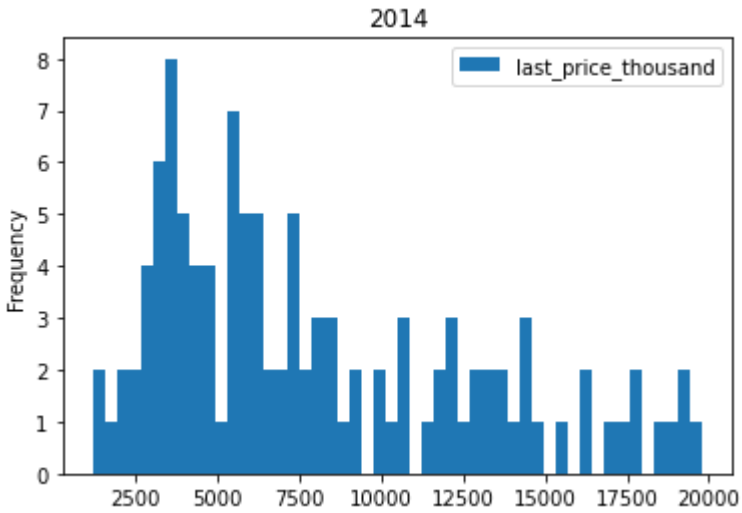
In [90]:
```
data.pivot_table(index='year', values='last_price_thousand', aggfunc='median').plot(g
plt.show()
```

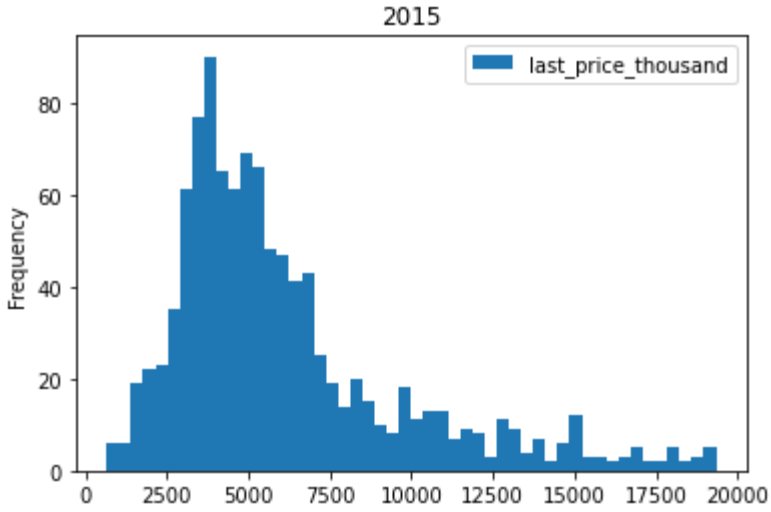


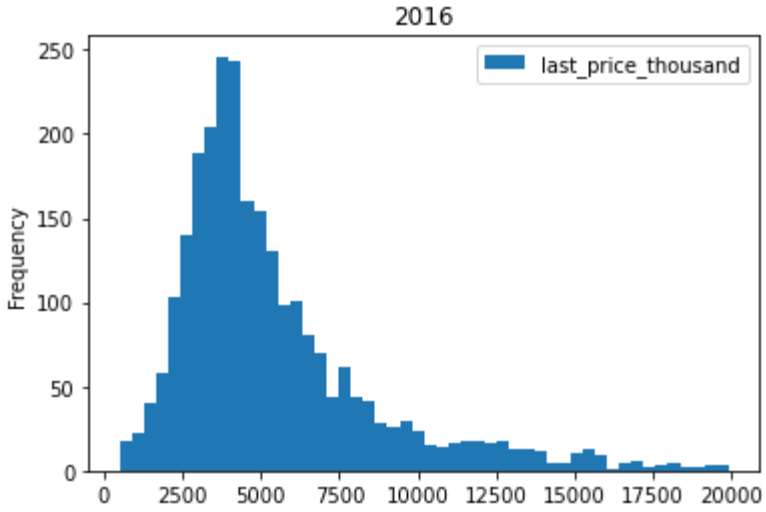
Падение средней цены на квартиру в 2016-2018 годах скорее всего связаны с последствиями кризиса 2014 года и резким снижением покупательной способности.

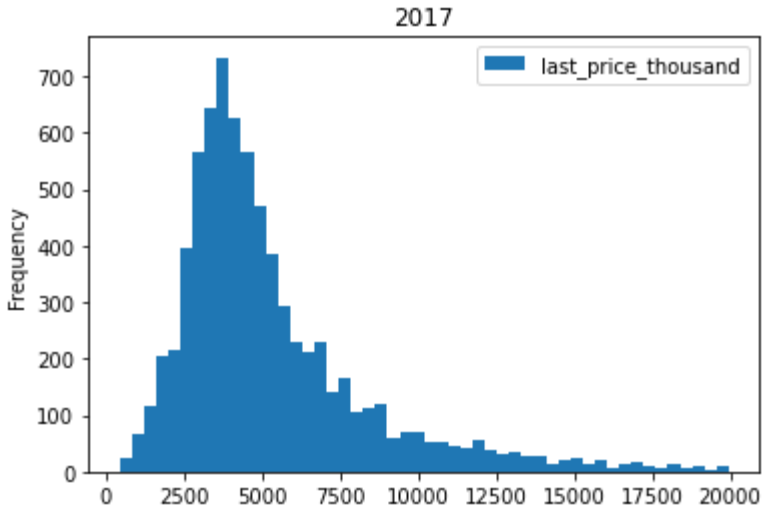
Сравним распределение цен продаваемых квартир по каждому году выборки:

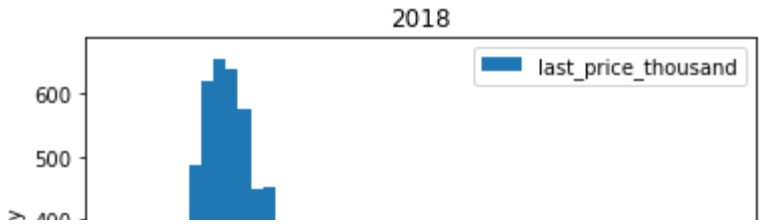
In [91]:
```
for year, group_data in data.groupby('year'):
    group_data.plot(y='last_price_thousand', title=year, kind='hist', bins=50)
```




2015
last_price_thousand
Frequency
0
20
40
60
80
0
2500
5000
7500
10000
12500
15000
17500
20000


2016
last_price_thousand
Frequency
0
50
100
150
200
250
0
2500
5000
7500
10000
12500
15000
17500
20000


2017
last_price_thousand
Frequency
0
100
200
300
400
500
600
700
0
2500
5000
7500
10000
12500
15000
17500
20000



По масштабу оси ординат видно, что количество продаваемых в разные годы объектов сильно отличается. В то время, как распределение цен примерно одинаково. Только в 2014 году наблюдаются странные провалы в предложении квартир определенных ценовых категорий, но это опять же может быть следствием кризиса.

Анализ цен по населенным пунктам

Посчитаем среднюю цену одного квадратного метра в 10 населённых пунктах с наибольшим числом объявлений.
Для этого найдем такие населенные пункты и подсчитаем, как часто они встречаются, так как каждое упоминание населенного пункта - это одно объявление.

In [92]:
```
data['locality_name'].value_counts().head(10)
```

Out[92]:
```
Санкт-Петербург      13962
поселок Мурино         491
поселок Шушары         404
Всеволожск             370
Пушкин                 337
Колпино                327
поселок Парголово      301
Гатчина                292
деревня Кудрово        262
Петергоф               195
Name: locality_name, dtype: int64
```

Создадим список - серию названий 10 населенных пунктов с наибольшим количеством объявлений.

In [93]:
```
locality_top = pd.Series(['Санкт-Петербург', 'поселок Мурино', 'поселок Шушары', 'Все
```

Построим сводную таблицу средних цен за 1 кв.м по населенным пунктам из созданного списка, упорядочим значения по убыванию и округлим.

In [94]:
```
price_top = data.query('locality_name in @locality_top').pivot_table(index='locality_
price_top
```

Out[94]:

	price_area
locality_name	
Санкт-Петербург	108766.0
Пушкин	102706.0
деревня Кудрово	93259.0

locality_name	price_area
поселок Парголово	90965.0
поселок Мурино	86898.0
Петергоф	84516.0
поселок Шушары	79293.0
Колпино	75555.0

Выделим населённые пункты с самой высокой и низкой стоимостью квадратного метра.

In [95]:
```
print('Населенный пункт с самой низкой стоимостью квадратного метра:', price_top.loc
```

```
Населенный пункт с самой низкой стоимостью квадратного метра:                price_ar
ea
locality_name
Всеволожск        67664.0
```

In [96]:
```
print('Населенный пункт с самой высокой стоимостью квадратного метра:', price_top.loc
```

```
Населенный пункт с самой высокой стоимостью квадратного метра:                  price
_area
locality_name
Санкт-Петербург    108766.0
```

Анализ изменения цены квадратного метра в Санкт-Петербурге для каждого километра по степени удалённости от центра

Для того, чтобы проанализировать изменение цены каждого километра по степени удаленности от центра, построим сводную таблицу для объектов Санкт-Петербурга. Данные отсортируем и округлим. Ограничим расстояние 10 км от центра.

In [97]:
```
data.query('locality_name == "Санкт-Петербург" and cityCenters_km <= 10').pivot_table
```

Out[97]:

cityCenters_km	last_price
0.0	10073333.0
1.0	10489862.0
2.0	10034816.0
3.0	9025293.0
4.0	9261302.0
5.0	9253431.0
6.0	8464931.0
7.0	7962239.0
8.0	7852299.0

last_price

cityCenters_km

Видим, что стоимость каждого километра уменьшается по мере удаления от центра. Чем ближе объект к центру, тем выше его цена. Посмотрим эту зависимость на графике:

In [98]:
```
data.query('locality_name == "Санкт-Петербург" and cityCenters_km <= 10').pivot_table
plt.title('Цена по мере удаленности от центра')
plt.xlabel('расстояние до центра в км')
plt.ylabel('цена объекта')
plt.show()
```

Пики и провалы цен на близком расстоянии можно объяснить особенностями районов расположения жилья. Даже вдали от центра может находиться элитный район с высокими ценами (например, район с коттеджами или таунхаусами, которые стоят больше, чем квартиры). Близко к центру может встретиться промышленная зона, рядом с которой цены будут ниже, чем в близлежащих районах.

Вывод:

Проведен исследовательский анализ следующих параметры объектов:

- общая площадь;
- жилая площадь;
- площадь кухни;
- цена объекта;
- количество комнат;
- высота потолков;
- этаж квартиры;
- тип этажа квартиры («первый», «последний», «другой»);
- общее количество этажей в доме;
- расстояние до центра города в метрах;
- расстояние до ближайшего аэропорта;

- расстояние до ближайшего парка;
- день и месяц публикации объявления.

Распределение данных в построенных гистограммах тяготеет к нормальному. На основании анализа можно выделить следующие средние характеристики наиболее часто продаваемого объекта: это двухкомнатная квартира общей площади около 50 кв.м, из них жилая площадь примерно 30 кв.м, площадь кухни 9 кв.м. Стоимость 4.5-5 млн. Высота потолка 2.6-2.7 метров. На 4 этаже 9-этажного дома. Расстояние до центра 13 300 м, до аэропорта 26 800 м, до парка 450 м. Объявление о продаже опубликовано в середине рабочей недели ранней весной или поздней осенью.

Большая часть объектов продается в промежутке от 44 до 234 дней с момента подачи объявления. Самые быстрые продажи занимают от 1 дня до полутора месяцев. Самые медленные тянутся от года до 4-х с чем-то лет.

Была исследована зависимость цены квартиры от:

- общей площади;
- жилой площади;
- площади кухни;
- количества комнат;
- этажа, на котором расположена квартира (первый, последний, другой);
- даты размещения (день недели, месяц, год).

Из всех этих параметров наиболее влияет на цену общая площадь помещения, чуть меньше жилая площадь (она чаще всего прямо пропорциональна жилой), тип этажа и площадь кухни. Цена растет с увеличением площади. Уменьшается для первого и последнего этажей. Зависимость цены от остальных параметров слабая, или должна учитывать влияние других факторов, не представленных в нашей выборке данных.

В ходе исследования найдены 10 населенных пунктов с наибольшим числом объявлений. Наибольшее число объявлений по Санкт-Петербургу, наименьшее из 10 по Петергофу. Как и следовало ожидать в этой десятке, населённый пункт с самой высокой стоимостью квадратного метра тоже Санкт-Петербург. Самая низкая цена квадратного метра во Всеволожске.

Для того, чтобы проанализировать изменение цены каждого километра по степени удаленности от центра, построена сводная таблица для объектов Санкт-Петербурга. Цена каждого километра уменьшается по мере удаления от центра.

Общий вывод

В ходе исследования архива объявлений о продаже квартир в Санкт-Петербурге и соседних населённых пунктов за несколько лет была проведена оценка размера массива данных, его характеристик, построены первоначальные гистограммы по всем видам данных, сделаны предварительные выводы.

Во время предобработки данных были устранены ошибки в измерениях, частично удалены пропуски в значениях некоторых параметров, устранены аномальные значения, изменены типы данных для проведения расчетов. Так же с целью проведения дальнейшего анализа в массиве были сведены в один столбец три столбца с типами объектов, добавлены столбцы с типом этажа, ценой кв.м, разбивкой даты на день, месяц и год публикации, измерено расстояние до центра в км.
Проведен исследовательский анализ различных параметров объектов, составлена характеристика наиболее часто продаваемого объекта. Сделаны выводы о набольшем влиянии на цену объекта его площади и расположения относительно центра,а также типа этажа, чуть меньше на цену влияет размер жилой площади и кухни, количество комнат.
Что касается исследований по дате публикации, подтверждается мнение о наибольшей активности на рынке недвижимости весной и осенью, затишье в праздники и летние месяцы, а также в мае. Падение средней цены на квартиру в 2016-2018 годах скорее всего связаны с последствиями кризиса 2014 года и резким снижением покупательной способности. Скорость продажи квартир в среднем от 1 месяца до года. Наиболее быстрые продажи случаются в срок от 1 дня до месяца, наблюдается рост продаж после одного и двух месяцев нахождения объявления на сайте, наиболее долгие продажи тянутся аж до 4 лет.
Были выделены 10 населенных пунктов с наибольшим количеством публикаций о продаже. Самым активным, как и самым дорогим по стоимости квадратного метра является Санкт-Петербург. Наименьшая цена квадратного метра из городов с наибольшим количеством объявлений - во Всеволожске.
Для Санкт-Петербурга создана сводная таблица изменения цены на каждый километр удаления от центра. Как и ожидалось, по мере удаления цена падает.

Для чистоты исследования предлагаю в дальнейшем разделить данные по объектам Санкт-Петербурга и остальных населенных пунктов в виду несопоставимой разницы цен и характеристик объектов. Если провести исследование по районам Петербурга, до более существенным окажется влияние количества парков и прудов. В то время как в какой-нибудь деревне эти характеристики вообще не имеют значения. Так же могут быть выявлены отдельные районы Питера, удаленные от центра, но равные ему по цене, благодаря совокупности других характеристик жилья и местности. Жилье в Санкт-Петербурге логично было бы делить на категории - есть существенная разница между ценой, площадью, высотой потолков и т.п. общежитий и элитных новостроек.