Тема - Исследование объявлений о продаже квартир

Содержание

- 1 Откройте файл с данными и изучите общую информацию.
  - 1.1 Первое знакомство с данными
  - 1.2 Проверим данные на явные дубликаты
- 2 Предобработка данных
  - 2.1 Приведем значение даты публикации в формат datetime для удобной работы с датой в дальнейшем исследовании
  - 2.2 Определим количество пустых столбцов в нашем датафрейме
  - 2.3 Посмотрим уникальные значения столбца `balcony`
  - 2.4 Проверим процент пропусков в столбце с количеством этажей - `floors_total`
  - 2.5 Для выяснения причин пропусков в поле `is_apartment` сгруппируем данные по этому столбцу
  - 2.6 Проверим, что если в поле `ponds_around3000` и `parks_around3000` есть пропуски, то и в полях `parks_nearest` и `ponds_nearest` пропущены значения, либо расстояние больше 3000 м.
  - 2.7 Займемся пропусками в столбце `locality_name`
  - 2.8 Займемся пропусками в столбце `days_exposition`
  - 2.9 Посмотрим на тип данных в столбцах
    - 2.9.1 Изучим столбец с названиями населенных пунктов `locality_name`
  - 2.10 Посмотрим на уникальные значения столбца `ceiling_height` (высота потолков)
  - 2.11 Посмотрим на уникальные значения столбца `days_exposition`
  - 2.12 Посмотрим , есть ли дубликаты в таблице
  - 2.13 Посмотрим еще раз на столбцы с пропусками
- 3 Посчитайте и добавьте в таблицу новые столбцы
  - 3.1 Добавим столбец с ценой 1 кв.м. `1m_price` в наш датасет
  - 3.2 Добавим столбец с днем публикации обьявления `ad_day_of_week`
    - 3.2.1 Добавим столбец с месяцем и годом публикации обьявления `ad_month` и `ad_year`
  - 3.3 Создадим функцию для столбца `floor_type` - будет 3 значения - first, last , other
  - 3.4 Создадим столбец `cityCenters_nearest_km` для расчета целого расстояния до центра в километрах
- 4 Проведите исследовательский анализ данных
  - 4.1 Изучим параметры объектов
    - 4.1.1 Посмотрим на параметры еще раз, начнем с Общей площади - `total_area`
    - 4.1.2 Параметр - жилая площадь - `living_area`
    - 4.1.3 Параметр - площадь кухни - `kitchen_area`
    - 4.1.4 Параметр - цена объекта - `last_price`
    - 4.1.5 Параметр - количество комнат - `rooms`
    - 4.1.6 Параметр - высота потолков - `ceiling_height`
    - 4.1.7 Параметр - этаж квартиры - `floor`
    - 4.1.8 Параметр - тип этажа квартиры - `floor_type`

        - 4.1.9 Параметр - Общее количество этажей в доме - `floors_total`
        - 4.1.10 Параметр - Расстояние до центра города в метрах - `cityCenters_nearest`
        - 4.1.11 Параметр - Расстояние до ближайшего аэропорта - `airports_nearest`
        - 4.1.12 Параметр - Расстояние до ближайшего парка - `parks_nearest`
        - 4.1.13 Параметр - День публикации обьявления - `ad_day_of_week`
        - 4.1.14 Параметр - месяц публикации обьявления - `ad_month`
    - 4.2 Изучим, как быстро продавались квартиры- `days_exposition` .
    - 4.3 Изучим влияние различных факторов на общую (полную) стоимость Объекта
    - 4.4 Посчитайте среднюю цену одного квадратного метра в 10 населённых пунктах с наибольшим числом объявлений. Выделите населённые пункты с самой высокой и низкой стоимостью квадратного метра. Эти данные можно найти по имени в столбце `locality_name.`
    - 4.5 Ранее мы посчитали расстояние до центра в километрах `cityCenters_nearest_km` .
- 5 Общий вывод

Описание проекта

В вашем распоряжении данные сервиса Яндекс.Недвижимость — архив объявлений о продаже квартир в Санкт-Петербурге и соседних населённых пунктов за несколько лет. Нужно научиться определять рыночную стоимость объектов недвижимости. Ваша задача — установить параметры. Это позволит 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.

Откройте файл с данными и изучите общую информацию.

In [1]:
```
import pandas as pd
import re
from datetime import datetime
import matplotlib.pyplot as plt
```

In [2]:
```
try:
    data = pd.read_csv('/datasets/real_estate_data.csv',sep='\t')
except:
    data = pd.read_csv('real_estate_data.csv',sep="\t")
```

Ознакомимся с Датасетом -

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

Out[3]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07T00:00:00	3	2.70	16.0	51.0	8	
1	7	3350000.0	40.4	2018-12-04T00:00:00	1	NaN	11.0	18.6	1	
2	10	5196000.0	56.0	2015-08-20T00:00:00	2	NaN	5.0	34.3	4	

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
3	0	64900000.0	159.0	2015-07-24T00:00:00	3	NaN	14.0	NaN	9	
4	2	10000000.0	100.0	2018-06-19T00:00:00	2	3.03	14.0	32.0	13	

5 rows × 22 columns

In [4]:
```
data.shape
```

Out[4]:
```
(23699, 22)
```

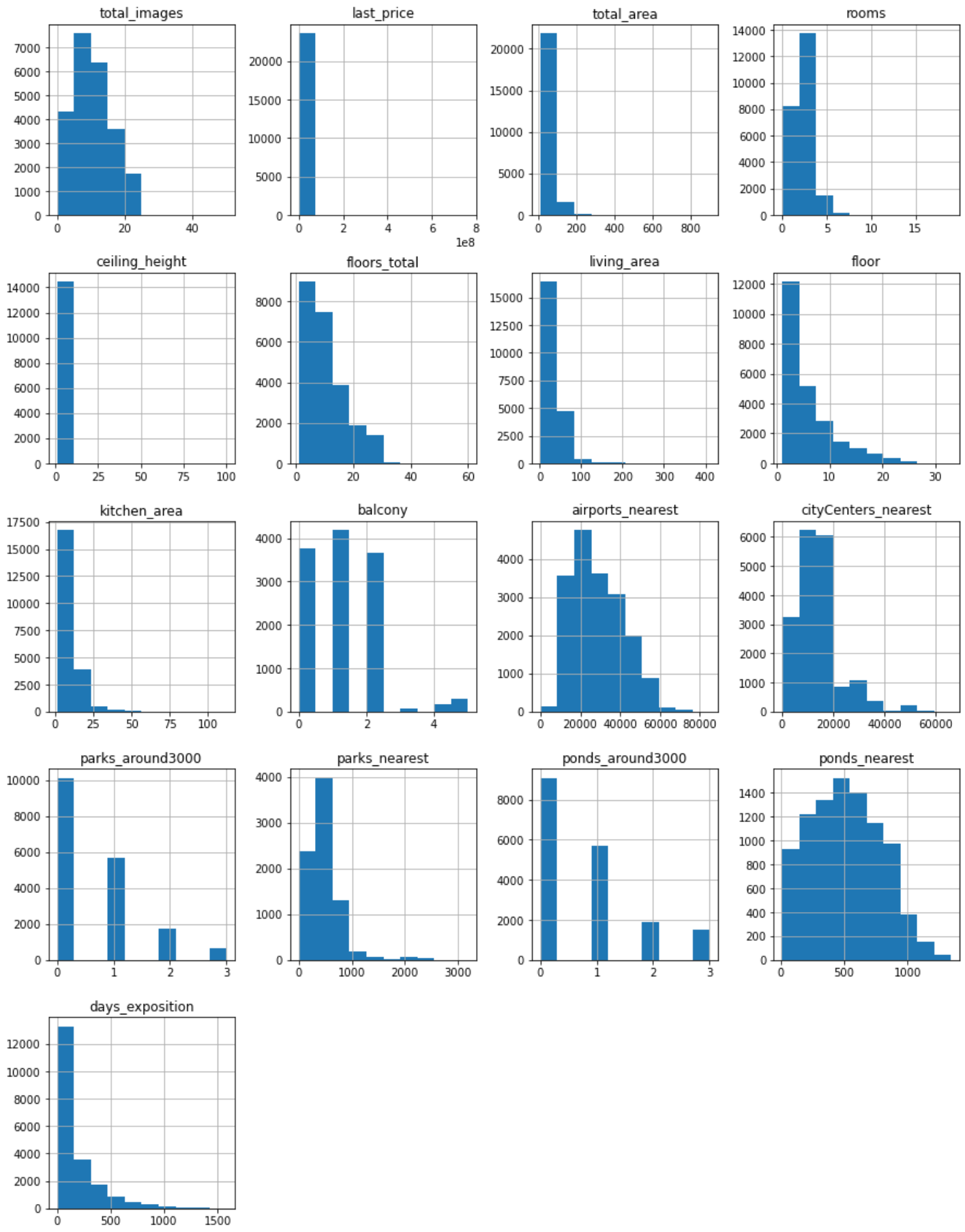
In [5]:
```
data.info()
```

```
<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)
memory usage: 3.7+ MB
```

Первое знакомство с данными

Согласно заданию, построим общую гистограмму для всех числовых столбцов таблицы.

In [6]:
```
ax = data.hist(figsize=(15, 20))
```



На первый взгляд, в нашем датасете присутствуют большая выборка обьявлений с квартирами различного класса, но основная масса прихолится на квартиры со следующими параметрами:

- Общая площадь до 100 кв.м.
- Количество комнат 1-4
- Находящиеся в домах этажностью до 30 этажей

- Жилой площадь. до 90 кв.м
- Этаж, на котором находится недвижимость - до 11
- Общей площадью кухни до 20 кв.м.
- Квартиры с одним, двумя балконами и без балконов
- Находящиеся на расстоянии от 20 до 60 км до ближайшего аэропорта
- Находящиеся в 10-20 км от центра города
- Рядом с квартирой от 0 до 2х парках в радиусе 3км
- Рядом с квартирой от 0 до 3 водоемов в радиусе 3км

Основная масса квартир была продана в срок до 250 дней.

Проверим данные на явные дубликаты

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

Out[7]:
```
0
```

Явных дубликатов в нашем датасете - нет.

Предобработка данных

Приведем значение даты публикации в формат datetime для удобной работы с датой в дальнейшем исследовании

In [8]:
```
print(data['first_day_exposition'].head())
```

```
0    2019-03-07T00:00:00
1    2018-12-04T00:00:00
2    2015-08-20T00:00:00
3    2015-07-24T00:00:00
4    2018-06-19T00:00:00
Name: first_day_exposition, dtype: object
```

In [9]:
```
data['first_day_exposition'] = pd.to_datetime(data['first_day_exposition'],format='%Y-%m-%
print(data['first_day_exposition'].head())
```

```
0   2019-03-07
1   2018-12-04
2   2015-08-20
3   2015-07-24
4   2018-06-19
Name: first_day_exposition, dtype: datetime64[ns]
```

Определим количество пустых столбцов в нашем датафрейме

In [10]:
```
data.isna().sum()
```

Out[10]:
```
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 [11]:
```
pd.DataFrame(round(data.isna().mean()*100,1)).sort_values(0,ascending=False).style.backgrc
```

Out[11]:

	0
is_apartment	88.300000
parks_nearest	65.900000
ponds_nearest	61.600000
balcony	48.600000
ceiling_height	38.800000
airports_nearest	23.400000
ponds_around3000	23.300000
parks_around3000	23.300000
cityCenters_nearest	23.300000
days_exposition	13.400000
kitchen_area	9.600000
living_area	8.000000
floors_total	0.400000
locality_name	0.200000
total_images	0.000000
last_price	0.000000
studio	0.000000
floor	0.000000
rooms	0.000000
first_day_exposition	0.000000
total_area	0.000000
open_plan	0.000000

Как мы видим, пропущенные значения есть в следующих столбцах:

- `is_apartment` - апартаменты (булев тип) - 88% пропусков
- `parks_nearest` — расстояние до ближайшего парка (м) - 66% пропусков
- `ponds_nearest` — расстояние до ближайшего водоёма (м) - 62% пропусков
- `balcony` — число балконов - 49% пропусков
- `ceiling_height` — высота потолков (м) - 39% пропусков

- `airports_nearest` — расстояние до ближайшего аэропорта в метрах (м) - 23% пропусков
- `cityCenters_nearest` - расстояние до центра города (м) - 23% пропусков
- `parks_around3000` — число парков в радиусе 3 км - 23% пропусков
- `ponds_around3000` — число водоёмов в радиусе 3 км - 23% пропусков
- `days_exposition` — сколько дней было размещено объявление (от публикации до снятия) - 13% пропусков
- `kitchen_area` — площадь кухни в квадратных метрах (м^2) - 9.6% пропусков
- `living_area` — жилая площадь в квадратных метрах (м^2) - 8% пропусков
- `floors_total` — всего этажей в доме - 0.3% пропусков
- `locality_name` - название населённого пункта - 0.2% пропусков

Посмотрим уникальные значения столбца `balcony`

In [12]:
```
data['balcony'].unique()
```

Out[12]:
```
array([nan,  2.,  0.,  1.,  5.,  4.,  3.])
```

Будем считать, что если продавец не указал число балконов, значит их нет. Заполним пропуски - нулями.

In [13]:
```
data['balcony'] = data['balcony'].fillna(0)
```

Проверим, что пропусков в столбце `balcony` больше нет

In [14]:
```
data['balcony'].isna().sum()
```

Out[14]:
```
0
```

Проверим процент пропусков в столбце с количеством этажей - `floors_total`

In [15]:
```
data['floors_total'].isna().mean()
```

Out[15]:
```
0.0036288450989493226
```

Пропуски составляют менее 1% от общего числа. Удалим данные датафрейма с пропусками в столбце `floors_total` .

In [16]:
```
data = data[data['floors_total'].notna()]
```

Проверим, что избавились от пропусков

In [17]:
```
data['floors_total'].isna().mean()
```

Out[17]:
```
0.0
```

Для выяснения причин пропусков в поле `is_apartment` сгруппируем данные по этому столбцу

In [18]:
```
data.groupby('is_apartment')['is_apartment'].count()
```

Out[18]:
```
is_apartment
False     2725
```

```
True       50
Name: is_apartment, dtype: int64
```

Как мы видим, большая часть объектов не является аппартаментами. Будем считать, что пропуски так же не являются аппратаментами. К тому же , в нашем исследовании данный столбец не критичен.

In [19]:
```
data['is_apartment']=data['is_apartment'].fillna(False)
```

Проверим, что избавились от пропусков

In [20]:
```
data['is_apartment'].isna().mean()
```

Out[20]:
```
0.0
```

Проверим, что если в поле `ponds_around3000` и `parks_around3000` есть пропуски, то и в полях `parks_nearest` и `ponds_nearest` пропущены значения, либо расстояние больше 3000 м.

Осуществим проверку

In [21]:
```
data[data['ponds_around3000'].isna()]['ponds_nearest'].unique()
```

Out[21]:
```
array([nan])
```

In [22]:
```
data[data['parks_around3000'].isna()]['parks_nearest'].unique()
```

Out[22]:
```
array([nan])
```

Наша гипотеза подтвердилась, можем смело указать значение **0** в пропуска `ponds_around3000` и `parks_around3000`

In [23]:
```
data['ponds_around3000'] = data['ponds_around3000'].fillna(0)
data['parks_around3000'] = data['parks_around3000'].fillna(0)
```

Проверим, что избавились от пропусков

In [24]:
```
data.isna().mean().sort_values(ascending=False)
```

Out[24]:
```
parks_nearest           0.659298
ponds_nearest           0.616271
ceiling_height          0.386143
airports_nearest        0.234278
cityCenters_nearest     0.233304
days_exposition         0.134333
kitchen_area            0.094482
living_area             0.079194
locality_name           0.002033
ponds_around3000        0.000000
parks_around3000        0.000000
balcony                 0.000000
total_images            0.000000
last_price              0.000000
studio                  0.000000
is_apartment            0.000000
floor                   0.000000
floors_total            0.000000
rooms                   0.000000
```

```
first_day_exposition    0.000000
total_area              0.000000
open_plan               0.000000
dtype: float64
```

Займемся пропусками в столбце `locality_name`

In [25]:
```
data['locality_name'].isna().mean()
```

Out[25]:
```
0.002032778554186253
```

Количество пропусков менее 0,3 процента - Запишем во все пропуски значение - **неизвестно**

In [26]:
```
data['locality_name'] = data['locality_name'].fillna('неизвестен')
```

Займемся пропусками в столбце `days_exposition`

In [27]:
```
data['days_exposition'].isna().mean()
```

Out[27]:
```
0.13433278278914157
```

Как мы видим, **количество пропущенных значений в столбце с датой снятия объявления около 13%.**

Можем предположить, что **данные квартиры до сих пор не продались**.

Значит мы можем **посчитать разницу между текущей датой и датой публикации обьявления**

Но чтобы не исказить данные, добавим для квартир, в которых есть дата публикации и время обьявление значение `sold = True` - тем самым показав, что квартиры проданы. А для квартир в которых пропущен параметр `days_exposition` поставим значение `sold= False`

In [28]:
```
data.loc[data['days_exposition'].notnull(),'sold'] = True
data['sold']=data['sold'].fillna(False)
```

In [29]:
```
data.head()
```

Out[29]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16.0	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	NaN	11.0	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	NaN	5.0	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	NaN	14.0	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14.0	32.0	13	

5 rows × 23 columns

Проставим для всех столбцов со значением `sold=False` значение `days_exposition` равное разницы между текущей датой и датой публикования обьявления

In [30]:
```
data.loc[data['sold']==False,'days_exposition']=(datetime.now()-data['first_day_expositio
```

In [31]:
```
data.head()
```

Out[31]:

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16.0	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	NaN	11.0	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	NaN	5.0	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	NaN	14.0	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14.0	32.0	13	

5 rows × 23 columns

Остальные столбцы с пропусками не являются важными в данном исследовании или количество пропусков в них не так критично, и чтобы не исказить данные, трогать их пока не будем

In [32]:
```
data.isna().mean().sort_values(ascending=False).head(8)
```

Out[32]:
```
parks_nearest          0.659298
ponds_nearest          0.616271
ceiling_height         0.386143
airports_nearest       0.234278
cityCenters_nearest    0.233304
kitchen_area           0.094482
living_area            0.079194
total_images           0.000000
dtype: float64
```

Посмотрим на тип данных в столбцах

In [33]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23613 entries, 0 to 23698
Data columns (total 23 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   total_images          23613 non-null  int64         
 1   last_price            23613 non-null  float64       
 2   total_area            23613 non-null  float64       
 3   first_day_exposition  23613 non-null  datetime64[ns]
 4   rooms                 23613 non-null  int64         
 5   ceiling_height        14495 non-null  float64       
 6   floors_total          23613 non-null  float64       
 7   living_area           21743 non-null  float64       
 8   floor                 23613 non-null  int64         
 9   is_apartment          23613 non-null  bool          
 10  studio                23613 non-null  bool          
 11  open_plan             23613 non-null  bool          
 12  kitchen_area          21382 non-null  float64       
```

```
 13  balcony               23613 non-null  float64
 14  locality_name         23613 non-null  object
 15  airports_nearest      18081 non-null  float64
 16  cityCenters_nearest   18104 non-null  float64
 17  parks_around3000      23613 non-null  float64
 18  parks_nearest         8045 non-null   float64
 19  ponds_around3000      23613 non-null  float64
 20  ponds_nearest         9061 non-null   float64
 21  days_exposition       23613 non-null  float64
 22  sold                  23613 non-null  bool
dtypes: bool(4), datetime64[ns](1), float64(14), int64(3), object(1)
memory usage: 4.2+ MB
```

Переведем значение столбцов `floors_total` , `balcony` , `parks_around3000` , `ponds_around3000` , `days_exposition` в формат `int` (так как значения этих данных не может быть не целым числом)

In [34]:
```
data['floors_total']=data['floors_total'].astype('int')
data['balcony']=data['balcony'].astype('int')
data['parks_around3000']=data['parks_around3000'].astype('int')
data['ponds_around3000']=data['ponds_around3000'].astype('int')
data['days_exposition']=data['days_exposition'].astype('int')
```

Изучим столбец с названиями населенных пунктов `locality_name`

Переведем значение столбца в текстовое(string)

In [35]:
```
data['locality_name']=data['locality_name'].astype('string')
```

Проверим уникальные значения столбца `locality_name`

In [36]:
```
data['locality_name'].unique()
```

Out[36]:
```
<StringArray>
[          'Санкт-Петербург',            'посёлок Шушары',
 'городской посёлок Янино-1',         'посёлок Парголово',
            'посёлок Мурино',                 'Ломоносов',
                 'Сертолово',                  'Петергоф',
                   'Пушкин',            'деревня Кудрово',
 ...
   'деревня Большое Рейзино',    'деревня Малая Романовка',
       'поселок Дружноселье',             'поселок Пчевжа',
       'поселок Володарское',             'деревня Нижняя',
 'коттеджный посёлок Лесное',         'деревня Тихковицы',
    'деревня Борисова Грива',      'посёлок Дзержинского']
Length: 365, dtype: string
```

Как мы видим, встречаются дубликаты. Попробуем их убрать.

- приведем все имена к нижнему регистру

In [37]:
```
data['locality_name'] = data['locality_name'].str.lower()
data['locality_name'].unique()
```

Out[37]:
```
<StringArray>
[          'санкт-петербург',            'посёлок шушары',
 'городской посёлок янино-1',         'посёлок парголово',
            'посёлок мурино',                 'ломоносов',
                 'сертолово',                  'петергоф',
                   'пушкин',            'деревня кудрово',
 ...
   'деревня большое рейзино',    'деревня малая романовка',
       'поселок дружноселье',             'поселок пчевжа',
```

```
       'поселок володарское',            'деревня нижняя',
 'коттеджный посёлок лесное',          'деревня тихковицы',
    'деревня борисова грива',       'посёлок дзержинского']
Length: 365, dtype: string
```

- Для замены буквы Е на Ё используем регулярные выражения

In [38]:
```
data['locality_name']=data['locality_name'].str.replace(r'ё','е',regex=False)
```

Заменим неочевидные дубликаты -

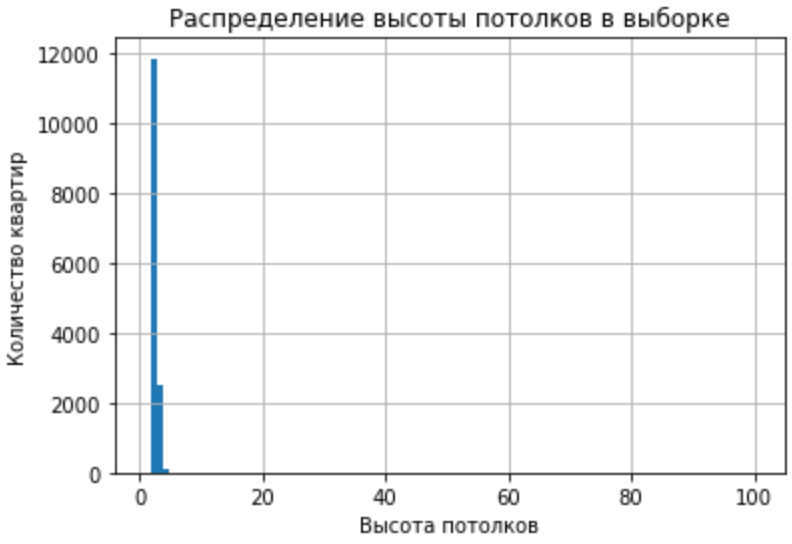
In [39]:
```
data['locality_name'] = data['locality_name'].replace('поселок городского типа рябово','по
data['locality_name'] = data['locality_name'].replace('городской поселок большая ижора','г
data['locality_name'] = data['locality_name'].replace('городской поселок мга','поселок гор
data['locality_name'] = data['locality_name'].replace('городской поселок павлово','поселок
data['locality_name'] = data['locality_name'].replace('городской поселок лесогорский','пос
data['locality_name'] = data['locality_name'].replace('городской поселок назия','поселок н
data['locality_name'] = data['locality_name'].replace('городской поселок рощино','поселок
```

In [40]:
```
data['locality_name'].unique()
```

Out[40]:
```
<StringArray>
[          'санкт-петербург',            'поселок шушары',
 'городской поселок янино-1',         'поселок парголово',
            'поселок мурино',                 'ломоносов',
                 'сертолово',                  'петергоф',
                    'пушкин',           'деревня кудрово',
 ...
   'деревня большое рейзино',   'деревня малая романовка',
       'поселок дружноселье',            'поселок пчевжа',
       'поселок володарское',            'деревня нижняя',
 'коттеджный поселок лесное',          'деревня тихковицы',
    'деревня борисова грива',       'поселок дзержинского']
Length: 324, dtype: string
```

Посмотрим на уникальные значения столбца `ceiling_height` (высота потолков)

In [41]:
```
data.hist(column='ceiling_height', bins=100)
plt.xlabel('Высота потолков')
plt.ylabel('Количество квартир')
plt.title('Распределение высоты потолков в выборке')
plt.show()
```



Как мы видим, в основном высота потолков составляет от 2 до 4 метров, есть много аномальных данных, и потолков выше 10-15 метров. Скорее всего ошибка при заполнении (не там стоит точка). Исправим.

In [42]:
```
data[data['ceiling_height'] >10]['ceiling_height'].unique()
```

Out[42]:
```
array([ 25. ,  32. ,  27. ,  24. ,  26. ,  14. ,  20. ,  22.6,  27.5,
        10.3, 100. ])
```

In [43]:
```
data['ceiling_height'] = data['ceiling_height'].replace(25,2.5)
data['ceiling_height'] = data['ceiling_height'].replace(32,3.2)
data['ceiling_height'] = data['ceiling_height'].replace(27,2.7)
data['ceiling_height'] = data['ceiling_height'].replace(24,2.4)
data['ceiling_height'] = data['ceiling_height'].replace(26,2.6)
data['ceiling_height'] = data['ceiling_height'].replace(20,2)
data['ceiling_height'] = data['ceiling_height'].replace(22.6,2.26)
data['ceiling_height'] = data['ceiling_height'].replace(27.5,2.75)
data['ceiling_height'] = data['ceiling_height'].replace(100,10)
data['ceiling_height'] = data['ceiling_height'].replace(14,1.4)
```

Посмотрим количество незаполненных строк в графе `ceiling_height`

In [44]:
```
data['ceiling_height'].isna().mean()
```

Out[44]:
```
0.3861432261889637
```

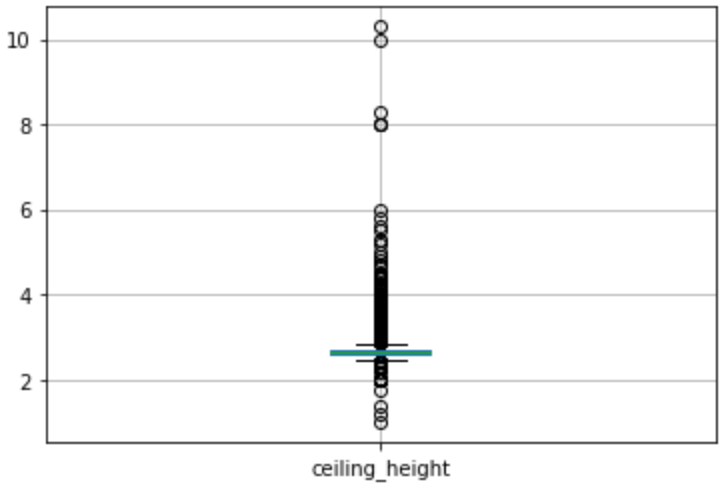
Количество пропусков велико и составляет ~ 39%. В связи с тем, что на необходимо провести исследование на влияния высоты потолков на успешную продажу, добавим в пустые стобцы - медианное значение высоты.

In [45]:
```
data['ceiling_height']= data['ceiling_height'].fillna(data['ceiling_height'].median())
data['ceiling_height'].isna().mean()
```

Out[45]:
```
0.0
```

In [46]:
```
data.boxplot(column='ceiling_height')
```

Out[46]:
```
<AxesSubplot:>
```



In [47]:
```
data.query('ceiling_height < 2 or ceiling_height > 6').count()
```

Out[47]:
```
total_images            10
```

```
last_price              10
total_area              10
first_day_exposition    10
rooms                   10
ceiling_height          10
floors_total            10
living_area             10
floor                   10
is_apartment            10
studio                  10
open_plan               10
kitchen_area             8
balcony                 10
locality_name           10
airports_nearest         8
cityCenters_nearest      8
parks_around3000        10
parks_nearest            4
ponds_around3000        10
ponds_nearest            5
days_exposition         10
sold                    10
dtype: int64
```

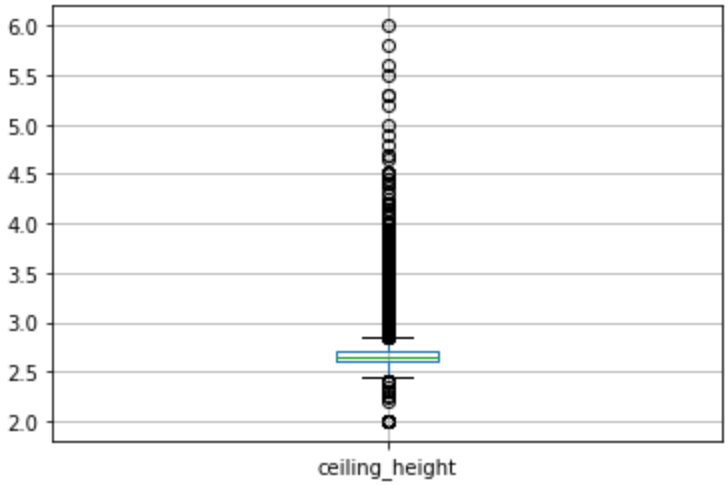
Объектов с высотой потолков больше 6 метров или меньше 2 - всего 10.

Оставим объекты недвижимости высотой потолков от 2 до 6 метров.

In [48]:
```
data = data.query('ceiling_height >= 2 and ceiling_height <= 6')
```

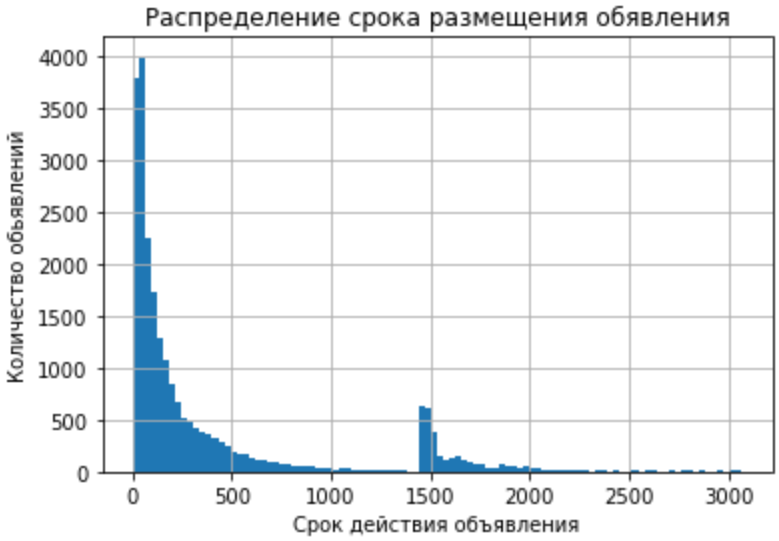
In [49]:
```
data.boxplot(column='ceiling_height')
```

Out[49]:
```
<AxesSubplot:>
```



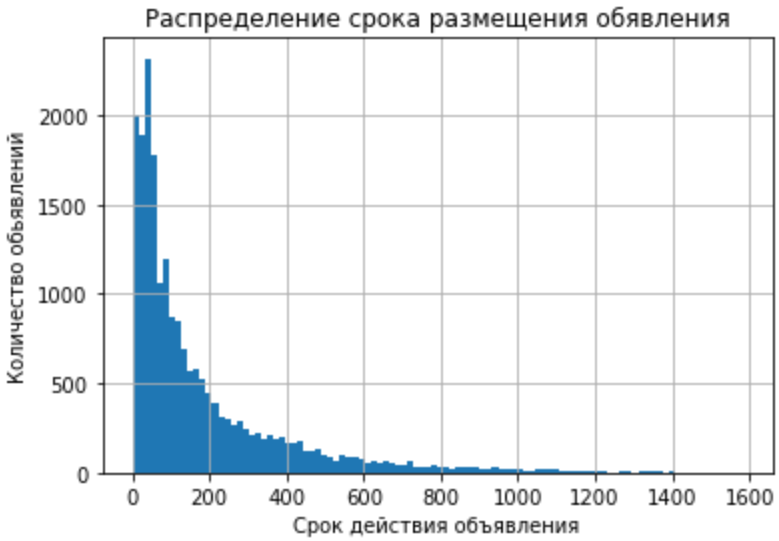
Посмотрим на уникальные значения столбца `days_exposition`

In [50]:
```
data['days_exposition'].hist( bins=100)
plt.xlabel('Срок действия объявления')
plt.ylabel('Количество обьявлений')
plt.title('Распределение срока размещения обявления')
plt.show()
```



In [51]:

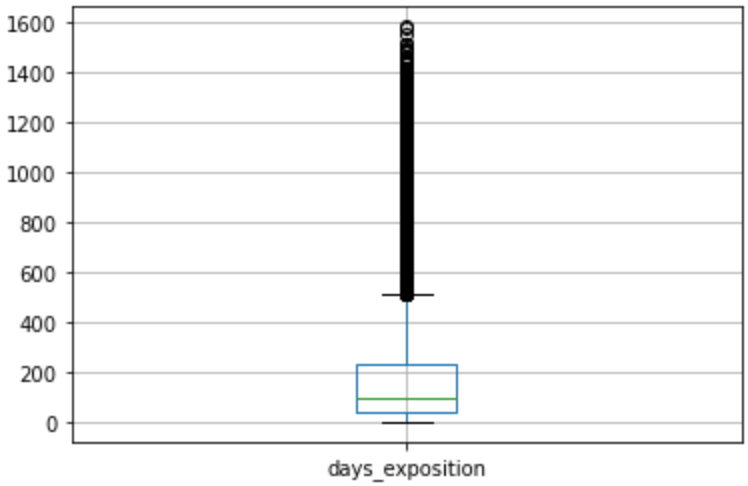
```
data[data['sold']==True]['days_exposition'].hist( bins=100)
plt.xlabel('Срок действия объявления')
plt.ylabel('Количество обьявлений')
plt.title('Распределение срока размещения обявления')
plt.show()
```



In [52]:

```
data[data['sold']==True].boxplot(column='days_exposition')
print(data[data['sold']==True]['days_exposition'].describe())
```

```
count    20431.000000
mean       180.881161
std        219.706903
min          1.000000
25%         45.000000
50%         95.000000
75%        232.000000
max       1580.000000
Name: days_exposition, dtype: float64
```



Как мы видим, большинство квартир продалось в промежутке от 45 до 232 дней. Второй пик на первой гистограме связан с тем, что все пропуски в столбце `days_exposition` мы заменили текущей датой и считаем, что что квартира до сих пор не продалась и объявление активно. Мы имели порядка 13% пропусков.

Посмотрим , есть ли дубликаты в таблице

In [53]:
```
data.duplicated().sum()
```

Out[53]:
```
0
```

Явных дубликатов не обнаружено - можно приступать к дальнейшему Анализу.

Посмотрим еще раз на столбцы с пропусками

In [54]:
```
data.isna().mean().sort_values(ascending = False).head(8)
```

Out[54]:
```
parks_nearest          0.659323
ponds_nearest          0.616320
airports_nearest       0.234292
cityCenters_nearest    0.233318
kitchen_area           0.094437
living_area            0.079227
total_images           0.000000
days_exposition        0.000000
dtype: float64
```

Оставшиеся пропуски найдены в следующих столбцах -

- `parks_nearest` - расстояние до ближайшего парка (м) -65% пропусков
- `ponds_nearest` -расстояние до ближайшего пруда (м)- 62% пропусков
- `ceiling_height` - высота потолков - 39% пропусков
- `airports_nearest` - расстояние до ближайшего аэропорта (м)- 23% пропусков
- `cityCenters_nearest` - расстояние до центра города (м) - 23% пропусков
- `kitchen_area` - площадь кухни - 9% пропусков
- `living_area` - жилая площадь - 8% пропусков

- Судя по пункта 2.6 исследования пропущенные значения в столбцах `parks_nearest` и `ponds_nearest` говорят о том, что в ближайших 3 км - прудов и парков - нет. Ставить значение 3,4,5 км вместо пустого - я считаю - некорректно. Может исказить итоговый вывод.

- Параметр высота потолков `ceiling_height` можно было бы поменять на медианный, но это тоже бы сильно повлияло на результаты итогового исследования.

- Параметр - расстояние от ближайшего аэропорта - мы можем бы заменить медианным значением по каждому населенному пункту.

In [55]:
```
for name in data['locality_name'].unique():
    median = data.loc[data['locality_name'] == name]['airports_nearest'].median()
    data.loc[(data['airports_nearest'].isna()) & (data['locality_name'] == name), 'airport

data['airports_nearest'].isna().mean()
```

Out[55]:
```
0.2300555014193111
```

Но как мы видим, **есть много населенных пунктов, в которых не указано расстояние до аэропорта**(мы могли бы вручную проставить, но это потребует дополнительных трудозатрат)

- Количество пустых значений в столбцах жилой площади и площади кухни составляет менее 10%. Чтобы не исказить данные оставим эти пропуски.

- Значение параметра `cityCenters_nearest` - я тоже не стал менять, чтобы не исказить данные. Можно было бы проставить расстояние в зависимости от цены недвижимости (например как мы в дальнейшем рассчитаем для Санкт-Петербурга в шаге 4.5). Но это потребует дополнительных временных затрат и для данного исследования не так критично.

Посчитайте и добавьте в таблицу новые столбцы

Добавим столбец с ценой 1 кв.м. `1m_price` в наш датасет

In [56]:
```
data['1m_price'] =data['last_price']/data['total_area']
```

Выведем 5 строк с новым столбцом

In [57]:
```
display(data.head())
```

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	32.0	13	

5 rows × 24 columns

Добавим столбец с днем публикации обьявления `ad_day_of_week`

In [58]:
```
data['ad_day_of_week'] = data['first_day_exposition'].dt.weekday
display(data.head())
```

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	32.0	13	

5 rows × 25 columns

Добавим столбец с месяцем и годом публикации обьявления `ad_month` и `ad_year`

In [59]:
```
data['ad_month'] = data['first_day_exposition'].dt.month
data['ad_year'] = data['first_day_exposition'].dt.year
display(data.head())
```

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	32.0	13	

5 rows × 27 columns

Создадим функцию для столбца `floor_type` - будет 3 значения - first, last , other

In [60]:
```
def floor_type(df):
    if df['floor']==1:
        return 'first'
    elif df['floor']==df['floors_total']:
        return 'last'
    else:
        return 'other'
```

Создадим столбец, применив к датафрейму функцию `floor_type`

In [61]:
```
data['floor_type']=data.apply(floor_type,axis=1)
display(data.head())
```

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	NaN	9	

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	32.0	13	

5 rows × 28 columns

Создадим столбец `cityCenters_nearest_km` для расчета целого расстояния до центра в километрах

In [62]:
```
data['cityCenters_nearest_km'] = (data['cityCenters_nearest']/1000).round()
display(data.head())
```

	total_images	last_price	total_area	first_day_exposition	rooms	ceiling_height	floors_total	living_area	floor	is
0	20	13000000.0	108.0	2019-03-07	3	2.70	16	51.0	8	
1	7	3350000.0	40.4	2018-12-04	1	2.65	11	18.6	1	
2	10	5196000.0	56.0	2015-08-20	2	2.65	5	34.3	4	
3	0	64900000.0	159.0	2015-07-24	3	2.65	14	NaN	9	
4	2	10000000.0	100.0	2018-06-19	2	3.03	14	32.0	13	

5 rows × 29 columns

В итоге мы добавили следующие столбцы, которые пригодятся в дальнейшем исследовании -

- `cityCenters_nearest_km` - расстояние до центра в километрах
- `floor_type` - тип этажа (первый, последний, другой)
- `ad_month` - месяц размещения обьявления
- `ad_day_of_week` - день недели (от 0 до 6) размещения обьявления
- `ad_year` - год размещения объявления
- `1m_price` - цена за 1 кв.м.

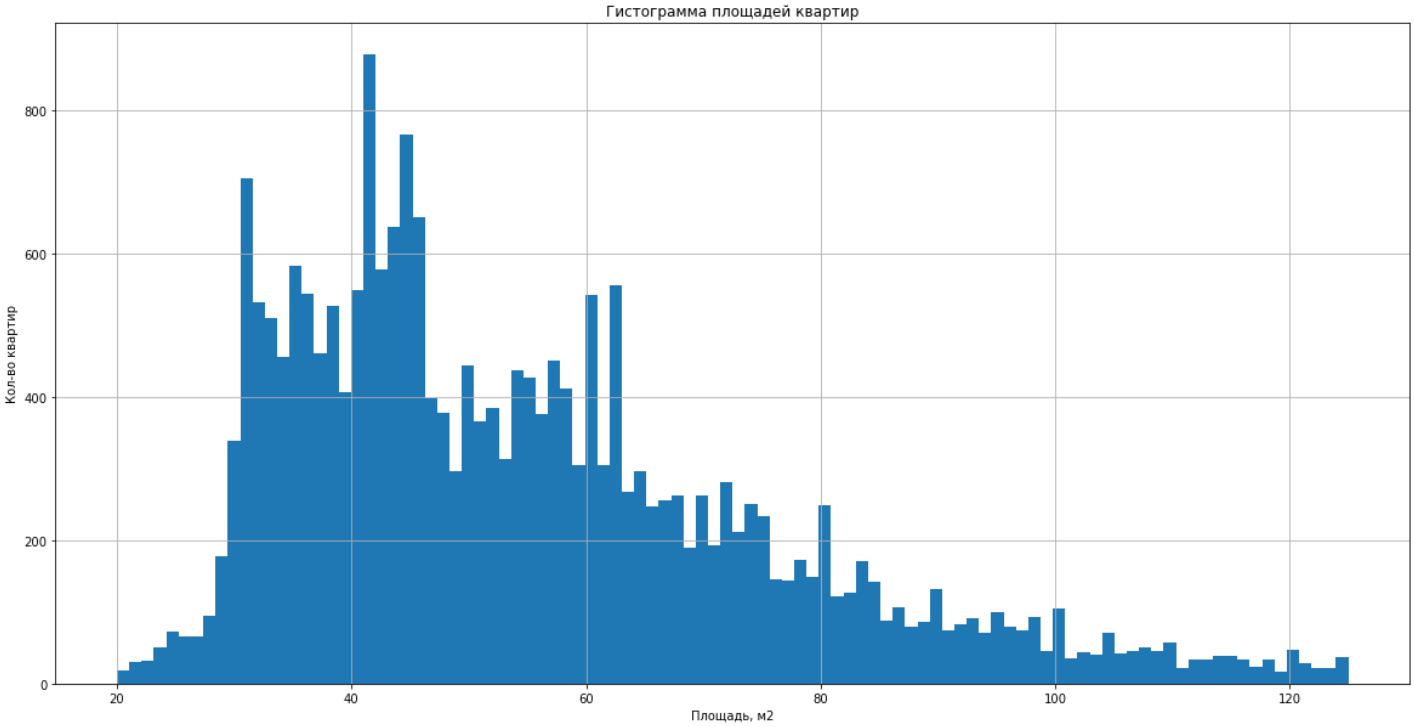
Проведите исследовательский анализ данных

Изучим параметры объектов

Посмотрим на параметры еще раз, начнем с Общей площади - `total_area`

In [63]:
```
ax = (
    data['total_area']
    .hist(bins=100, range=(20, 125), figsize=(20,10))
    .set(title = 'Гистограмма площадей квартир', xlabel = 'Площадь, м2', ylabel = 'Кол-во
    )
data['total_area'].describe()
```

Out[63]:
```
count    23603.000000
mean        60.346746
std         35.654465
min         12.000000
25%         40.000000
50%         52.000000
75%         69.800000
max        900.000000
Name: total_area, dtype: float64
```



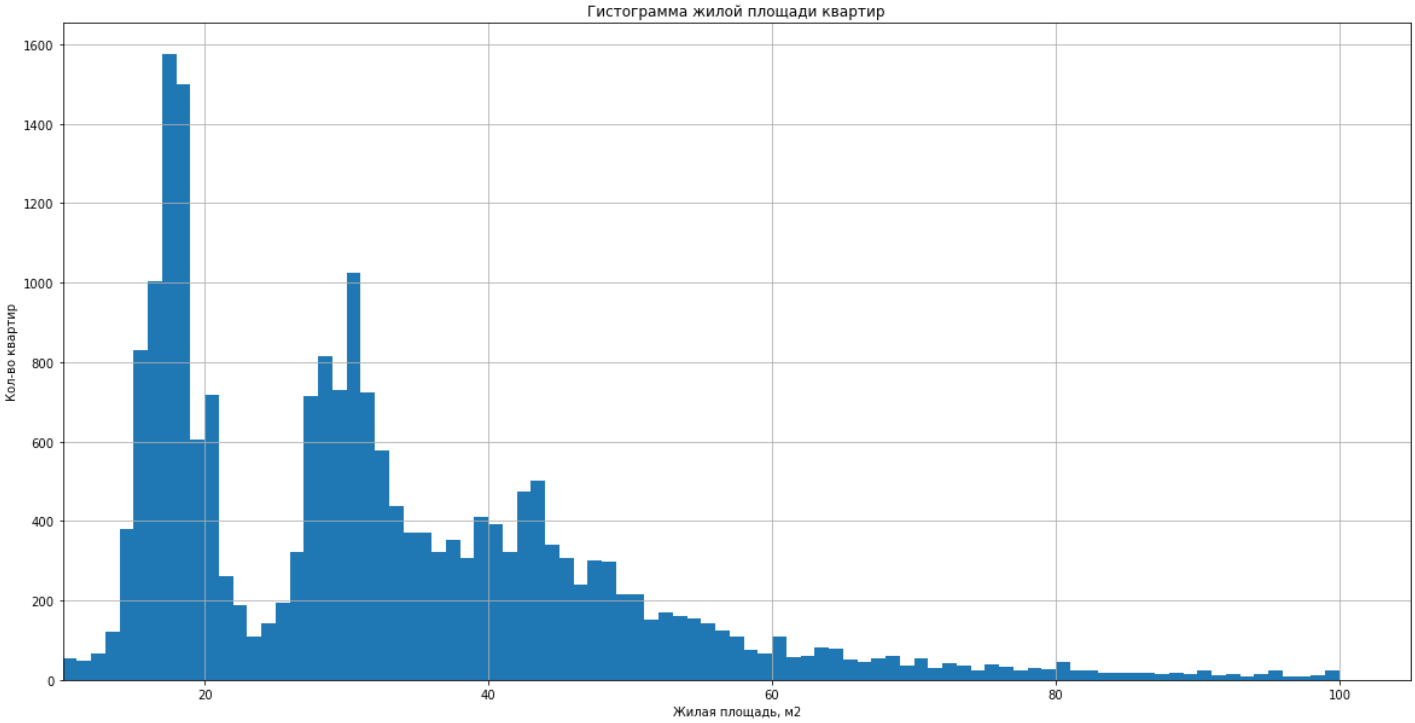
Как мы видим, основная масса обьявления приходится **на квартиры общей площадью от 30 до 70 кв.м.**

Самые популярные площади квартиры с площадью - 30, 42, 44, 61, 63 и 80 кв. м

Параметр - жилая площадь - `living_area`

In [64]:
```
ax = (
    data['living_area']
    .hist(bins=100, range=(0, 100), figsize=(20,10))
    .set(title = 'Гистограмма жилой площади квартир', xlabel = 'Жилая площадь, м2',xlim=10
    )
data['living_area'].describe()
```

Out[64]:
```
count    21733.000000
mean        34.467421
std         22.040627
min          2.000000
25%         18.600000
50%         30.000000
75%         42.300000
max        409.700000
Name: living_area, dtype: float64
```



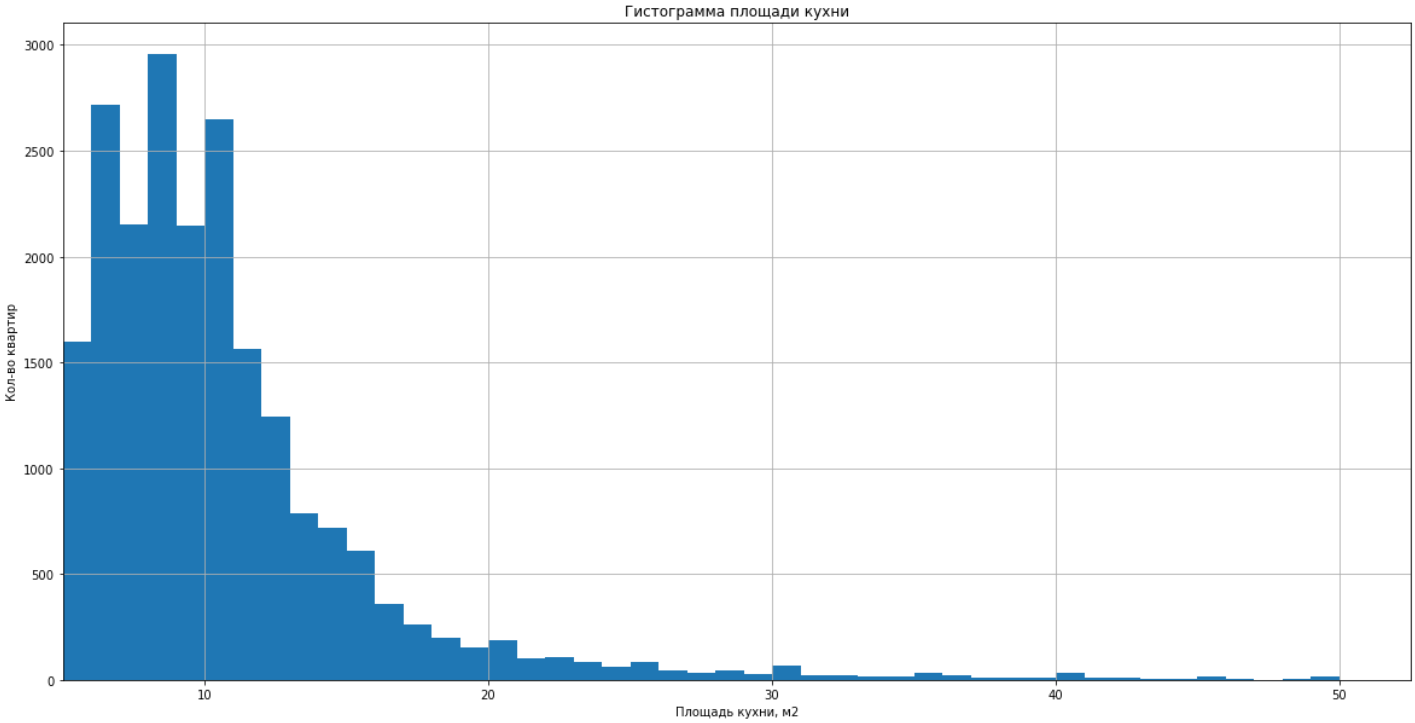
Жилая площадь заведомо меньше Общей площади. Судя по гистограмме - **самые популярные обьявления с жилой площадью от 18 до 42 кв.м.**

Самая популярная жилая площадь, встречающаяся в обьявлениях - 18-19 кв.м., 30 кв.м., 44 кв.м, 61кв.м

Параметр - площадь кухни - `kitchen_area`

In [65]:
```
ax = (
    data['kitchen_area']
    .hist(bins=50, range=(0, 50), figsize=(20,10))
    .set(title = 'Гистограмма площади кухни', xlabel = 'Площадь кухни, м2', xlim=5,ylabel
    )
data['kitchen_area'].describe()
```

Out[65]:
```
count    21374.000000
mean        10.564936
std          5.905188
min          1.300000
25%          7.000000
50%          9.100000
75%         12.000000
max        112.000000
Name: kitchen_area, dtype: float64
```



Самая **популярная площадь кухни от 7,5 до 12 кв.м.**

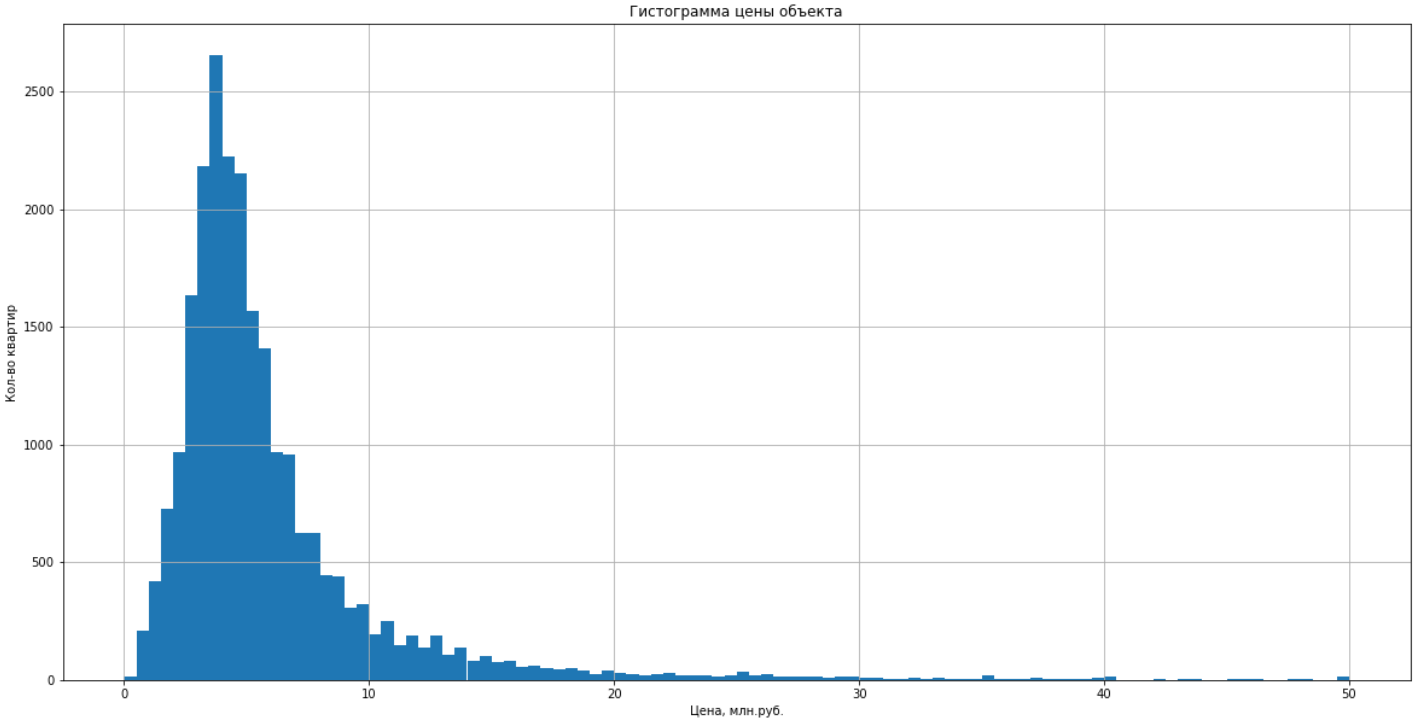
Самые часто встречающиеся площади кухни в обьявлениях - 7,9,11 кв.м.

Параметр - цена объекта - `last_price`

Для корректного отображения гистограммы - создадим столбец `last_price_million` и укажем цену в млн. руб.

In [66]:
```
data['last_price_million'] = data['last_price']/10**6
ax = (
    data['last_price_million'].hist(bins=100,range=(0,50),figsize=(20,10))
    .set(title = 'Гистограмма цены объекта', xlabel = 'Цена, млн.руб.',ylabel = 'Кол-во кв
)
data['last_price_million'].describe()
```

Out[66]:
```
count    23603.000000
mean         6.540975
std         10.903769
min          0.012190
25%          3.400000
50%          4.650000
75%          6.799000
max        763.000000
Name: last_price_million, dtype: float64
```



Посчитаем количество квартир стоимостью более 200 млн.рублей

In [67]:
```
print(data[data['last_price_million']>200]['last_price_million'].count())
print(100*data[data['last_price_million']>200]['last_price_million'].count()/data['last_pr
```

```
9
0.03813074609159853
```

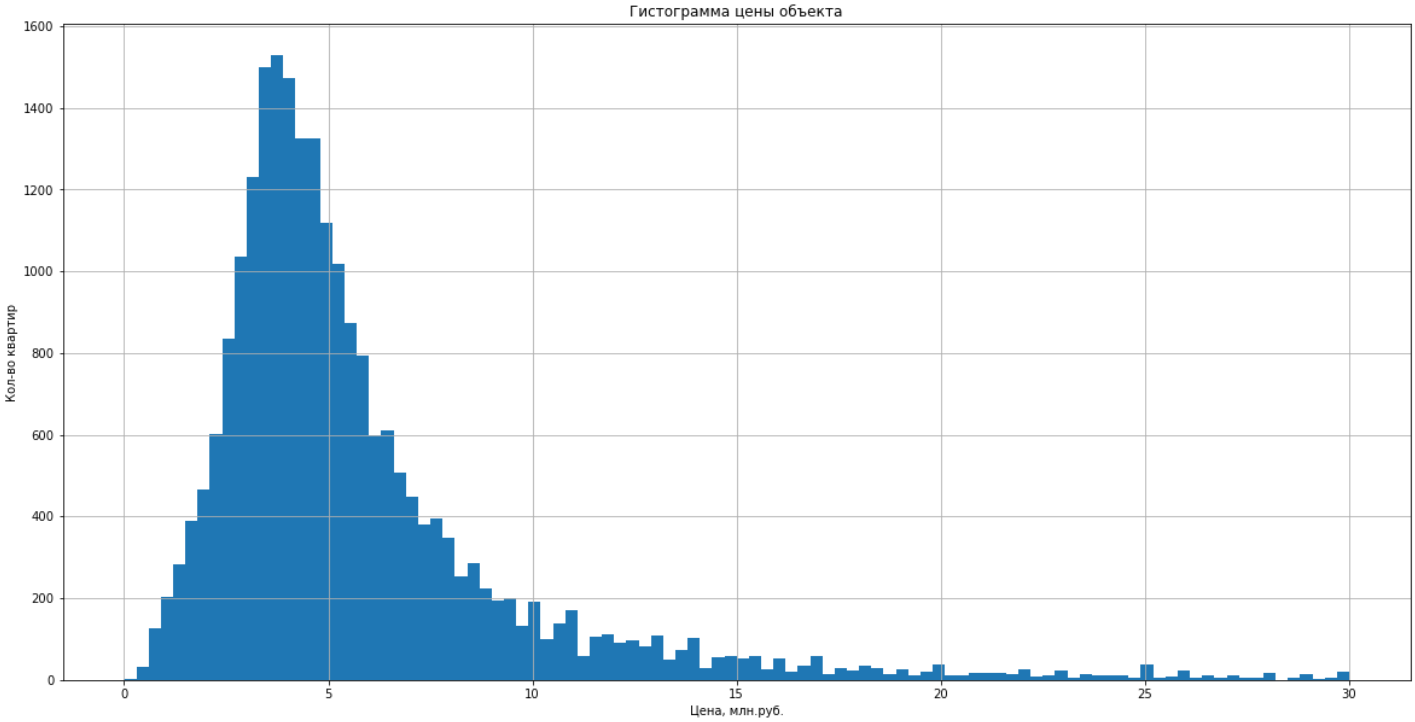
Мы получили 9 квартир, что составляет порядка 0.04 процентов от всей выборки.

Будем считать их за выбросы и удалим их.

In [68]:
```
data = data[data['last_price_million']<=200]
```

In [69]:
```
data['last_price_million'] = data['last_price']/10**6
ax = (
    data['last_price_million'].hist(bins=100,range=(0,30),figsize=(20,10))
    .set(title = 'Гистограмма цены объекта', xlabel = 'Цена, млн.руб.',ylabel = 'Кол-во кв
)
data['last_price_million'].describe()
```

Out[69]:
```
count    23594.000000
mean         6.407057
std          7.906050
min          0.012190
25%          3.400000
50%          4.646000
75%          6.790000
max        190.870000
Name: last_price_million, dtype: float64
```



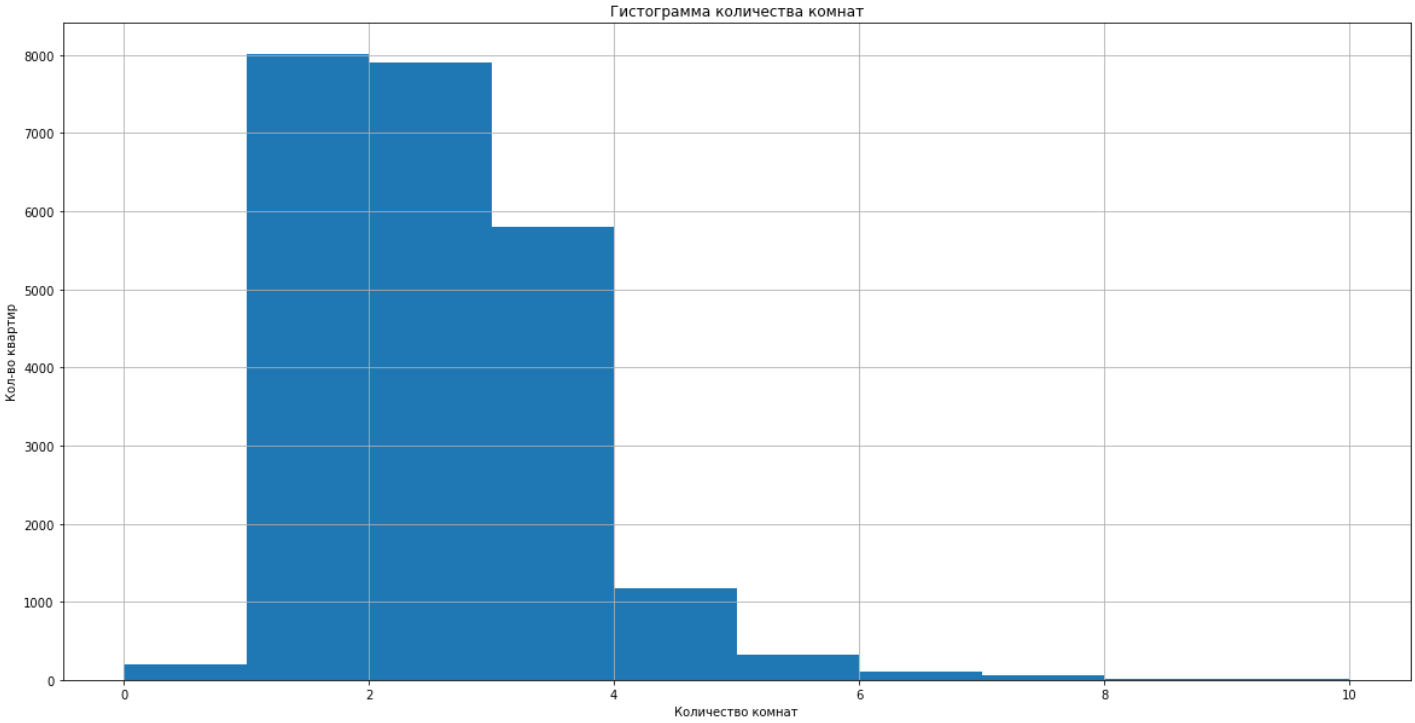
Как мы видим **медианное значение продаж - 4,65 млн.руб.** Основной разброс цены от 3 до 6,8 млн руб.

Параметр - количество комнат - `rooms`

In [70]:
```
ax = (
    data['rooms'].hist(bins=10,range=(0,10),figsize=(20,10))
    .set(title = 'Гистограмма количества комнат', xlabel = 'Количество комнат',ylabel = 'К

)
data['rooms'].describe()
```

Out[70]:
```
count    23594.000000
mean         2.069848
std          1.074689
min          0.000000
25%          1.000000
50%          2.000000
75%          3.000000
max         19.000000
Name: rooms, dtype: float64
```



Как мы видим, есть квартиры с количеством комнат - 0 . Это явно некорректно. Сделаем из них 1-комнатную квартиру.

Посмотрим их количество -

In [71]:
```
data.query('rooms==0')['rooms'].count()
```

Out[71]:
```
194
```

Произведём замену -

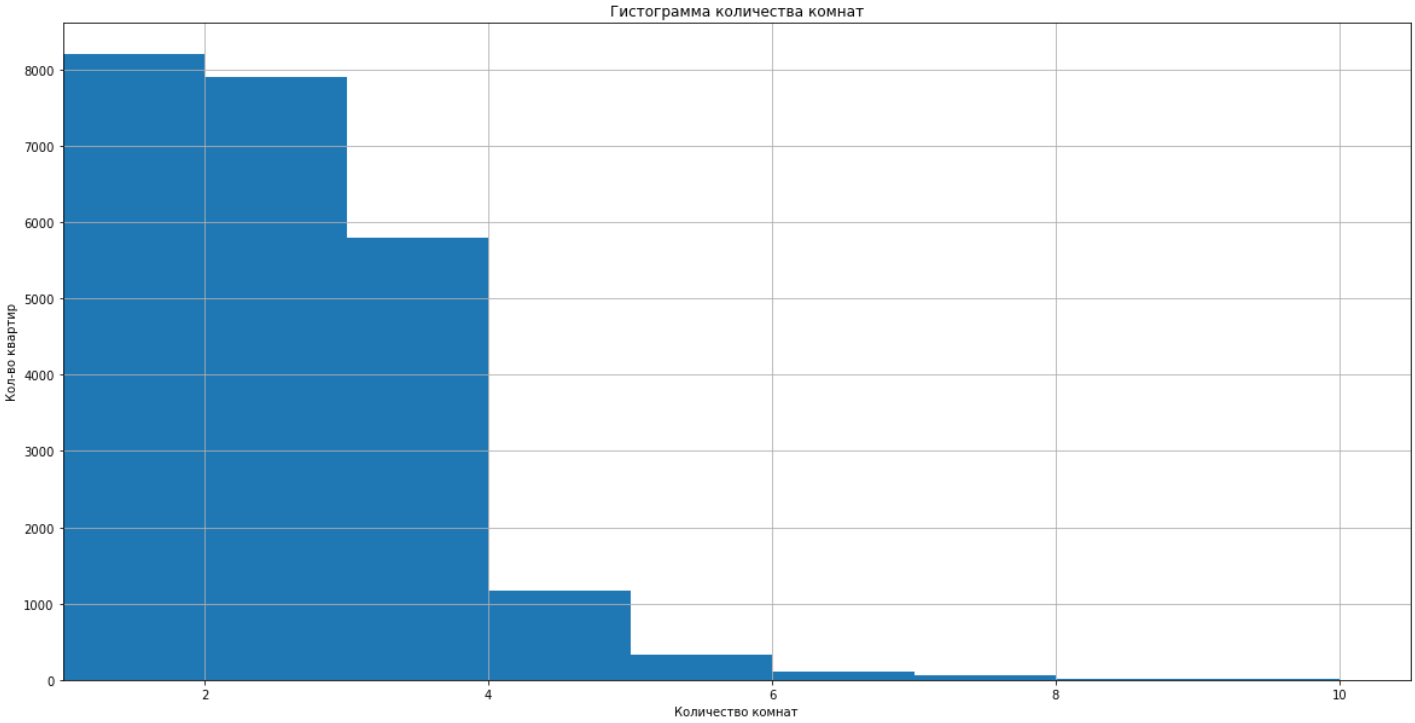
In [72]:
```
data['rooms'] = data['rooms'].replace(0,1)
```

Посмотрим на новый график -

In [73]:
```
ax = (
    data['rooms'].hist(bins=10,range=(0,10),figsize=(20,10))
    .set(title = 'Гистограмма количества комнат',xlim=1, xlabel = 'Количество комнат',ylab

)
data['rooms'].describe()
```

Out[73]:
```
count    23594.000000
mean         2.078071
std          1.062578
min          1.000000
25%          1.000000
50%          2.000000
75%          3.000000
max         19.000000
Name: rooms, dtype: float64
```



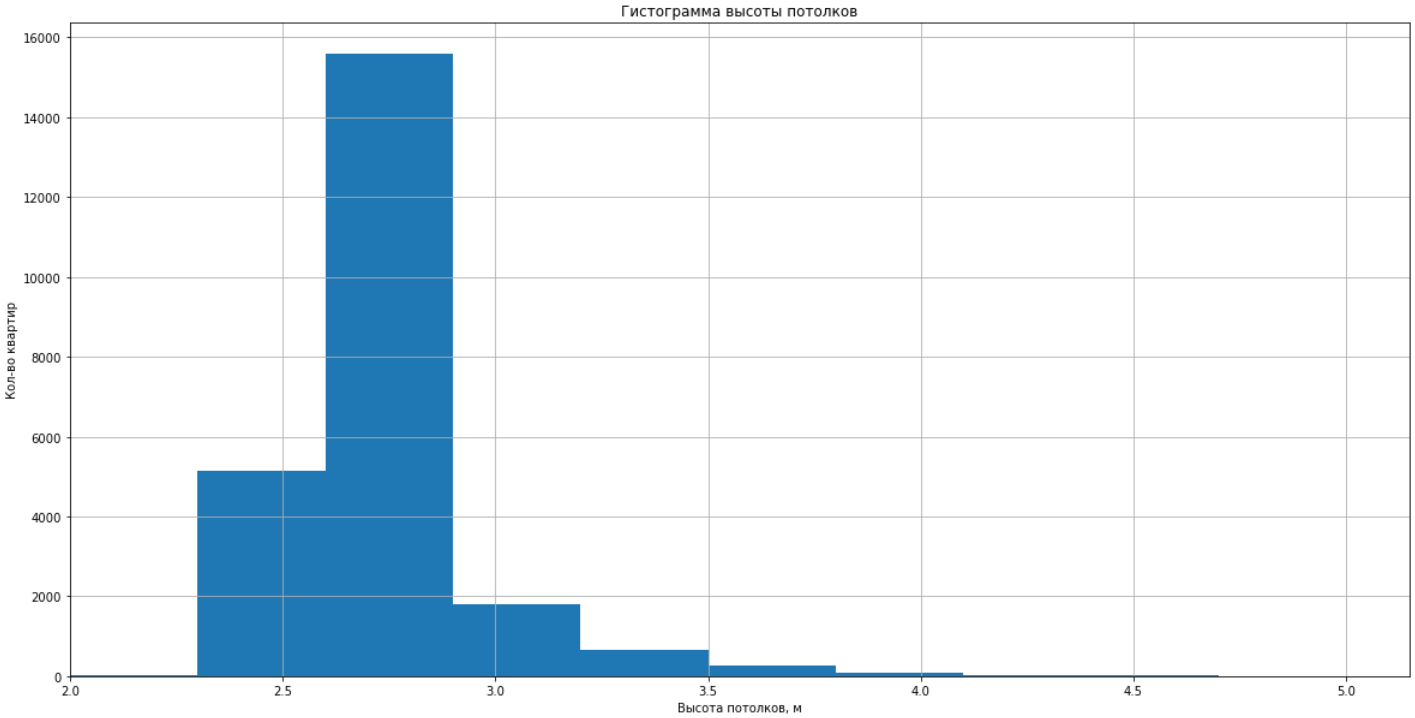
Судя по гистограмме, самые популярные варианты - это **1-3 комнатные квартиры**

- На первом месте по числу обьявлений - однокомнатные
- Второе место - двушки
- Третье место - 3 комнатные квартиры

Параметр - высота потолков - `ceiling_height`

In [74]:
```
ax = (
    data['ceiling_height'].hist(bins=10,range=(2,5),figsize=(20,10))
    .set(title = 'Гистограмма высоты потолков',xlim=2, xlabel = 'Высота потолков, м',ylabe
)
data['ceiling_height'].describe()
```

Out[74]:
```
count    23594.000000
mean         2.696856
std          0.221347
min          2.000000
25%          2.600000
50%          2.650000
75%          2.700000
max          6.000000
Name: ceiling_height, dtype: float64
```

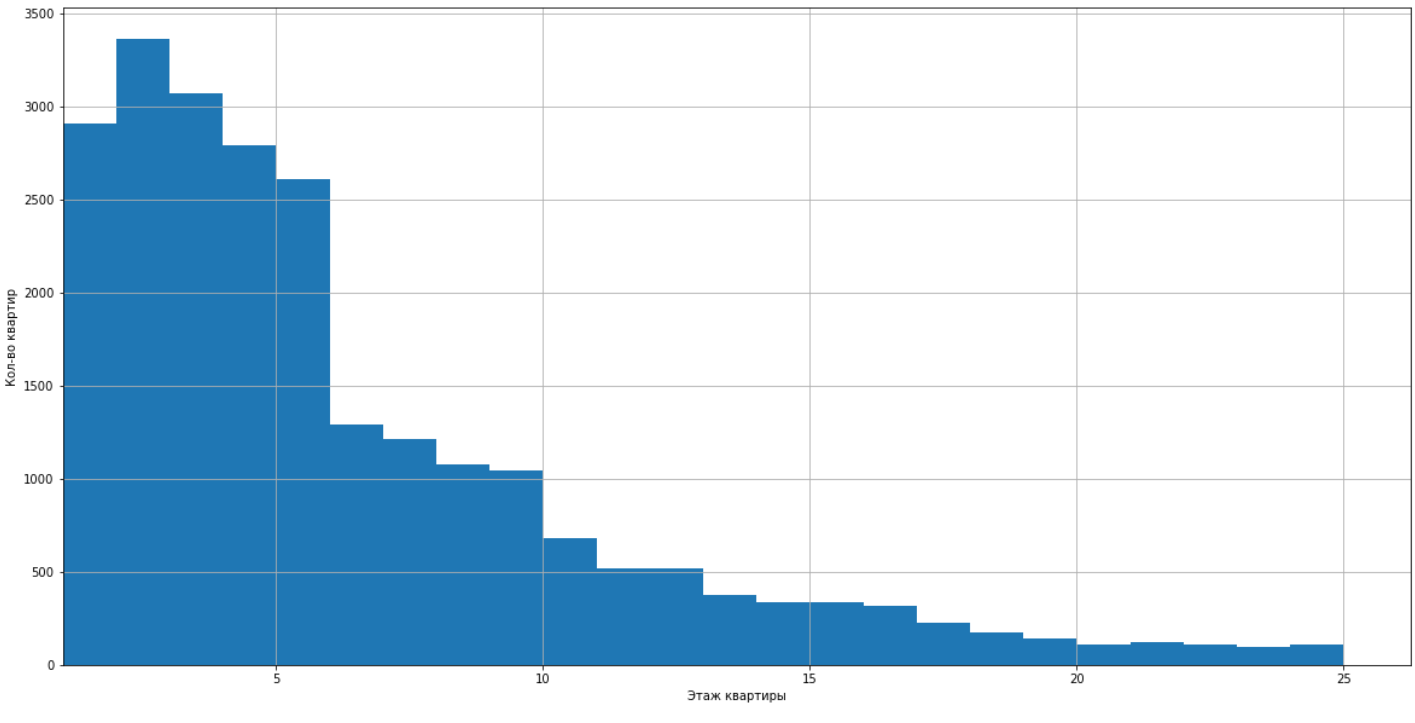


Сдуя по гистограмме - **высота потолков в пределе от 2.5 до 3 метров**

Параметр - этаж квартиры - `floor`

In [75]:
```
ax = (
    data['floor'].hist(bins=25,figsize=(20,10),range=(0,25))
    .set(label='Гистограмма этажа квартиры',xlabel ='Этаж квартиры', ylabel='Кол-во кварти
)
data['floor'].describe()
```

Out[75]:
```
count    23594.000000
mean         5.876324
std          4.872759
min          1.000000
25%          2.000000
50%          4.000000
75%          8.000000
max         33.000000
Name: floor, dtype: float64
```



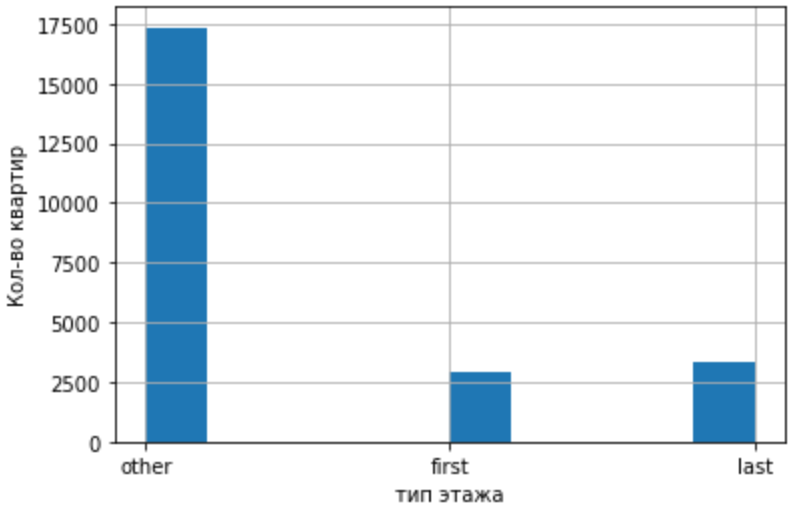
Сулдя по гистограмме - основной процент обьявлений **приходится на квартиры с 1 по 10 этаж.**

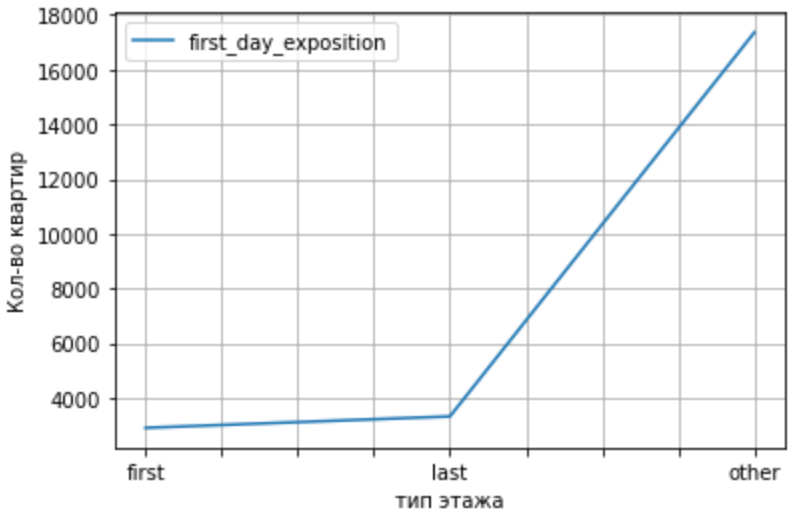
Больше всего обьявлений про квартиры на 1-5 этаже.

Параметр - тип этажа квартиры - `floor_type`

In [76]:
```
data['floor_type'].hist(bins=10).set(label='Гистограмма типа этажа', xlabel = 'тип этажа',
(
    data.pivot_table(index='floor_type',values='first_day_exposition',aggfunc='count')
    .plot(grid=True)
    .set(label='Гистограмма типа этажа', xlabel = 'тип этажа', ylabel = 'Кол-во квартир')
)
```

Out[76]:
```
[None, Text(0.5, 0, 'тип этажа'), Text(0, 0.5, 'Кол-во квартир')]
```





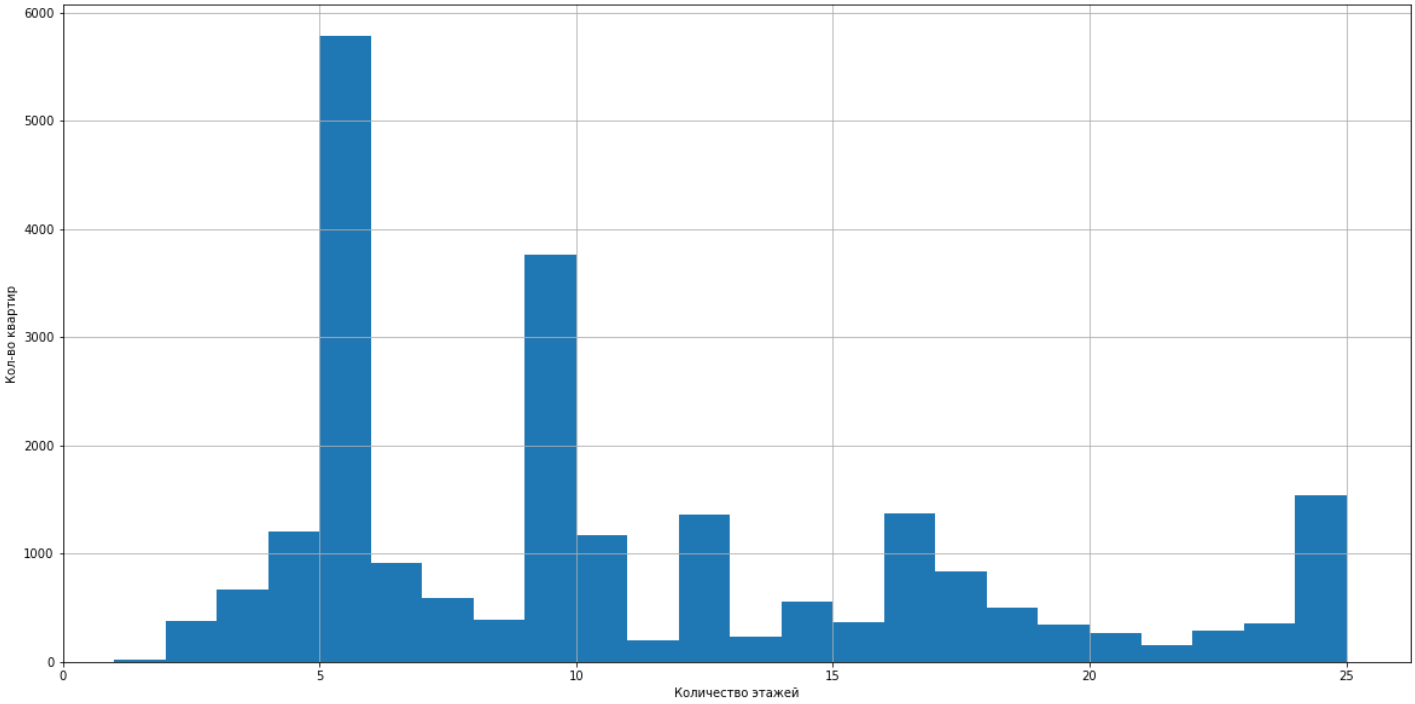
Как мы видим, **большая часть обьявлений приходится на тип этажа - other (не первый, не последний).**

Параметр - Общее количество этажей в доме - `floors_total`

In [77]:
```
ax = (
    data['floors_total'].hist(bins=25,figsize=(20,10),range=(0,25))
    .set(label='Гистограмма общего количества этажей в доме',xlabel ='Количество этажей',
)
data['floors_total'].describe()
```

Out[77]:
```
count    23594.000000
mean        10.673392
std          6.595780
```

```
min          1.000000
25%          5.000000
50%          9.000000
75%         16.000000
max         60.000000
Name: floors_total, dtype: float64
```



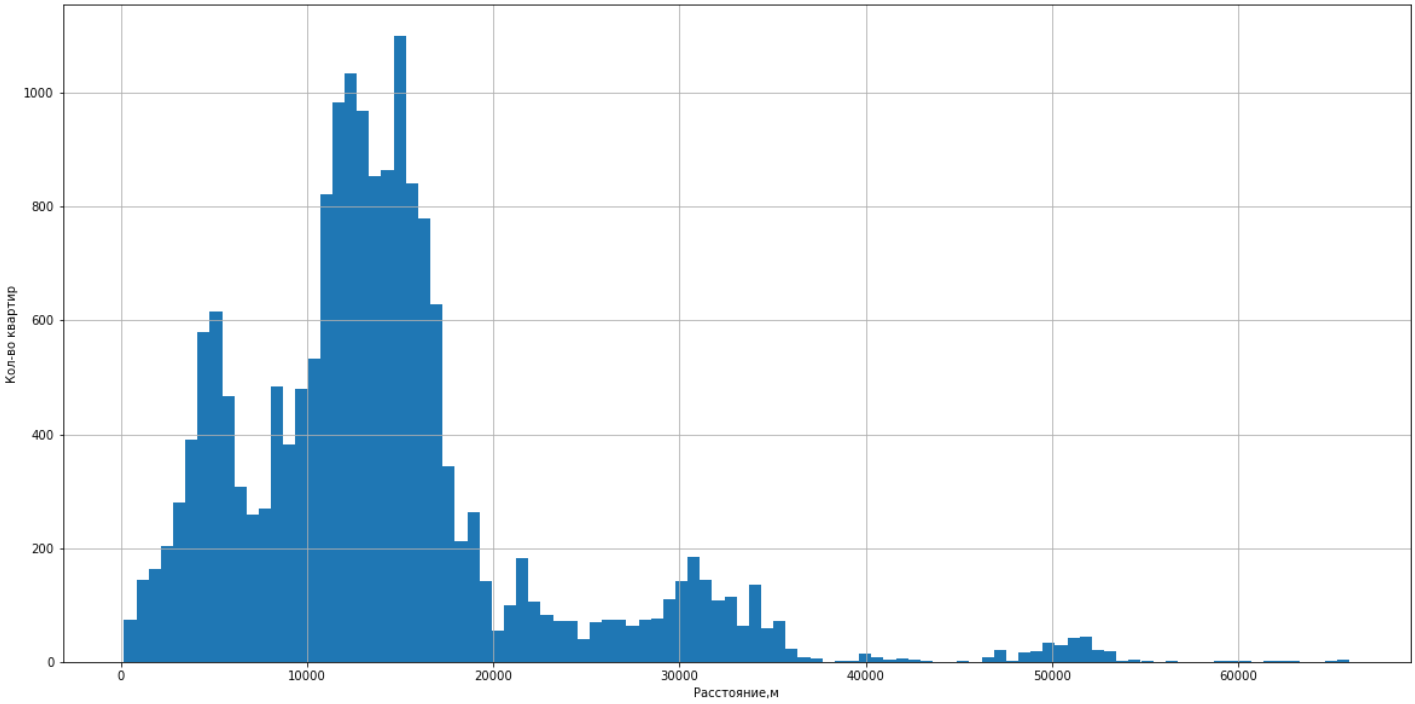
Самые популярные обьявления в домах от 5 до 16 ти этажей

Самые популярные - 5(пяти) и 9 (девяти) этажки

Параметр - Расстояние до центра города в метрах - `cityCenters_nearest`

In [78]:
```
ax = (
    data['cityCenters_nearest'].hist(bins=100,figsize=(20,10))
    .set(label='Гистограмма расстояния до центра города',xlabel='Расстояние,м',ylabel='Кол
)
data['cityCenters_nearest'].describe()
```

Out[78]:
```
count    18087.000000
mean     14189.738099
std       8614.270530
min        181.000000
25%       9234.000000
50%      13094.000000
75%      16293.000000
max      65968.000000
Name: cityCenters_nearest, dtype: float64
```



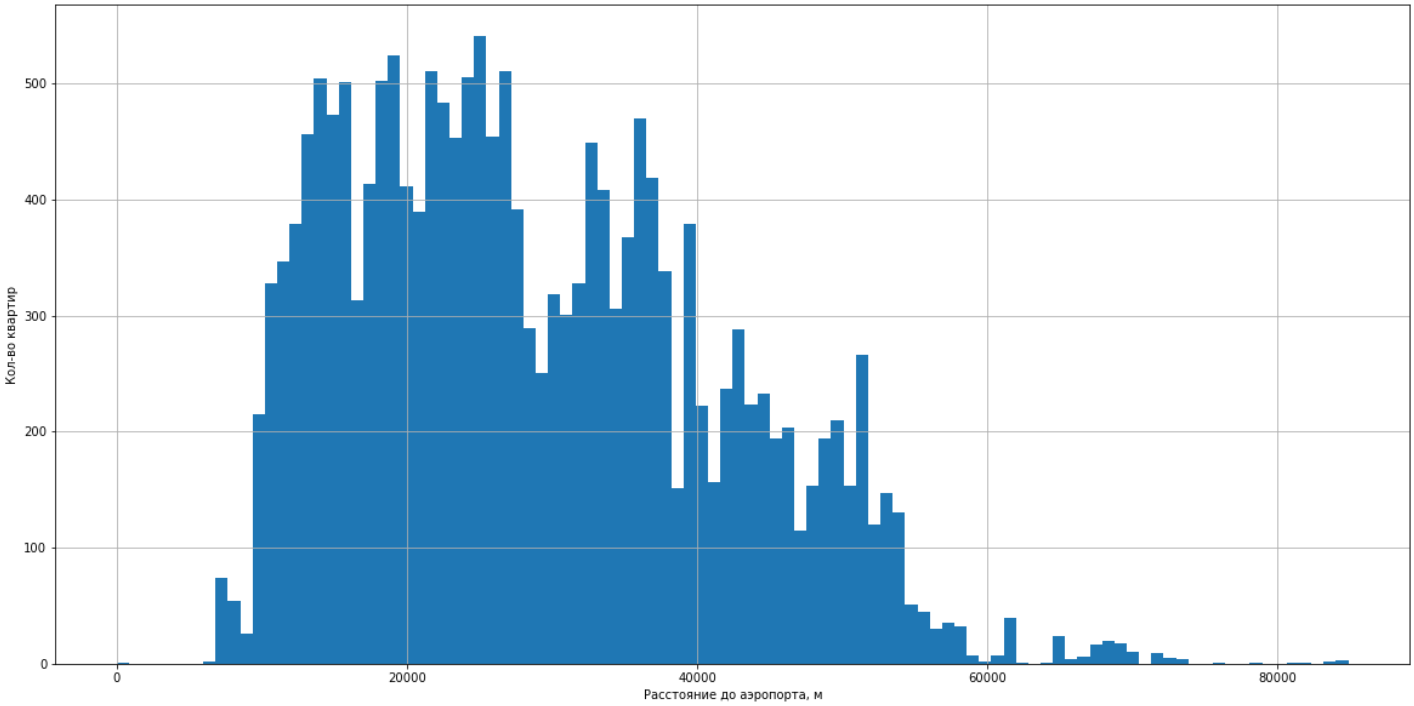
Медианное расстояние до центра города - 13 км, межквартильный размах от 9 до 16 км.

- также много квартир с расстоянием 5 км от центра города.

Параметр - Расстояние до ближайшего аэропорта - `airports_nearest`

In [79]:
```
ax = (
    data['airports_nearest'].hist(bins=100,figsize=(20,10))
    .set(label='Расстояние до ближайшего аэропорта',xlabel='Расстояние до аэропорта, м', y
)
data['airports_nearest'].describe()
```

Out[79]:
```
count    18164.000000
mean     28778.322038
std      12601.873834
min          0.000000
25%      18605.750000
50%      26760.500000
75%      37197.500000
max      84869.000000
Name: airports_nearest, dtype: float64
```

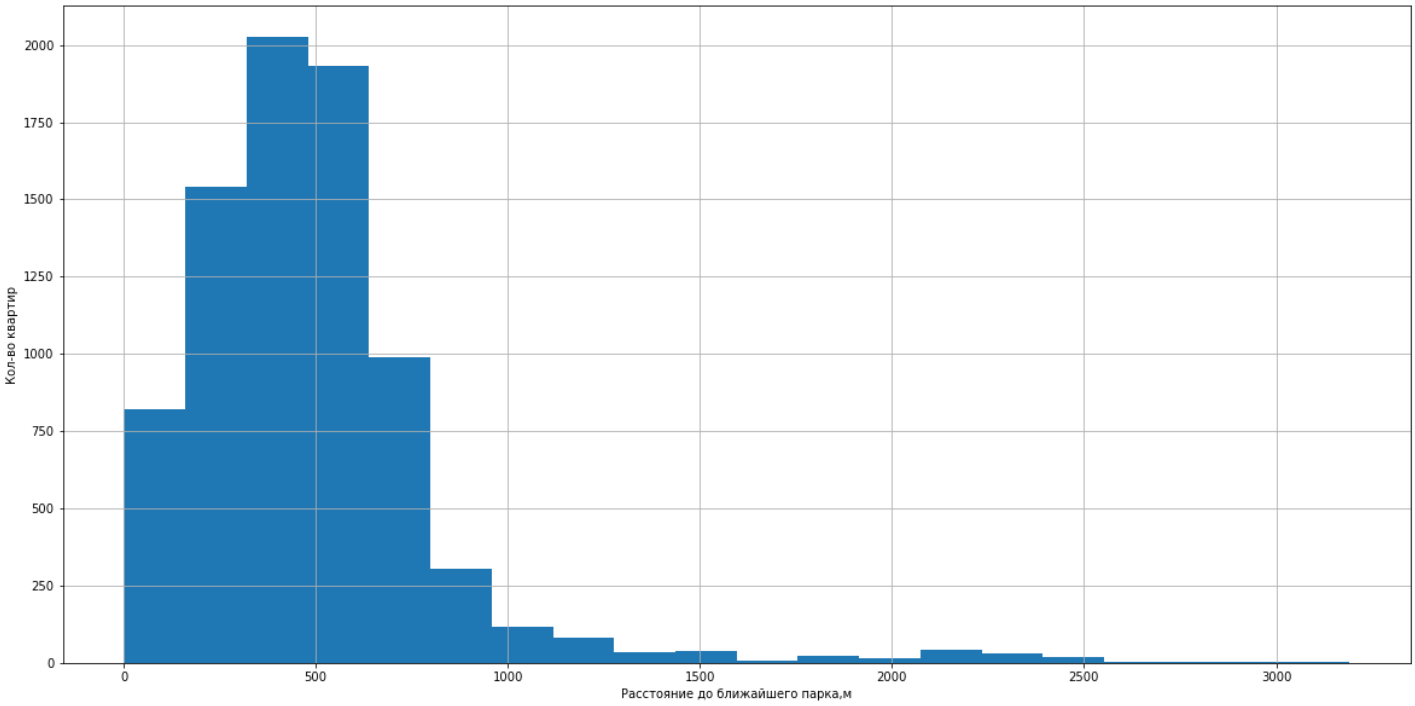


В наших объявления основной разброс расстояний до аэропорта составляет **от 18 до 37 км**

Параметр - Расстояние до ближайшего парка - `parks_nearest`

In [80]:
```
ax =(
    data['parks_nearest'].hist(bins=20,figsize=(20,10))
    .set(xlabel= 'Расстояние до ближайшего парка,м',ylabel='Кол-во квартир')
)
data['parks_nearest'].describe()
```

Out[80]:
```
count    8036.000000
mean      490.598930
std       341.492883
min         1.000000
25%       288.000000
50%       455.000000
75%       612.000000
max      3190.000000
Name: parks_nearest, dtype: float64
```

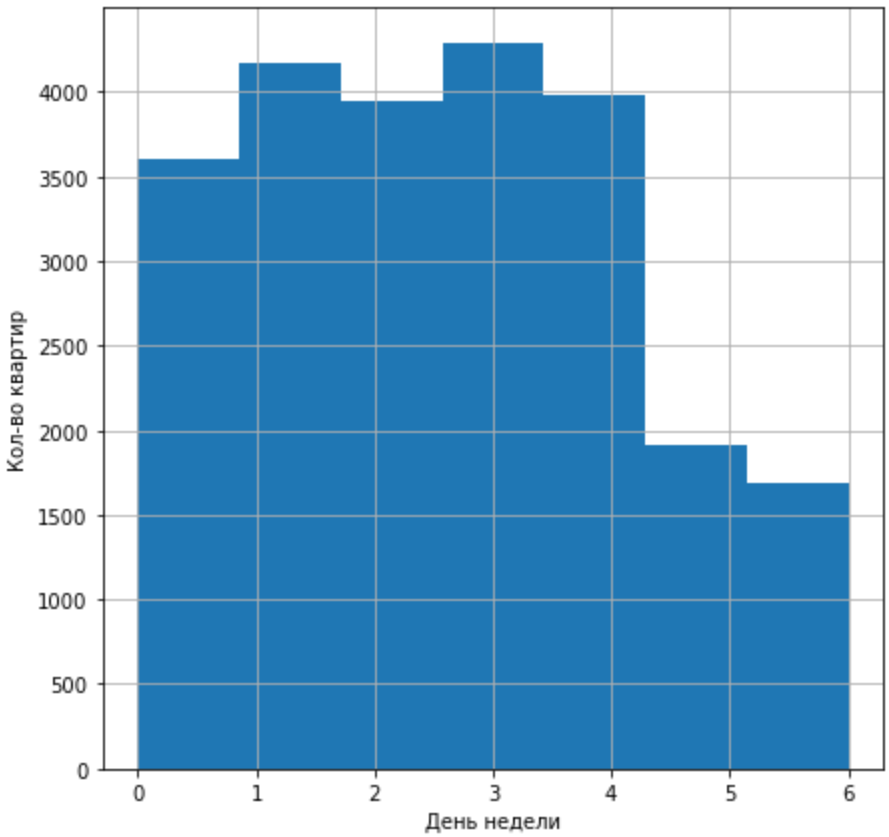


В основной массе доступных объявлений (без учета обьявлений с пропусками) **ближайший парк находится на расстоянии от 290 до 610 метров**

- Либо находится на расстоянии более 3км от квартиры , согласно пункту 2.6 исследования

Параметр - День публикации обьявления - `ad_day_of_week`

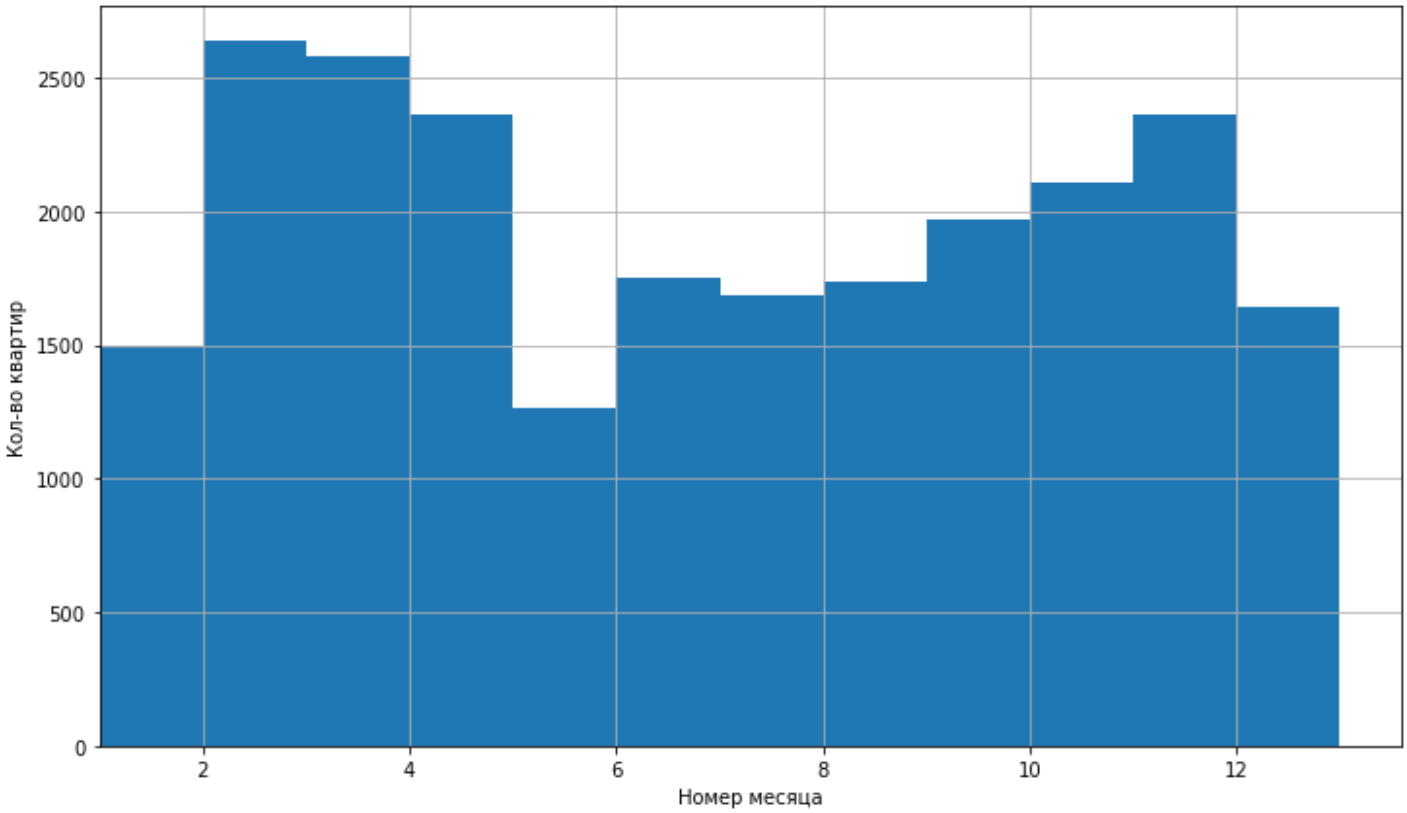
In [81]:
```
ax = (
    data['ad_day_of_week'].hist(bins=7, figsize=(7,7))
    .set(xlabel= 'День недели',label=' Гистограмма дня публикации обьявления', ylabel='Кол
)
```



Большая часть обьявлений **выкладывается в будние дни, пики - вторник, четверг**

Параметр - месяц публикации обьявления - `ad_month`

In [82]:
```
ax = (
    data['ad_month'].hist(bins=12, figsize=(12,7),range=(1,13))
    .set(xlabel= 'Номер месяца',label=' Гистограмма месяца публикации обьявления',xlim=1,
)
```



Самые популярные месяца для продажи квартиры - **февраль, март, апрель, сентябрь, октябрь, ноябрь.**

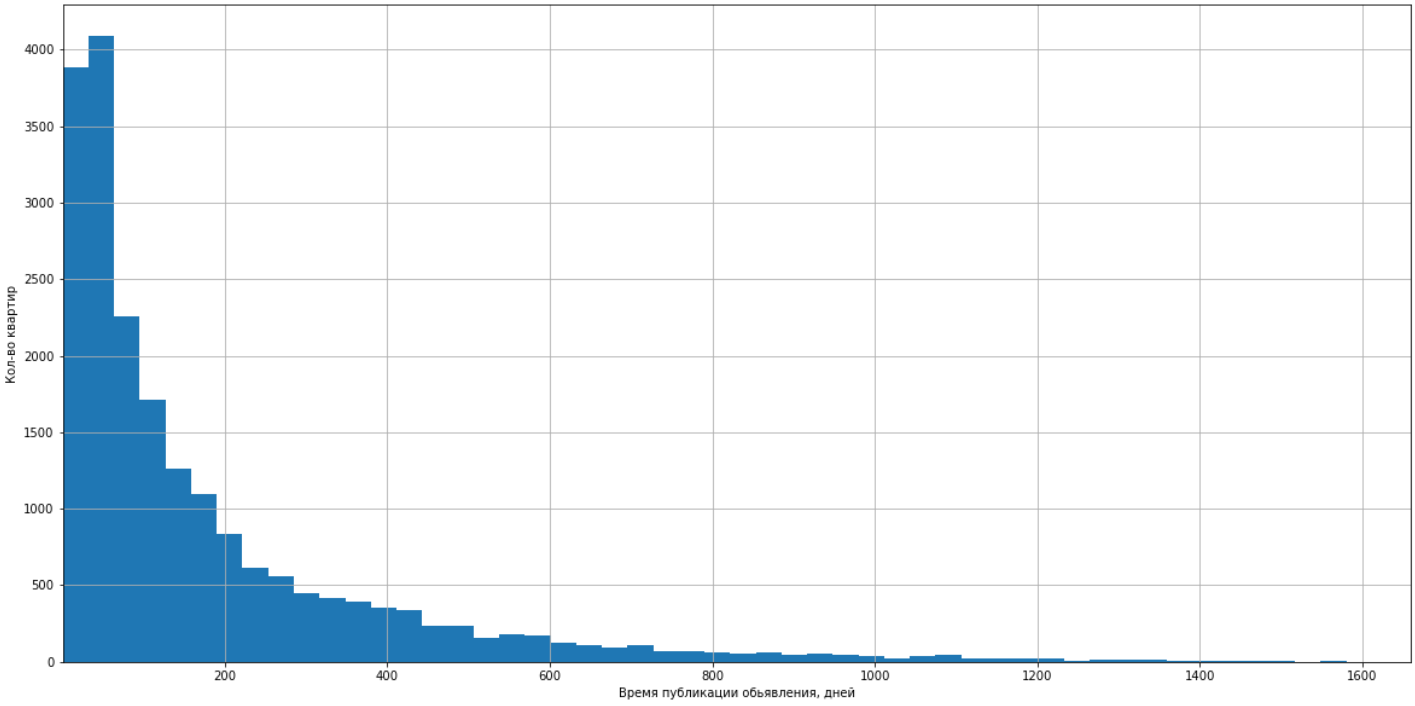
Самый непопулярный - май (скорее всего связан с большим количеством праздничных дней)

Изучим, как быстро продавались квартиры- `days_exposition` .

Этот параметр показывает, сколько дней было размещено каждое объявление.

In [83]:
```
ax = (
    data[data['sold']==True]['days_exposition']
    .hist(bins=50, figsize=(20,10))
    .set(xlabel= 'Время публикации обьявления, дней',label=' Гистограмма продолжительности
         xlim=1, ylabel='Кол-во квартир'))
data[data['sold']==True]['days_exposition'].describe()
```

Out[83]:
```
count    20423.000000
mean       180.884738
std        219.738549
min          1.000000
25%         45.000000
50%         95.000000
75%        232.000000
max       1580.000000
Name: days_exposition, dtype: float64
```



- Медианное значение продажи составляет - 95 дней
- Среднее значение - 180 дней

Судя по информации из нашей таблицы,в большой части обьявлений **время продажи составило от 45 до 232 дней.**

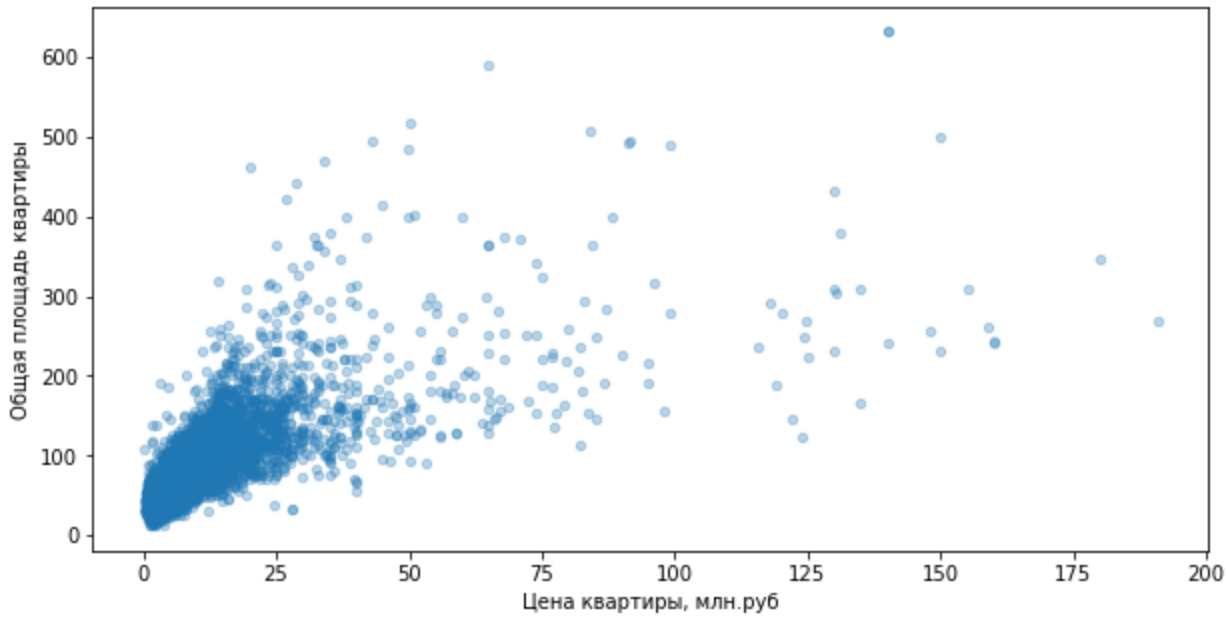
Мы можем считать **быстрыми** продажи квартиры - со временем продажи **до 1(одного) месяца.**

Долгие продажи - более одного года .

Изучим влияние различных факторов на общую (полную) стоимость Объекта

- Посмотрим влияние **общей площади** на цену недвижимости.

In [84]:
```
ax = data.plot(x='last_price_million', y='total_area', kind = 'scatter', alpha=0.3,\
        xlabel = 'Цена квартиры, млн.руб',ylabel='Общая площадь квартиры',figsize=(10,5))
```



На диаграмме рассеивания мы видим положительную корреляцию - при увеличение Общей площади

возрастает и стоимость Объекта - что логично.

Ниже расчитаем коэффициент корреляции Пирсона -

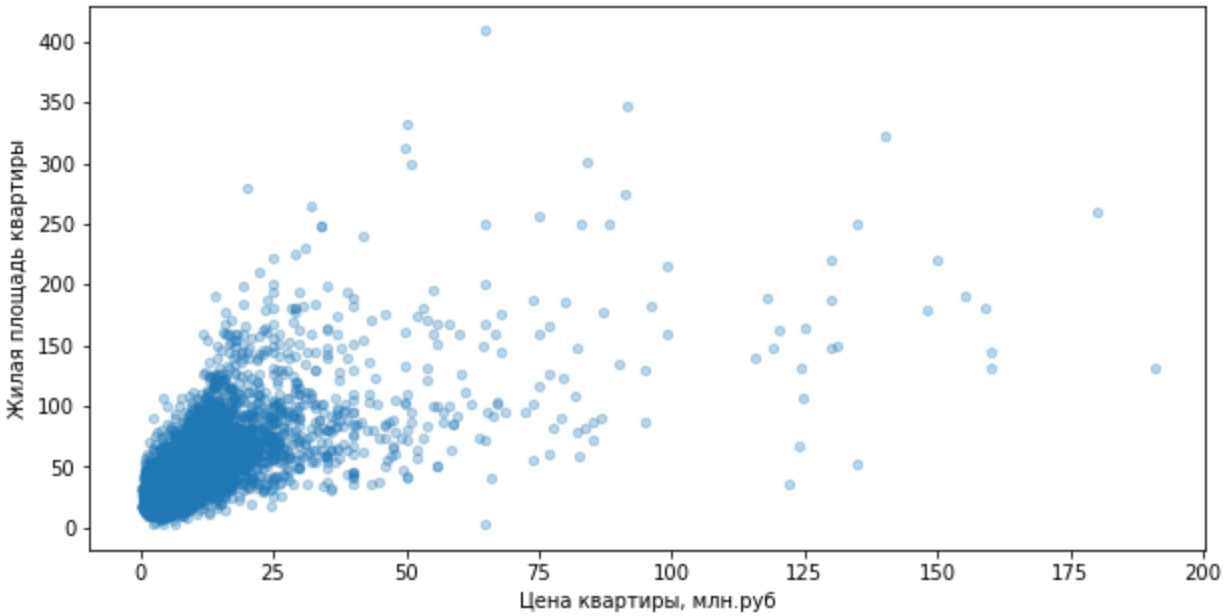
In [85]:
```
print(data['last_price'].corr(data['total_area']))
```

```
0.7396981250526052
```

- Посмотрим влияние **жилой площади** на цену недвижимости.

In [86]:
```
ax = data.plot(x='last_price_million', y='living_area', kind = 'scatter', alpha=0.3,\
               xlabel = 'Цена квартиры, млн.руб',ylabel='Жилая площадь квартиры',figsize=(1
print(data['last_price'].corr(data['living_area']))
```

```
0.6591309163808898
```

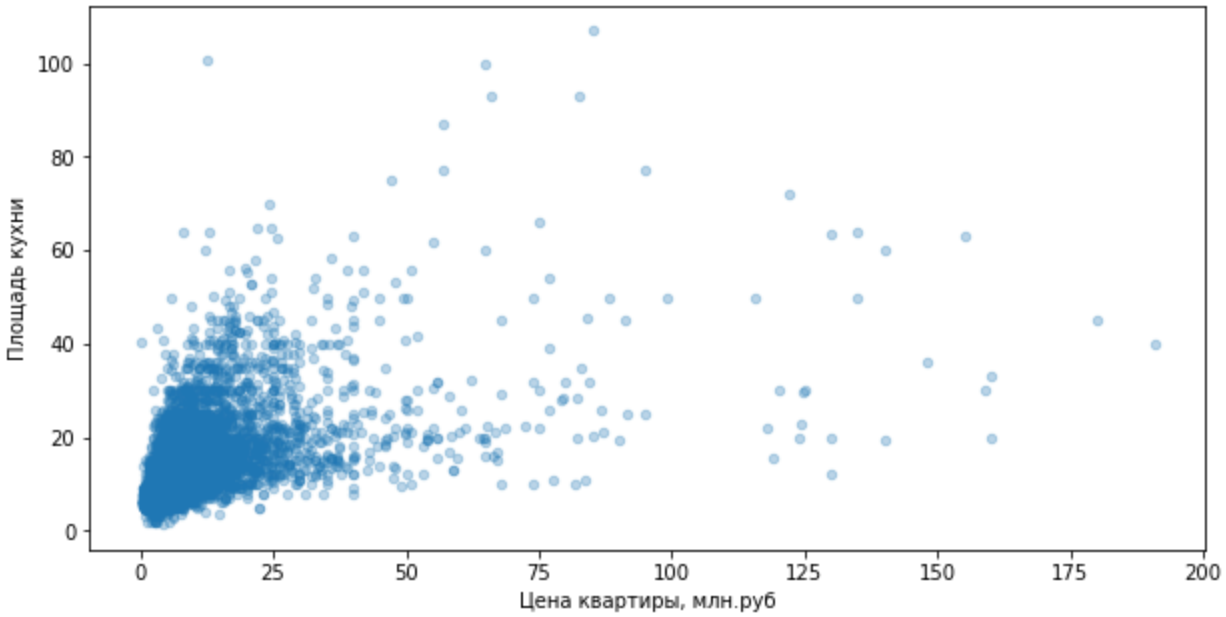


На графике мы видим **прямую зависимость жилой площади на стоимость недвижимости**, аналогично общей площади.

- Изучим влияние **площади кухни** на цену недвижимости.

In [87]:
```
ax = (
    data.plot(x='last_price_million', y='kitchen_area', kind = 'scatter', alpha=0.3,\
              xlabel = 'Цена квартиры, млн.руб',ylabel='Площадь кухни',figsize=(10,5)))
print(data['last_price'].corr(data['kitchen_area']))
```

```
0.5635951012195863
```

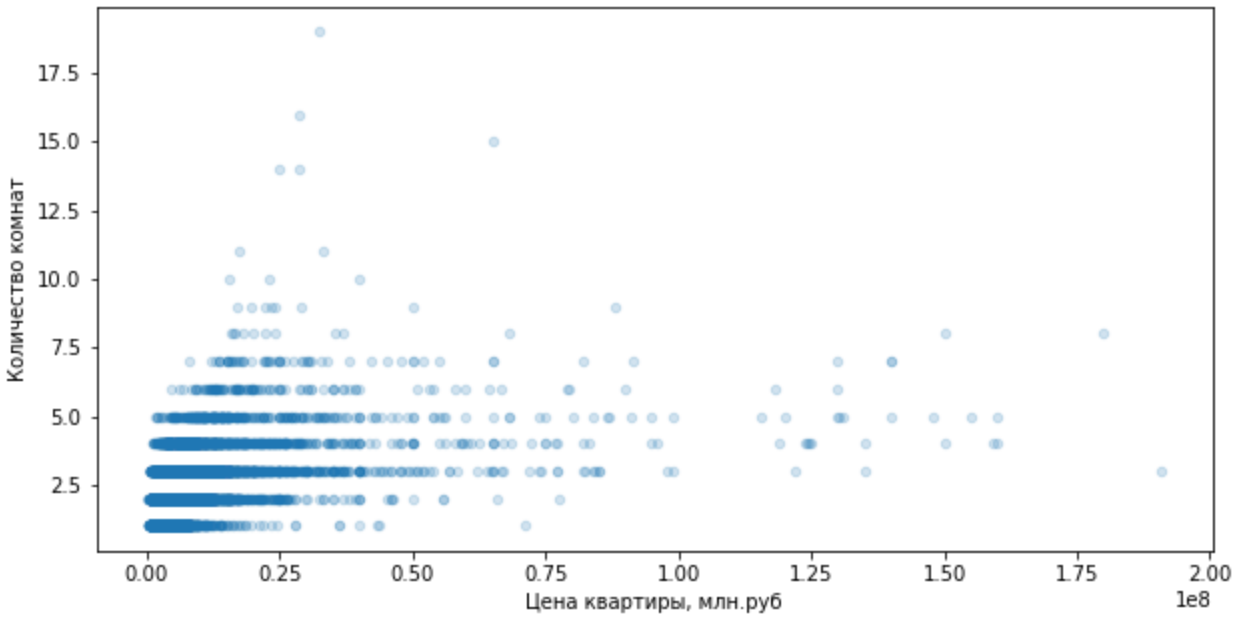


Как мы видим, при **увеличении площади кухни стоимость объекта увеличивается.**

- Изучим влияние параметра **количество комнат** на общую стоимость жилья

In [88]:
```
ax = data.plot(x='last_price', y='rooms', kind = 'scatter', alpha=0.2,\
               xlabel = 'Цена квартиры, млн.руб',ylabel='Количество комнат',figsize=(10,5))
print(data['last_price'].corr(data['rooms']))
```

```
0.43769333855441733
```



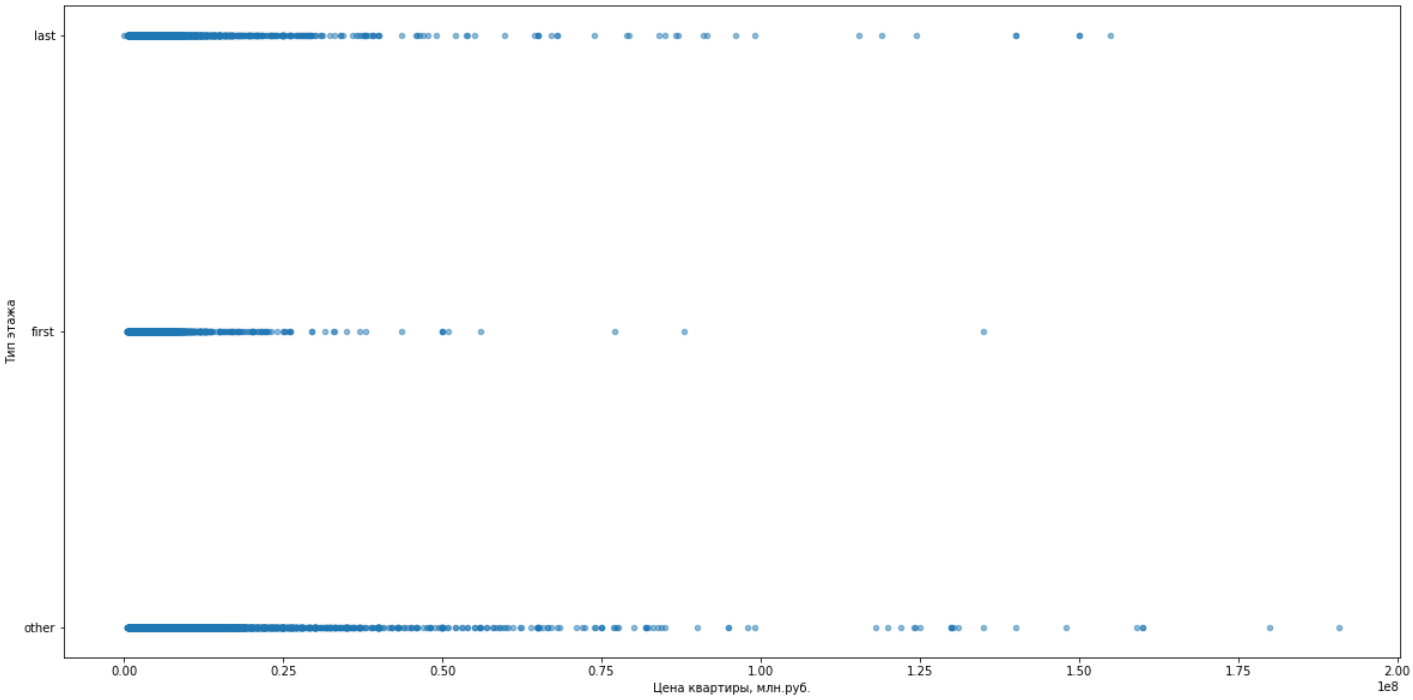
Как мы видим зависимость есть, но она не такая сильная. При увеличении количества комнат от 1 до 5х - цена недвижимости увеличивается.

Квартиры с большим количеством комнат (от 7ми) плохо продаются, и цену приходится снижать.

- Изучим влияние параметра **типа этажа (первый, последний, другой)** `floor_type` на общую стоимость жилья

In [89]:
```
ax = (
    data.plot(x='last_price', y='floor_type', kind = 'scatter', alpha=0.5,
```

```
                xlabel='Цена квартиры, млн.руб.',ylabel='Тип этажа', figsize=(20,10))
)
```



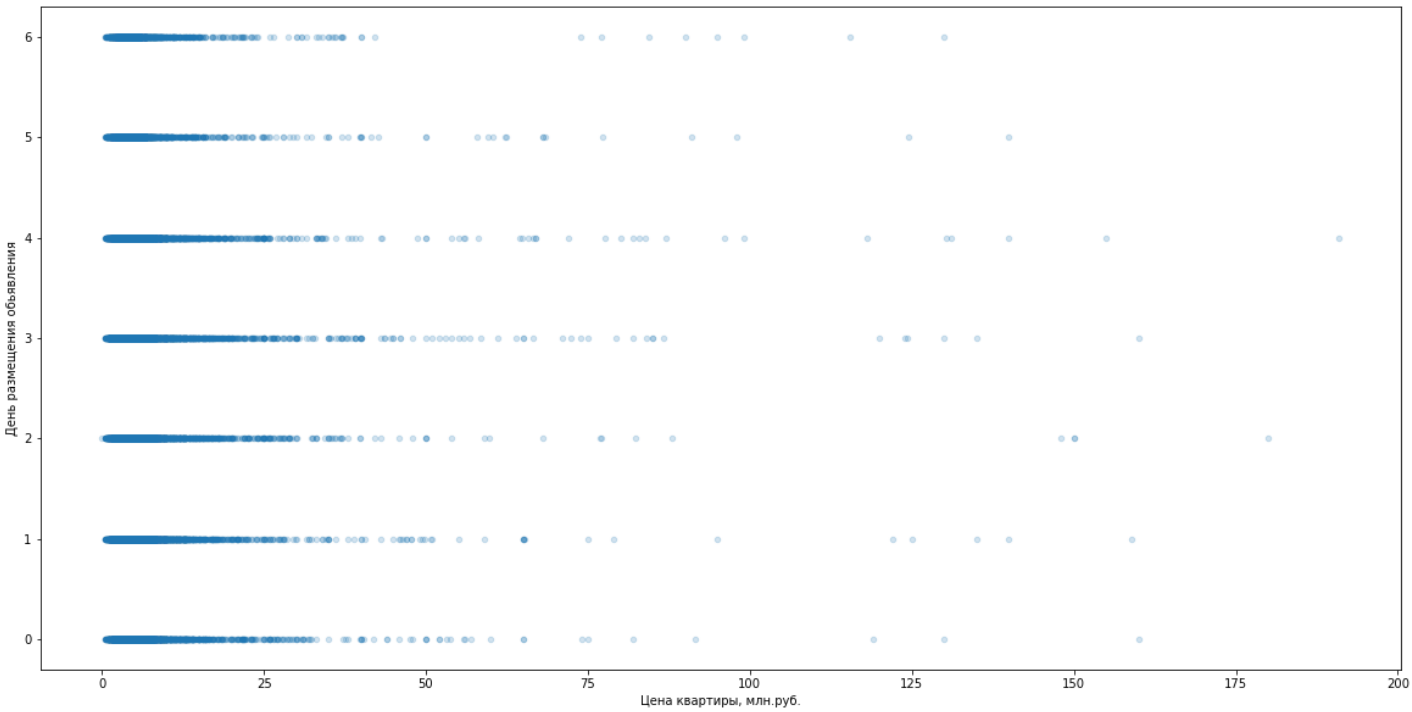
Как мы видим, самые дорогие варианты квартир находятся на этаже "другой", либо "последний" - это, скорее всего , элитное жилье - пентхаусы.

Квартиры на первом этаже стоят дешевле всего.

- Изучим влияние параметра **дня недели публикации обьявления** ad_day_of_week на общую стоимость жилья

In [90]:
```
ax = data.plot(x='last_price_million', y='ad_day_of_week', kind = 'scatter', alpha=0.2,
                xlabel='Цена квартиры, млн.руб.',ylabel='День размещения обьявления', figsi
print(data['last_price_million'].corr(data['ad_day_of_week']))
```

```
-0.001252749252383894
```

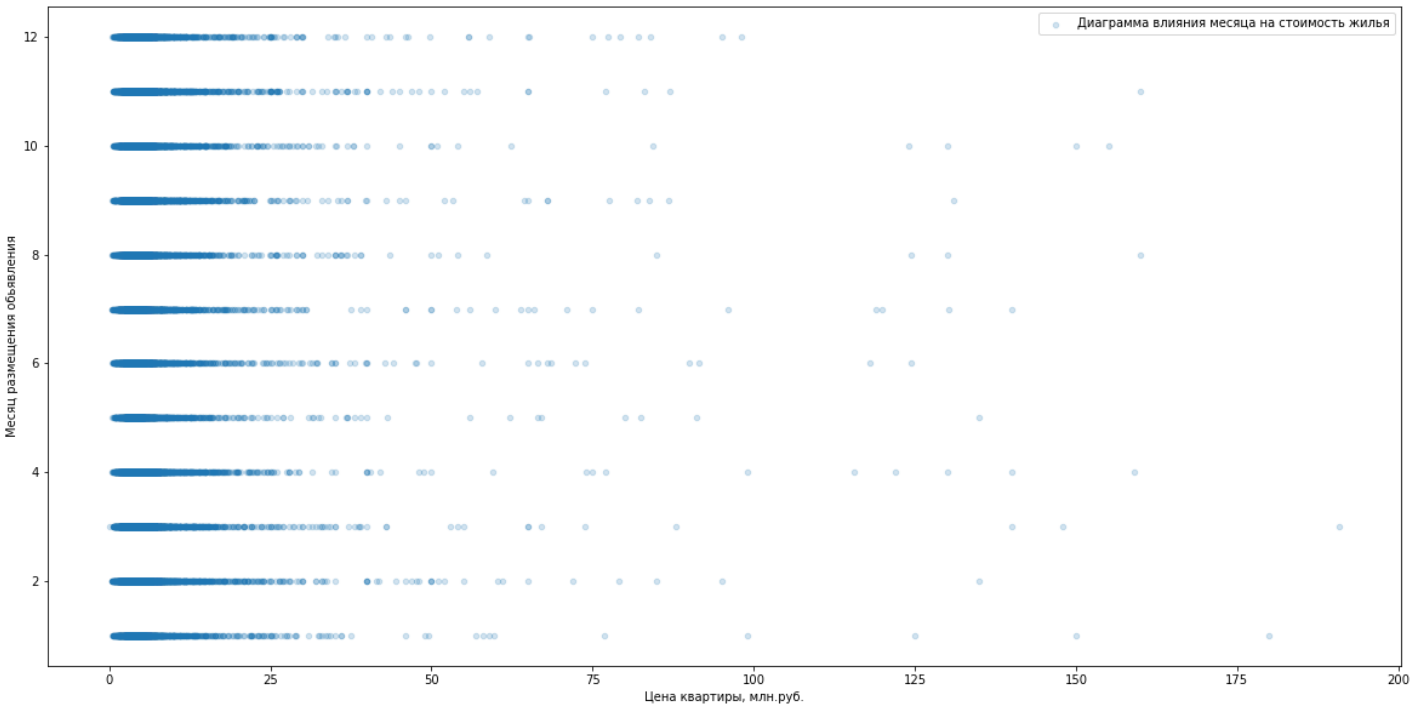


Как мы видим, корреляция нулевая. Это значит, что нет зависимости в какой из дней опубликовано обьявление.

- Изучим влияние параметра **Месяца публикации обьявления** `ad_month` на общую стоимость жилья

In [91]:
```
ax = (
    data.plot(x='last_price_million', y='ad_month', kind = 'scatter', alpha=0.2,figsize=(2
              label='Диаграмма влияния месяца на стоимость жилья',
               xlabel='Цена квартиры, млн.руб.',ylabel='Месяц размещения обьявления'))
print(data['last_price_million'].corr(data['ad_month']))
```

```
0.002275953623940217
```

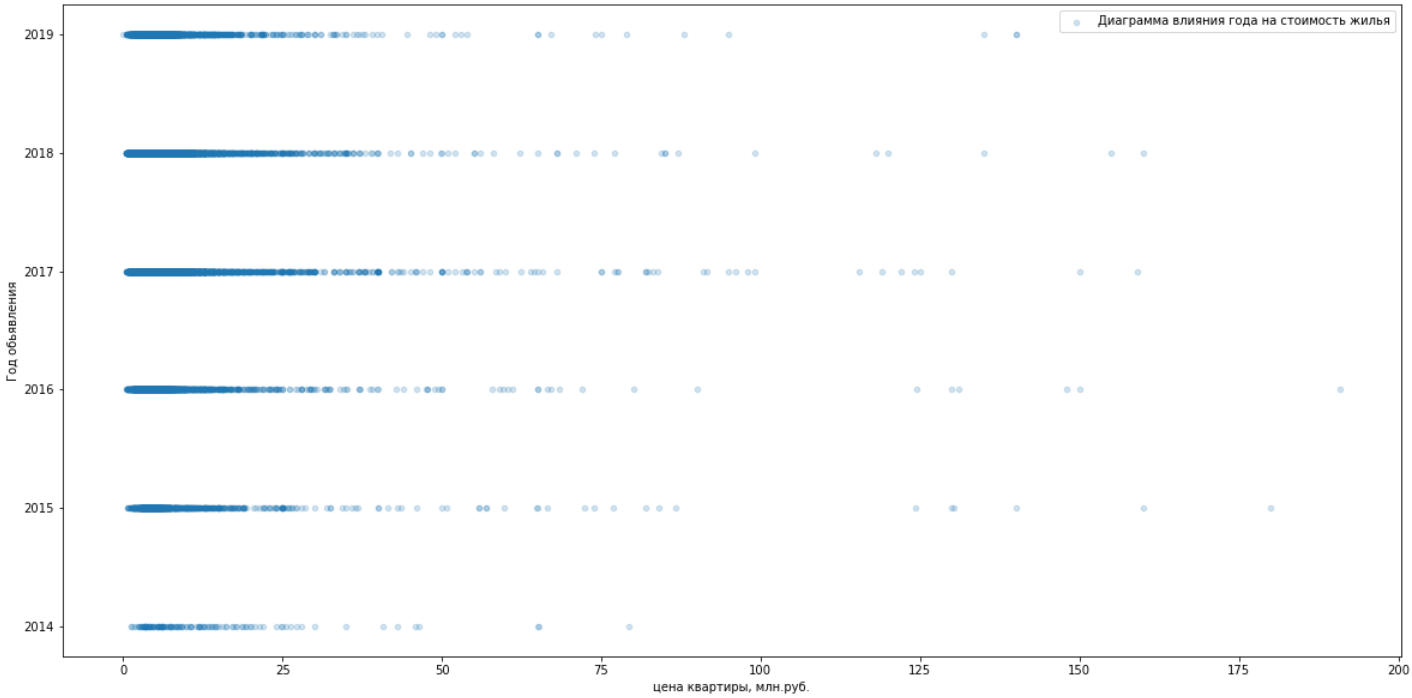


Как мы видим, корреляция нулевая. Это значит, что нет зависимости в какой из месяцев опубликовано обьявление.

- Изучим влияние параметра **года публикации обьявления** `ad_year` на общую стоимость жилья

In [92]:
```
ax = (
    data.plot(x='last_price_million', y='ad_year', kind = 'scatter', alpha=0.2,figsize=(20
              label='Диаграмма влияния года на стоимость жилья',xlabel='цена квартиры, млн.
)
print(data['last_price_million'].corr(data['ad_year']))
```

```
-0.055955487833756035
```



Как мы видим, в 2014 году были самые минимальные цена на недвижимость, скорее всего, связанные с кризисом 2014 года. Затем цены пошли вверх до 2017 года.

В 2018 - цены закрепились на уровне 2017 года.

В 2019 - цены немного спали.

Посчитайте среднюю цену одного квадратного метра в 10 населённых пунктах с наибольшим числом объявлений. Выделите населённые пункты с самой высокой и низкой стоимостью квадратного метра. Эти данные можно найти по имени в столбце `locality_name`.

Найдем населенные пункты с самым большим количеством обьявлений

In [93]:
```
top10_data = data.groupby('locality_name')['locality_name'].count().sort_values(ascending=
print(top10_data)
```

```
locality_name
санкт-петербург      15635
поселок мурино         552
поселок шушары         439
всеволожск             398
пушкин                 369
колпино                338
поселок парголово      327
гатчина                307
деревня кудрово        299
выборг                 237
Name: locality_name, dtype: int64
```

Найдем населенный пункт из этого списка с **самой высокой "медианной" ценой квадратного метра**

In [94]:
```
data.query('locality_name in @top10_data.index').groupby('locality_name')['1m_price'].mean
```

Out[94]:
```
locality_name
санкт-петербург    114328.896375
Name: 1m_price, dtype: float64
```

Как мы и ожидали, это **Санкт-Петербург с медианной ценой 115 000 рублей за квадратный метр.**

Найдем населенный пункт из этого списка с самой низкой "медианной" ценой квадратного метра

In [95]:
```
data.query('locality_name in @top10_data.index').groupby('locality_name')['1m_price'].mean
```

Out[95]:
```
locality_name
выборг    58141.909153
Name: 1m_price, dtype: float64
```

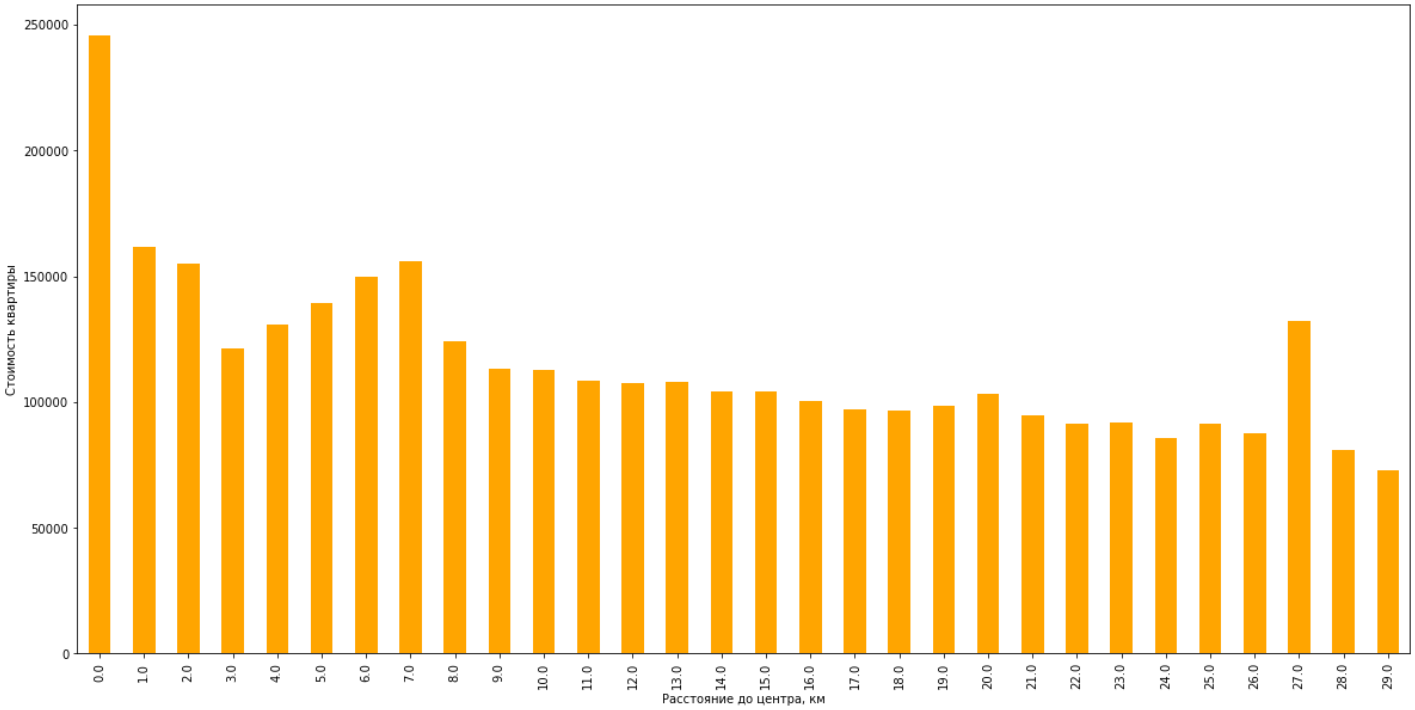
Самая низкая цена в Выборге с медианной ценой 58 000 рублей за квадратный метр.

Ранее мы посчитали расстояние до центра в километрах `cityCenters_nearest_km`.

Теперь выделим квартиры в Санкт-Петербурге с помощью столбца `locality_name` и вычислим среднюю цену каждого километра. Опишите, как стоимость объектов зависит от расстояния до центра города.

In [96]:
```
avg_price_km = data[data['locality_name'] == "санкт-петербург"].groupby('cityCenters_neare
print(avg_price_km)
ax = (
    avg_price_km.plot
    .bar(color='orange',figsize=(20,10))
    .set(xlabel = 'Расстояние до центра, км',ylabel='Стоимость квартиры')
)
```

```
cityCenters_nearest_km
0.0     245834.0
1.0     161608.0
2.0     155121.0
3.0     121508.0
4.0     130700.0
5.0     139237.0
6.0     149867.0
7.0     156177.0
8.0     124272.0
9.0     113159.0
10.0    112607.0
11.0    108388.0
12.0    107470.0
13.0    108099.0
14.0    104201.0
15.0    104231.0
16.0    100543.0
17.0     96970.0
18.0     96362.0
19.0     98658.0
20.0    103057.0
21.0     94470.0
22.0     91330.0
23.0     91829.0
24.0     85737.0
25.0     91531.0
26.0     87799.0
27.0    132116.0
28.0     81162.0
29.0     72953.0
Name: 1m_price, dtype: float64
```



Как мы видим, **самое дорогое жилье в центре города - со стоимостью 246 000 рублей за кв.м.**

и с каждым километром отдаления от центра - стоимость квадратного метра снижается.

На расстоянии 10 км от центра - стоимость составляет - 113 000 рублей.

На расстоянии 20 км от центра - цена за кв.м. - 103 000 рублей.

На столбчатой диаграмме видны всплески **на расстоянии 6-7 и 27км** от центра города - говорит о том, что скорее всего более дорогой/престижный район

Общий вывод

Задачей исследования было изучить архив объявлений за последние несколько лет по Санкт-Петербургу и Ленинградской области и выявить интересные особенности и зависимости, которые существуют на рынке недвижимости.

Были выявлены следующие факты:

- Основной процент продаж составляют квартиры общей площадью от 30 до 70 кв.м.
- Самые популярные квартиры с площадью - 30, 42, 44, 61, 63 и 80 кв. м
- Медианное значение цены на квартиру в Санкт-Петербурге и области - 4,65 млн.руб. Основной разброс цены от 3 до 6,8 млн руб.
- Самой большой популярностью пользуются 1-3 комнатные квартиры
- По количеству обьявлений - на 1м месте - однушки, 2м - двухкомнатные квартиры, 3е - 3-комнатные квартиры
- Наименьшей популярностью пользуются квартиры на 1м и последнем этаже
- Большинство обьявлений приходится на 5 и 9ти этажки
- Большая часть обьявлений в черте города, но на расстоянии от 9 до 16 км
- Ближайший парк находится на расстоянии от 300 до 600 метров.
- Большая часть объявлений выкладываются в будние дни. Пики - вторник, четверг
- Самые популярные месяца по количеству обьявлений о продаже квартиры - февраль, март, апрель, сентябрь, октябрь, ноябрь.
- Самый непопулярный месяц по количеству обьявлений - май

- Быстрыми продажими можно называть продажи до 1 (одного) месяца
- Долгие продажи - более (1) одного года

Далее мы изучили как различные факторы влиют на итогговую цену объекта недвижимости

- При увеличении жилой и общей площади квартиры - ценник на жилье увеличивается). Корелляция ~ 63%
- При увеличении количества комнат от 1 до 3х - цена недвижимости увеличивается. При дальнейшем увеличении - рост цены незначительный
- Самые дешевые цены на квартиры на 1м этаже.

Влияние даты обьявления -

- День публикации обьявления никак не влияет на цену недвижимости
- Месяц публикации обьявления никак не влияет на цену недвижимости
- Год публикации влияет на стоимость недвижимости (в зависимости от экономической ситуации в мире и в России). В кризис - цены меньше.

Ниже ТОП-3 населенных пунктов с самым большим количеством обьявлений -

1. санкт-петербург - 15705
2. поселок мурино - 556
3. поселок шушары - 440

Самая дорогая стоимость квадратного метра в Санкт-Петербурге. Медианная стоимость составляет - 114 000 рублей.

Самая низкая цена за квадратный метр в Выборге с ценой 58 000 рублей за кв.м.

Мы изучили цену влияние каждого километра близости к центру в Санкт-Петербурге получили следующую информацию -

1. Цена в центре СПБ - 245 000 рублей за кв.м.
2. Цена в 5 км от центра - 139 000 рублей за кв.м.
3. Цена в 10 км от центра - 113 000 рублей за кв.м.
4. Цена в 25 км от центра - 91 500 рублей за кв.м.

Как мы видим, самое дорогое жилье в центре города - со стоимостью 245 000 рублей за кв.м. и **с каждым километром отдаления от центра - стоимость квадратного метра снижается.**

Данные этого исследования и полученные выводы можно использовать для построения модели машинного обучения по анализу корректности цены в обьявлениях на недвижимость