



✓ Проект: Анализ резюме из HeadHunter

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 import plotly.express as px
```

✓ Исследование структуры данных

1. Прочитайте данные с помощью библиотеки Pandas. Совет: перед чтением обратите внимание на разделитель внутри файла.

```
1 try: from google.colab import drive
2 except ModuleNotFoundError:
3     !pip install --upgrade google-colab
4     from google.colab import drive
5 drive.mount('/content/gdrive')
6
7 import pandas as pd
8 hh = pd.read_csv('/content/gdrive/MyDrive/PY/1/dst-3.0_16_1_hh_database.csv', sep = ";", on_bad_lines='warn')      # 44 секунды
9 hhe = pd.read_csv('/content/gdrive/MyDrive/PY/ExchangeRates.csv', sep = ",", on_bad_lines='warn')
10 # hh = pd.read_csv('https://theants.ru/dst_3_0_16_1_hh_database.csv', sep = ";", on_bad_lines='warn')      # - 3:42
11 import pandas as pd

↗ Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
```

2. Выведите несколько первых (последних) строк таблицы, чтобы убедиться в том, что ваши данные не повреждены.
Ознакомьтесь с признаками и их структурой.

```
1 # Вывожу несколько первых строк таблицы, чтобы убедиться в том, что ваши данные не повреждены.
2 display(hh.head(3))
```

↗

	Пол, возраст	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	Последнее/ нынешнее место работы	Последняя/ нынешняя должность	Образование и ВУЗ	Обновл ри
0	Мужчина , 39 лет , родился 27 ноября 1979	29000 руб.	Системный администратор	Советск (Калининградская область) , не готов к...	частичная занятость, проектная работа, полная ...	гибкий график, полный день, сменный график, ва...	Опыт работы 16 лет 10 месяцев Август 2010 — п...	МАОУ "СОШ № 1 г.Немана"	Системный администратор	Неоконченное высшее образование 2000 Балтийск...	16.04
						гибкий	Опыт				

3. Выведите основную информацию о числе непустых значений в столбцах и их типах в таблице.

```
1 # вывожу информацию о числе непустых значений в столбцах и их типах в таблице.
2 hh.info()
```

↗

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44744 entries, 0 to 44743
Data columns (total 12 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Пол, возраст                          44744 non-null  object
1   ЗП                                     44744 non-null  object
2   Ищет работу на должность:             44744 non-null  object
3   Город, переезд, командировки          44744 non-null  object
4   Занятость                             44744 non-null  object
5   График                               44744 non-null  object
6   Опыт работы                           44576 non-null  object
7   Последнее/нынешнее место работы       44743 non-null  object
```

```
8 Последняя/нынешняя должность 44742 non-null object
9 Образование и ВУЗ 44744 non-null object
10 Обновление резюме 44744 non-null object
11 Авто 44744 non-null object
dtypes: object(12)
memory usage: 4.1+ MB
```

```
1 # Задание 2.5
2 # Сколько уникальных значений содержится в столбце «Опыт работы»?
3 hh['Опыт работы'].nunique()
```

↗ 44413

4. Обратите внимание на информацию о числе непустых значений.

```
1 # Вывожу информацию о числе непустых значений и пустых для наглядности
2 non_nan = hh.shape[0] - hh.isna().sum()
3 missing = non_nan[non_nan != hh.shape[0]]
4 pd.DataFrame({'Колонки с пропущенными значениями': missing.index, 'Не пустые': missing.values, 'Пустые': hh.shape[0]-missing.values, 'Итого': hh.shape[0]})
```

↗

	Колонки с пропущенными значениями	Не пустые	Пустые	Итого:	
0	Опыт работы	44576	168	44744	📊
1	Последнее/нынешнее место работы	44743	1	44744	
2	Последняя/нынешняя должность	44742	2	44744	

5. Выведите основную статистическую информацию о столбцах.

```
1 #Вывожу основную статистическую информацию о столбцах.
2 hh.describe()
```

↗

	Пол, возраст	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	Последнее/нынешнее место работы	Последняя/ нынешняя должность	Образование и ВУЗ:
count	44744	44744	44744	44744	44744	44744	44576	44743	44742	44744
unique	16003	690	14929	10063	38	47	44413	30214	16927	40148
top	Мужчина , 32 года	50000	Системный	Москва , не готов к	полная	полный	Опыт работы 10 лет 8 месяцев	Индивидуальное предпринимательство	Системный	Высшее образование 1987

```
1 # Задание 2.1
2 # Чему равна размерность таблицы? Введите её в виде кортежа (число строк, число столбцов).
3 hh.shape
```

↗ (44744, 12)

✓ Преобразование данных

1. Начнем с простого - с признака **"Образование и ВУЗ"**. Его текущий формат это: **<Уровень образования год выпуска ВУЗ специальность...>**. Например:

- Высшее образование 2016 Московский авиационный институт (национальный исследовательский университет)...
- Неоконченное высшее образование 2000 Балтийская государственная академия рыбопромыслового флота... Нас будет интересовать только уровень образования.


Создайте с помощью функции-преобразования новый признак **"Образование"**, который должен иметь 4 категории: "высшее", "неоконченное высшее", "среднее специальное" и "среднее".

Выполните преобразование, ответьте на контрольные вопросы и удалите признак "Образование и ВУЗ".

Совет: обратите внимание на структуру текста в столбце **"Образование и ВУЗ"**. Гарантируется, что текущий уровень образования соискателя всегда находится в первых 2ух слов и начинается с заглавной буквы. Воспользуйтесь этим.

Совет: проверяйте полученные категории, например, с помощью метода `unique()`

```
1 # Создаю с помощью функции-преобразования новый признак "Образование", который должен иметь 4 категории: "Высшее", "Неоконченное высшее", "Среднее специальное" и "Среднее".
2 # т.к. искомое гарантировано присутствует в начале строки, то ограничим поиск максимальной длиной 19 символов (Неоконченное высшее)
3 hh["Образование"] = hh["Образование и ВУЗ"].apply(lambda x: next(word for word in ["Высшее", "Неоконченное высшее", "Среднее специальное", "Среднее образование"] if word in x[:19]), None)
4
5 # Проверка полученных категории, с помощью метода unique()
6 display(list(hh["Образование"].unique()))
7
8 # Ответ на контрольный вопрос задания - Сколько соискателей имеет средний уровень образования (школьное образование)?
9 display(hh[hh["Образование"] == 'Среднее образование']["Образование"].value_counts())
10
11
12 # Удаляю столбец "Образование и ВУЗ"
13 hh = hh.drop(['Образование и ВУЗ'], axis=1)
```

 <ipython-input-16-c41b2486d36d>:3: FutureWarning: the convert_dtype parameter is deprecated and will be removed in a future version
hh["Образование"] = hh["Образование и ВУЗ"].apply(lambda x: next(word for word in ["Высшее", "Неоконченное высшее", "Среднее специальное", "Среднее образование"] if word in x[:19]), None)

count	
Образование	
Среднее образование	559

←  →

2. Теперь нас интересует столбец **"Пол, возраст"**. Сейчас он представлен в формате **<Пол , возраст , дата рождения >**. Например:

- Мужчина , 39 лет , родился 27 ноября 1979
- Женщина , 21 год , родилась 13 января 2000 Как вы понимаете, нам необходимо выделить каждый параметр в отдельный столбец.


Создайте два новых признака **"Пол"** и **"Возраст"**. При этом важно учесть:

- Признак пола должен иметь 2 уникальных строковых значения: 'М' - мужчина, 'Ж' - женщина.
- Признак возраста должен быть представлен целыми числами.

Выполните преобразование, ответьте на контрольные вопросы и удалите признак "**Пол, возраст**" из таблицы.


Совет: обратите внимание на структуру текста в столбце, в части на то, как разделены параметры пола, возраста и даты рождения между собой - символом ' , '. Гарантируется, что структура одинакова для всех строк в таблице. Вы можете воспользоваться этим.

```
1 # Создаю с помощью функции-преобразования новый признак ['Sex'] срез первой буквы из ['Пол, возраст']
2 hh['Sex'] = hh.apply(lambda r: r['Пол, возраст'][0], axis=1)
3 # Создаю с помощью функции-преобразования новый признак ['Age'] из среза ['Пол, возраст']
4 hh['Age'] = hh.apply(lambda r: int(r['Пол, возраст'].split(',')[1][2:4]), axis=1).astype('Int16')
5 # Удаляю столбец 'Пол, возраст'
6 hh = hh.drop(['Пол, возраст'], axis =1)
7
8 hh.head(3)
9
10 # hh['Sex'], hh['Age'] = zip(*hh.apply(lambda r: (r['Пол, возраст'][0], int(r['Пол, возраст'].split(',')[1][2:4])), axis=1))
```



ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	Последнее/нынешнее место работы	Последняя/нынешняя должность	Обновление резюме	Авто	Образование
0 29000 руб.	Системный администратор	Советск (Калининградская область), не готов к...	частичная занятость, проектная работа, полная ...	гибкий график, полный день, сменный график, ва...	Опыт работы 16 лет 10 месяцев Август 2010 — п...	МАОУ "СОШ № 1 г.Немана"	Системный администратор	16.04.2019 15:59	Имеется собственный автомобиль	Неоконч...

```
1 #Ответы на контрольные вопросы
2
3 display('Сколько процентов женских резюме представлено в наших данных? Ответ округлите до сотых', (hh["Sex"].value_counts(normalize=True) * 100).round(2)['Ж'])
4
5 display('Чему равен средний возраст соискателей? Ответ округлите до десятых', hh.Age.mean().round(1))
```



```
'Сколько процентов женских резюме представлено в наших данных? Ответ округлите до сотых '
19.07
'Чему равен средний возраст соискателей? Ответ округлите до десятых '
32.2
```

- Следующим этапом преобразуем признак "**Опыт работы**". Его текущий формат - это: **<Опыт работы: n лет m месяцев, периоды работы в различных компаниях...>**.

Из столбца нам необходимо выделить общий опыт работы соискателя в месяцах, новый признак назовем "Опыт работы (месяц)"

Для начала обсудим условия решения задачи:


- Во-первых, в данном признаке есть пропуски. Условимся, что если мы встречаем пропуск, оставляем его как есть (функция-преобразование возвращает NaN)

- Во-вторых, в данном признаке есть скрытые пропуски. Для некоторых соискателей в столбце стоит значения "Не указано". Их тоже обозначим как NaN (функция-преобразование возвращает NaN)
- В-третьих, нас не интересует информация, которая описывается после указания опыта работы (периоды работы в различных компаниях)
- В-четвертых, у нас есть проблема: опыт работы может быть представлен только в годах или только месяцах. Например, можно встретить следующие варианты:
 - Опыт работы 3 года 2 месяца...
 - Опыт работы 4 года...
 - Опыт работы 11 месяцев...
 - Учитывайте эту особенность в вашем коде

Учитывайте эту особенность в вашем коде

В результате преобразования у вас должен получиться столбец, содержащий информацию о том, сколько месяцев проработал соискатель. Выполните преобразование, ответьте на контрольные вопросы и удалите столбец **"Опыт работы"** из таблицы.

```
1 # Определяем функцию y2m для преобразования строки с опытом работы в месяцы
2 def y2m(i):
3     # Проверка, является ли входное значение строкой и длина строки больше 11 символов (если стоит "Не указано")
4     if not isinstance(i, str) or len(i) <= 11:
5         return None # Если условия не выполнены, возвращаем None
6
7     # Разбиваем строку по пробелу
8     i = i.split(' ')
9
10    # Проверяем первый символ четвертого элемента для определения вида опыта (лет или месяцев)
11    if i[3][0] in {'л', 'г'}:
12        # Если опыт указан в годах ('лет' или 'год'), преобразуем его в месяцы
13        return int(int(i[2]) * 12 + (int(i[4]) if i[4].isdigit() else 0))
14    elif i[3][0] == 'м':
15        # Если опыт указан в месяцах ('мес'), возвращаем значение как есть
16        return int(i[2])
17    else:
18        return 0 # Если формат не распознан, возвращаем 0
19
20 # Применение функции y2m к столбцу 'Опыт работы' и создание на его основе нового столбца 'Опыт работы (месяц)'
21 hh["Опыт работы (месяц)"] = hh["Опыт работы"].apply(y2m).astype('Int16')
22
23 # Удаление старого столбца 'Опыт работы'
24 hh = hh.drop(['Опыт работы'], axis=1)
25
26 # Ответ на Задание 3.3
27 print('Чему равен медианный опыт работы (в месяцах) в нашей таблице?', hh["Опыт работы (месяц)"].median())
```

 Чему равен медианный опыт работы (в месяцах) в нашей таблице? 100.0

1

4. Хорошо идем! Следующий на очереди признак "Город, переезд, командировки". Информация в нем представлена в следующем виде: **<Город , (метро) , готовность к переезду (города для переезда) , готовность к командировкам>**. В скобках указаны

необязательные параметры строки. Например, можно встретить следующие варианты:

- Москва , не готов к переезду , готов к командировкам
- Москва , м. Беломорская , не готов к переезду, не готов к командировкам
- Воронеж , готов к переезду (Сочи, Москва, Санкт-Петербург) , готов к командировкам

Создадим отдельные признаки **"Город"**, **"Готовность к переезду"**, **"Готовность к командировкам"**. При этом важно учесть:

- Признак **"Город"** должен содержать только 4 категории: "Москва", "Санкт-Петербург" и "город-миллионник" (их список ниже), остальные обозначьте как "другие".

Список городов-миллионников:

```
million_cities = ['Новосибирск', 'Екатеринбург', 'Нижний Новгород', 'Казань', 'Челябинск', 'Омск', 'Самара', 'Ростов-на-Дону', 'Уфа', 'Красноярск', 'Пермь', 'Воронеж', 'Волгоград']
```

Информация о метро, рядом с которым проживает соискатель нас не интересует.

- Признак **"Готовность к переезду"** должен иметь два возможных варианта: True или False. Обратите внимание, что возможны несколько вариантов описания готовности к переезду в признаке "Город, переезд, командировки". Например:

- ... , готов к переезду , ...
- ... , не готова к переезду , ...
- ... , готова к переезду (Москва, Санкт-Петербург, Ростов-на-Дону)
- ... , хочу переехать (США) , ...

Нас интересует только сам факт возможности или желания переезда.

- Признак **"Готовность к командировкам"** должен иметь два возможных варианта: True или False. Обратите внимание, что возможны несколько вариантов описания готовности к командировкам в признаке "Город, переезд, командировки". Например:

- ... , готов к командировкам , ...
- ... , готова к редким командировкам , ...
- ... , не готов к командировкам , ...

Нас интересует только сам факт готовности к командировке.

Еще один важный факт: при выгрузке данных у некоторых соискателей "потерялась" информация о готовности к командировкам. Давайте по умолчанию будем считать, что такие соискатели не готовы к командировкам.

Выполните преобразования и удалите столбец **"Город, переезд, командировки"** из таблицы.

Совет: обратите внимание на то, что структура текста может меняться в зависимости от указания ближайшего метро. Учите это, если будете использовать порядок слов в своей программе.

```
1 # Признак "Город" должен содержать только 4 категории: "Москва", "Санкт-Петербург" и "город-миллионник" (их список ниже), остальные обозначьте как "другие".
2 # Список городов-миллионников:
3 million_cities = ['Новосибирск', 'Екатеринбург', 'Нижний Новгород', 'Казань', 'Челябинск', 'Омск', 'Самара', 'Ростов-на-Дону', 'Уфа', 'Красноярск', 'Пермь', 'Воронеж', 'Волгоград']
4
5 #Создаю новый столбец "Город" на основании столбца "Город, переезд, командировки" с помощью лямбда функции - ишу города разделяя по запятым и удаляя лишние пробелы
6 hh['Город'] = hh["Город, переезд, командировки"].apply(
7     lambda x: 'город-миллионник' if x.split(',')[0].replace(' ', '') in million_cities
8     else x.split(" \\\n")[-1].replace(" ", "") if x.split(" \\\n")[-1].replace(" ", "") in ['Москва', 'Санкт-Петербург']
9     else 'другие'
```


Давайте создадим признаки-мигалки для каждой категории: если категория присутствует в списке желаемых соискателем, то в столбце на месте строки рассматриваемого соискателя ставится True, иначе - False.

Такой метод преобразования категориальных признаков называется One Hot Encoding и его схема представлена на рисунке ниже:

Схема One Hot Encoding преобразования

Занятость		полная занятость	частичная занятость	проектная работа	стажировка	волонтерство
полная занятость, частичная занятость, стажировка		True	True	False	True	False
полная занятость, проектная работа		True	False	True	False	False
стажировка, проектная работа, волонтерство		False	False	True	True	True

Выполните данное преобразование для признаков "Занятость" и "График", ответьте на контрольные вопросы, после чего удалите их из таблицы

```

1 #Создаю новые столбцы с булево
2 hh['полная занятость'] = hh["Занятость"].str.contains('полная занятость')
3 hh['частичная занятость'] = hh["Занятость"].str.contains('частичная занятость')
4 hh['проектная работа'] = hh["Занятость"].str.contains('проектная работа')
5 hh['стажировка'] = hh["Занятость"].str.contains('стажировка')
6 hh['волонтерство'] = hh["Занятость"].str.contains('волонтерство')
7
8 # Удаляю столбец согласно заданию
9 hh = hh.drop(["Занятость"], axis=1)

1 #Создаю новые столбцы с булево
2 # print(hh['График'].value_counts())
3
4 hh['полный день'] = hh["График"].str.contains('полный день')
5 hh['гибкий график'] = hh["График"].str.contains('гибкий график')
6 hh['сменный график'] = hh["График"].str.contains('сменный график')
7 hh['удаленная работа'] = hh["График"].str.contains('удаленная работа')
8 hh['вахтовый метод'] = hh["График"].str.contains('вахтовый метод')
9
10 # Удаляю столбец согласно заданию
11 hh = hh.drop(["График"], axis=1)

```

```
1 # Ответы на 3.5
2 print('Сколько людей ищут проектную работу и волонтерство (в обоих столбцах стоит True):', ((hh["проектная работа"] & hh["волонтерство"]).sum()))
3 print('Сколько людей хотят работать вахтовым методом и с гибким графиком (в обоих столбцах стоит True)?:', ((hh["вахтовый метод"] & hh["гибкий график"]).sum()))
```

➡ Сколько людей ищут проектную работу и волонтерство (в обоих столбцах стоит True): 436
Сколько людей хотят работать вахтовым методом и с гибким графиком (в обоих столбцах стоит True)?: 2311

6. (2 балла) Наконец, мы добрались до самого главного и самого важного - признака заработной платы "**ЗП**". В чем наша беда? В том, что помимо желаемой заработной платы соискатель указывает валюту, в которой он бы хотел ее получать, например:

- 30000 руб.
- 50000 грн.
- 550 USD

Нам бы хотелось видеть заработную плату в единой валюте, например, в рублях. Возникает вопрос, а где взять курс валют по отношению к рублю?

На самом деле язык Python имеет в арсенале огромное количество возможностей получения данной информации, от обращения к API Центробанка, до использования специальных библиотек, например `ruscbf`. Однако, это не тема нашего проекта.

Поэтому мы пойдем в лоб: обратимся к специальным интернет-ресурсам для получения данных о курсе в виде текстовых файлов. Например, [MDF.RU](https://mdf.ru/), данный ресурс позволяет удобно экспортировать данные о курсах различных валют и акций за указанные периоды в виде csv файлов. Мы уже сделали выгрузку курсов валют, которые встречаются в наших данных за период с 29.12.2017 по 05.12.2019. Скачать ее вы можете **на платформе**

Создайте новый DataFrame из полученного файла. В полученной таблице нас будут интересовать столбцы:

- "currency" - наименование валюты в ISO кодировке,
- "date" - дата,
- "proportion" - пропорция,
- "close" - цена закрытия (последний зафиксированный курс валюты на указанный день).

Перед вами таблица соответствия наименований иностранных валют в наших данных и их общепринятых сокращений, которые представлены в нашем файле с курсами валют. Пропорция - это число, за сколько единиц валюты указан курс в таблице с курсами. Например, для казахстанского тенге курс на 20.08.2019 составляет 17.197 руб. за 100 тенге, тогда итоговый курс равен - $17.197 / 100 = 0.17197$ руб за 1 тенге. Воспользуйтесь этой информацией в ваших преобразованиях.

	Наименование валюты в данных	Наименование валюты в ISO кодировке	Пропорция	Расшифровка
	грн	UAH	10	гривна
	USD	USD	1	доллар
	EUR	EUR	1	евро
	белруб	BYN	1	белорусский рубль
	KGS	KGS	10	киргизский сом
	сум	UZS	10000	узбекский сум
	AZN	AZN	1	азербайджанский манат

1

	KZT	KZT	100	казахстанский тенге
--	-----	-----	-----	---------------------

Итак, давайте обсудим возможный алгоритм преобразования:

1. Перевести признак "Обновление резюме" из таблицы с резюме в формат datetime и достать из него дату. В тот же формат привести признак "date" из таблицы с валютами.
2. Выделить из столбца "ЗП" сумму желаемой заработной платы и наименование валюты, в которой она исчисляется. Наименование валюты перевести в стандарт ISO согласно с таблицей выше.
3. Присоединить к таблице с резюме таблицу с курсами по столбцам с датой и названием валюты (подумайте, какой тип объединения надо выбрать, чтобы в таблице с резюме сохранились данные о заработной плате, изначально представленной в рублях). Значение close для рубля заполнить единицей 1 (курс рубля самого к себе)
4. Умножить сумму желаемой заработной платы на присоединенный курс валюты (close) и разделить на пропорцию (обратите внимание на пропуски после объединения в этих столбцах), результат занести в новый столбец "ЗП (руб)".

```

1 # Создаю функцию конвертации валют          #10 секунд выполнение кода
2 def convert_salary(row, exchange_rates):
3     # разделяю строку ЗП на число и имя валюты
4     salary, currency = row['ЗП'].split()[:2]
5     salary = int(salary)
6     currency = currency.rstrip('.').upper()
7
8     # Конвертация 'БЕЛ.РУБ' для удобства дальнейшей работы
9     if currency == 'БЕЛ.РУБ':
10         currency = 'BYN'
11     # Если 'РУБ', то оставляем как есть
12     if currency == 'РУБ':
13         return salary
14     else:
15         # перевод в формат даты
16         date = pd.to_datetime(row['Обновление резюме'], format='%d.%m.%Y %H:%M').date()
17
18         # Попытка найти курс обмена
19         exchange_rate_row = exchange_rates.loc[(exchange_rates['date'] == date) & (exchange_rates['currency'].str.upper() == currency)]
20         if not exchange_rate_row.empty:
21             return (salary * exchange_rate_row['close'].iloc[0] / exchange_rate_row['proportion'].iloc[0]).round(2)
22         else:
23             return None
24
25 # Преобразование колонки 'date' в формат даты
26 hhe['date'] = pd.to_datetime(hhe['date'], format='%d/%m/%y').dt.date
27
28 # Передача датафрейма hhe в функцию в качестве аргумента, чтобы избежать его постоянного поиска в глобальном пространстве
29 hh['ЗП (руб)'] = hh.apply(convert_salary, axis=1, exchange_rates=hhe)
30
31 # Удаляю столбец ЗП
32 hh = hh.drop(["ЗП"], axis=1)

1 # Ответ на Задание 3.6
2
3 display('Чему равна желаемая медианная заработная плата соискателей в нашей таблице (в рублях)? Ответ приведите в тысячах рублей и округлите до целого:',
4 int(hh['ЗП (руб)'].median()/1000))

→ 'Чему равна желаемая медианная заработная плата соискателей в нашей таблице (в рублях)? Ответ приведите в тысячах рублей и округлите до целого:'
sq

1

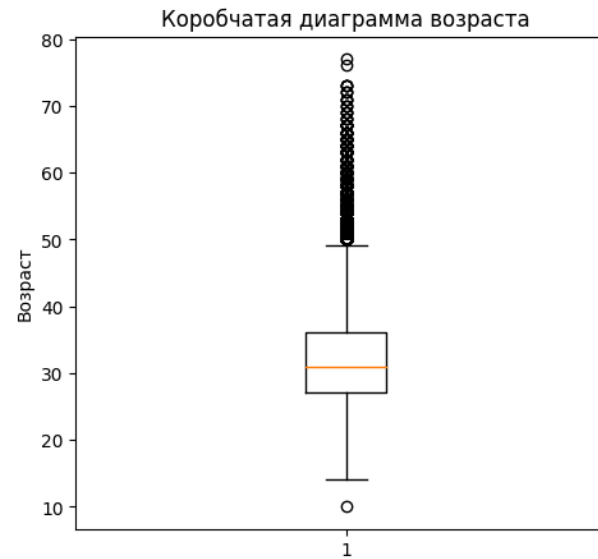
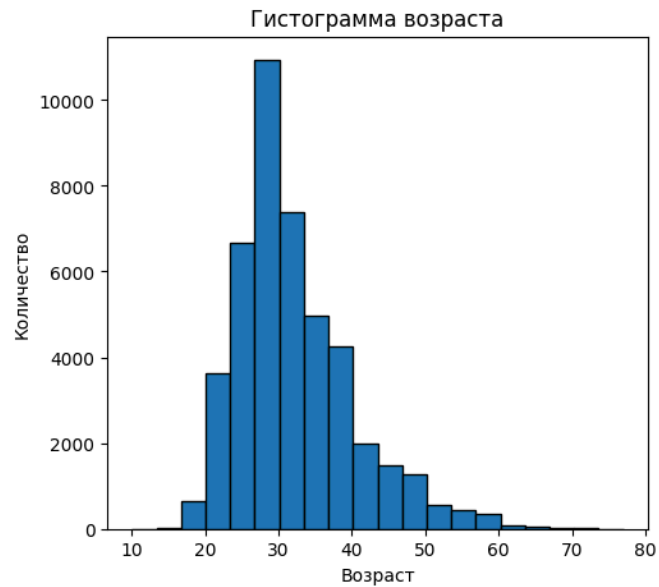
```

✓ Исследование зависимостей в данных

1. Постройте распределение признака **"Возраст"**. Опишите распределение, отвечая на следующие вопросы: чему равна мода распределения, каковы предельные значения признака, в каком примерном интервале находится возраст большинства соискателей? Есть ли аномалии для признака возраста, какие значения вы бы причислили к их числу?

Совет: постройте гистограмму и коробчатую диаграмму рядом.

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Создаем фигуру с двумя подграфиками: гистограмма и коробчатая диаграмма
5 fig, axs = plt.subplots(1, 2, figsize=(12, 5))
6
7 # Гистограмма распределения возраста
8 axs[0].hist(hh['Age'].dropna(), bins=20, edgecolor='black')
9 axs[0].set_title('Гистограмма возраста')
10 axs[0].set_xlabel('Возраст')
11 axs[0].set_ylabel('Количество')
12
13 # Коробчатая диаграмма для возраста
14 axs[1].boxplot(hh['Age'].dropna())
15 axs[1].set_title('Коробчатая диаграмма возраста')
16 axs[1].set_ylabel('Возраст')
17
18 plt.show()
19
20 # Выводы
21 print('Мода возраста:', hh['Age'].mode()[0])
22 print('Минимальный возраст:', hh['Age'].min())
23 print('Максимальный возраст:', hh['Age'].max())
24 print('Возраст большинства соискателей находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%):', int(hh['Age'].describe()['25%']), '-', int(hh['Age'].describe()['75%']), 'лет')
25 # print(f"Верхний квартиль (75%): {int(hh['Age'].describe()['75%'])}")
26 print('Аномалии для признака возраста - минимальный возраст', hh['Age'].min(), 'лет.', 'Минимальный возраст для работы - 14 лет (исключение работа в цирке или театре)')
27
28 print('Задание 4.2 - Чему равен максимальный опыт работы (в месяцах)?', hh['Опыт работы (месяц)'].max())
```



Мода возраста: 30

Минимальный возраст: 10

Максимальный возраст: 77

Возраст большинства соискателей находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%): 27 - 36 лет

Аномалии для признака возраста - минимальный возраст 10 лет. Минимальный возраст для работы - 14 лет (исключение работа в цирке

Задание 4.2 - Чему равен максимальный опыт работы (в месяцах)? 1188

ваши выводы по графику здесь

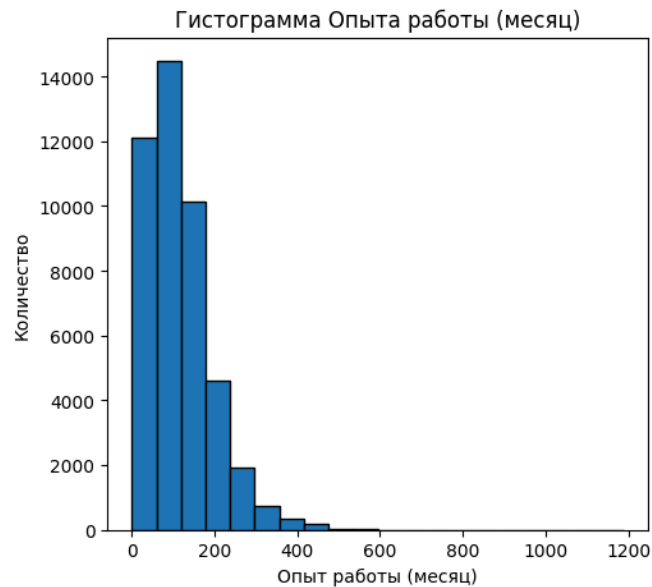
- Мода возраста: 30
- Минимальный возраст: 10
- Максимальный возраст: 77
- Возраст большинства соискателей находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%): 27 - 36 лет
- Аномалии для признака возраста - минимальный возраст 10 лет. Минимальный возраст для работы - 14 лет (исключение работа в цирке или театре)

1

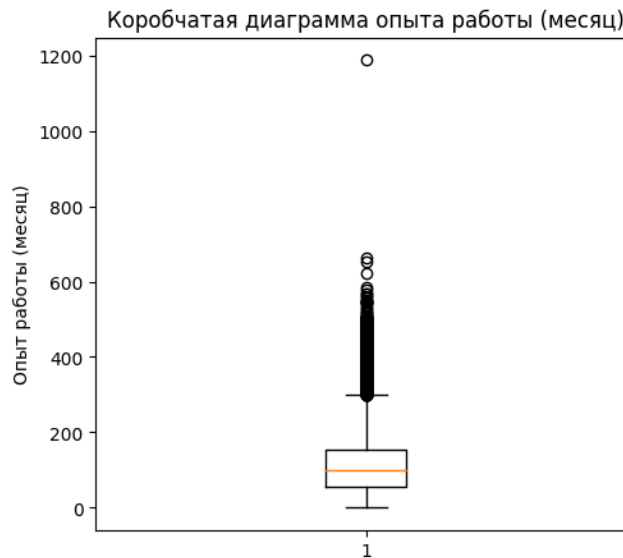
2. Постройте распределение признака "**Опыт работы (месяц)**". Опишите данное распределение, отвечая на следующие вопросы: чему равна мода распределения, каковы предельные значения признака, в каком примерном интервале находится опыт работы большинства соискателей? Есть ли аномалии для признака опыта работы, какие значения вы бы причислили к их числу?

Совет: постройте гистограмму и коробчатую диаграмму рядом.

```
1 # ваш код здесь
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 # Создаем фигуру с двумя подграфиками: гистограмма и коробчатая диаграмма
6 fig, axs = plt.subplots(1, 2, figsize=(12, 5))
7
8 # Гистограмма распределения Опыта работы (месяц)
9 axs[0].hist(hh['Опыт работы (месяц)'].dropna(), bins=20, edgecolor='black')
10 axs[0].set_title('Гистограмма Опыта работы (месяц)')
11 axs[0].set_xlabel('Опыт работы (месяц)')
12 axs[0].set_ylabel('Количество')
13
14 # Коробчатая диаграмма для Опыта работы (месяц)
15 axs[1].boxplot(hh['Опыт работы (месяц)'].dropna())
16 axs[1].set_title('Коробчатая диаграмма опыта работы (месяц)')
17 axs[1].set_ylabel('Опыт работы (месяц)')
18
19 plt.show()
20
21 # Выводы
22 print('Мода столбца Опыт работы (месяц):', hh['Опыт работы (месяц)'].mode()[0])
23 print('Минимальное значение:', hh['Опыт работы (месяц)'].min())
24 print('Максимальное значени:', hh['Опыт работы (месяц)'].max())
25 print('Опыт работы находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%):', int(hh['Опыт работы (месяц)'].describe()['25%']), '-', int(hh['Опыт работы (месяц)'].describe()['75%']), 'м')
26 print('Аномалии для признака опыта работы - максимальный опыт работы:', int(hh['Опыт работы (месяц)'].max() / 12), 'лет.')
27
```



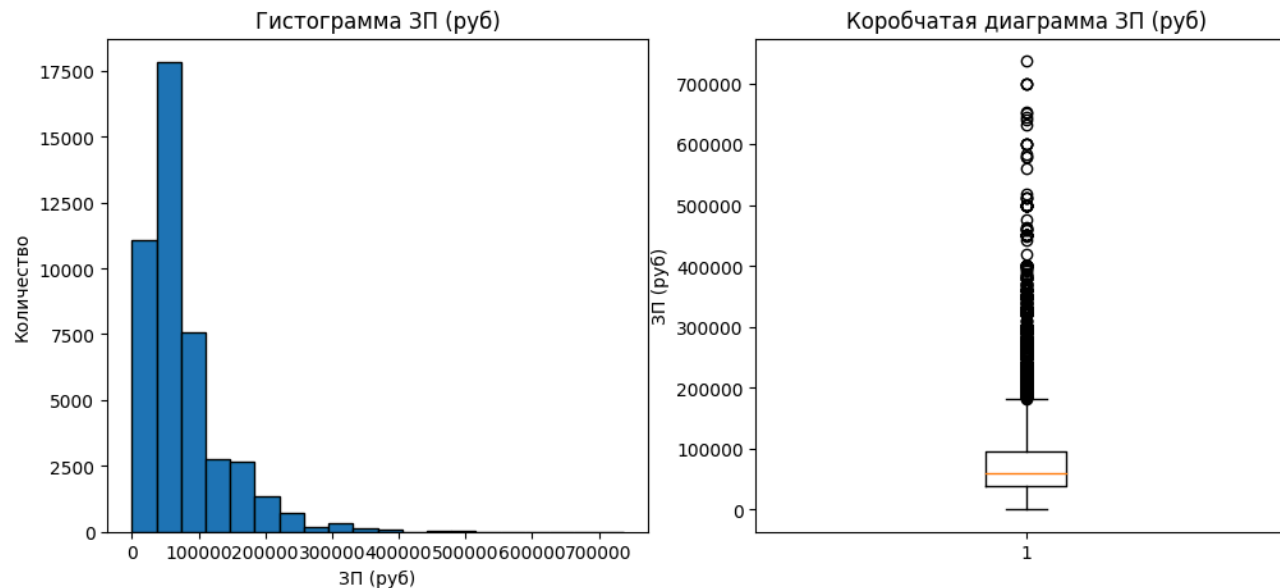
Мода столбца Опыт работы (месяц): 81
Минимальное значение: 1
Максимальное значени: 1188
Опыт работы находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%): 57 - 154 месяцев
Аномалии для признака опыта работы - максимальный опыт работы: 99 лет.




```

6 fig, axs = plt.subplots(1, 2, figsize=(12, 5))
7
8 # Гистограмма распределения ЗП (руб) - уберу топ 10, с ними график бесполезен
9 axs[0].hist((hh['ЗП (руб)'][~hh['ЗП (руб)'].isin(hh['ЗП (руб)'].nlargest(10))]).dropna(), bins=20, edgecolor='black')
10 axs[0].set_title('Гистограмма ЗП (руб)')
11 axs[0].set_xlabel('ЗП (руб)')
12 axs[0].set_ylabel('Количество')
13
14 # Коробчатая диаграмма для ЗП (руб)
15 axs[1].boxplot(hh['ЗП (руб)'][~hh['ЗП (руб)'].isin(hh['ЗП (руб)'].nlargest(10))]).dropna())
16 axs[1].set_title('Коробчатая диаграмма ЗП (руб)')
17 axs[1].set_ylabel('ЗП (руб)')
18
19 plt.show()
20
21 # Выводы
22 print('Мода столбца Опыт работы (месяц):', hh['ЗП (руб)'].mode()[0])
23 print('Минимальное значение:', hh['ЗП (руб)'].min())
24 print('Максимальное значение:', hh['ЗП (руб)'].max())
25 print('ЗП (руб) находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%):', int(hh['ЗП (руб)'].describe()['25%']), '-', int(hh['ЗП (руб)'].describe()['75%']), 'руб')
26 print('Аномалии для признака ЗП (руб): \n - Присутствуют максимальные зарплаты, они не дают возможности корректно отобразить графики:', list(hh['ЗП (руб)'].nlargest(10)), 'руб', '\n - Присутствуют а
27

```



Мода столбца Опыт работы (месяц): 50000.0

Минимальное значение: 1.0

Максимальное значени: 24304876.0

ЗП (руб) находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%): 38000 - 95000 руб

Аномалии для признака ЗП (руб):

- Присутствуют максимальные зарплаты, они не дают возможности корректно отобразить графики: [24304876.0, 7675224.0, 3000000.0,
- Присутствуют аномально низкие зарплаты {1.0, 18.0, 25.0, 26.99, 30.0, 40.0, 45.0, 50.0, 55.0, 59.55, 60.0, 63.98, 64.24, 64.2

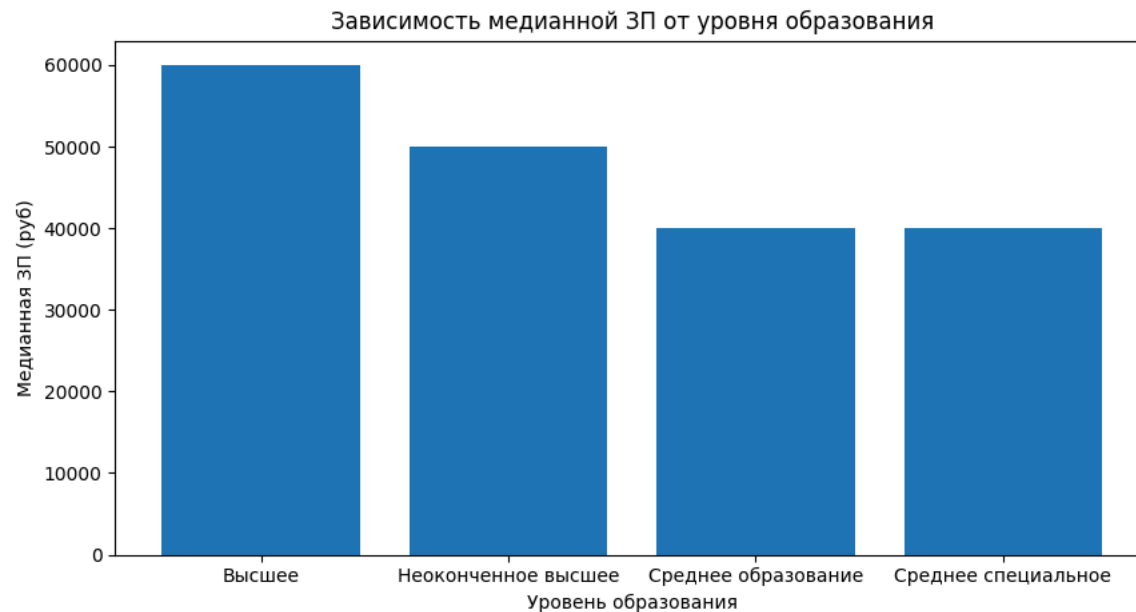
ваши выводы здесь

- Мода столбца Опыт работы (месяц): 50000.0
- Минимальное значение: 1.0
- Максимальное значени: 24304876.0
- ЗП (руб) находится в диапазоне: Нижний квартиль (25%) - Верхний квартиль (75%): 38000 - 95000 руб
- Аномалии для признака ЗП (руб):
 - Присутствуют максимальные зарплаты, они не дают возможности корректно отобразить графики: [24304876.0, 7675224.0, 3000000.0, 2500000.0, 1750000.0, 1000000.0, 923983.0, 900000.0, 800000.0, 750000.0] руб
 - Присутствуют аномально низкие зарплаты {1.0, 18.0, 25.0, 26.99, 30.0, 40.0, 45.0, 50.0, 55.0, 59.55, 60.0, 63.98, 64.24, 64.25, 64.68, 65.0, 65.33, 63.96, 70.0, 71.92, 85.0, 90.0, 100.0, 101.0, 110.0, 111.0, 120.0} руб

4. Постройте диаграмму, которая показывает зависимость **медианной** желаемой заработной платы ("**ЗП (руб)**") от уровня образования ("**Образование**"). Используйте для диаграммы данные о резюме, где желаемая заработная плата меньше 1 млн рублей.

Сделайте выводы по представленной диаграмме: для каких уровней образования наблюдаются наибольшие и наименьшие уровни желаемой заработной платы? Как вы считаете, важен ли признак уровня образования при прогнозировании заработной платы?

```
1 # Применение условия заработная плата меньше 1 млн рублей к выбранным колонкам и расчет медианы
2 median_salary = hh[['Образование', 'ЗП (руб)']][hh['ЗП (руб)' < 1000000].groupby('Образование').median().reset_index()
3
4 # Построение графика
5 plt.figure(figsize=(10,5))
6 plt.bar(median_salary['Образование'], median_salary['ЗП (руб)'])
7 plt.xlabel('Уровень образования')
8 plt.ylabel('Медианная ЗП (руб)')
9 plt.title('Зависимость медианной ЗП от уровня образования')
10 plt.show()
```



ваши выводы здесь

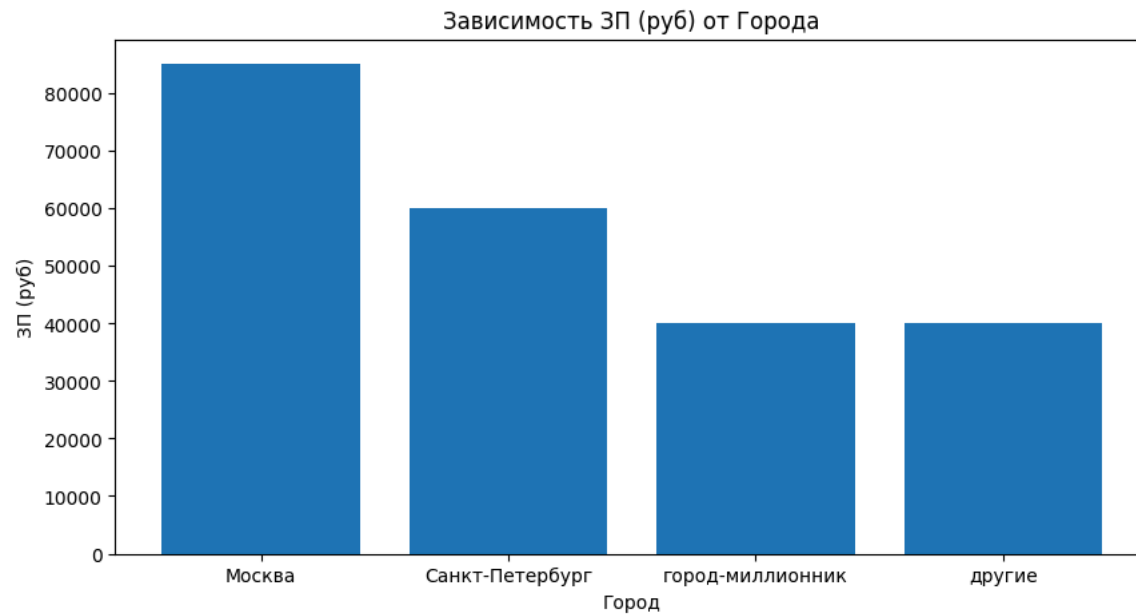
- для уровней образования Высшее и Неоконченное высшее наблюдаются наибольшие и для уровней образования Среднее и Среднее специальной - наименьшие уровни желаемой заработной платы
- признак уровня образования при прогнозировании заработной платы важен

1

5. Постройте диаграмму, которая показывает распределение желаемой заработной платы ("**ЗП (руб)**") в зависимости от города ("**Город**"). Используйте для диаграммы данные о резюме, где желая заработная плата меньше 1 млн рублей.

Сделайте выводы по полученной диаграмме: как соотносятся медианные уровни желаемой заработной платы и их размах в городах? Как вы считаете, важен ли признак города при прогнозировании заработной платы?

```
1 # Применение условия заработная плата меньше 1 млн рублей к выбранным колонкам и расчет медианы
2 median_salary = hh[['Город', 'ЗП (руб)']][hh['ЗП (руб)' < 1000000].groupby('Город').median().reset_index()]
3
4 # Построение графика
5 plt.figure(figsize=(10,5))
6 plt.bar(median_salary['Город'], median_salary['ЗП (руб)'])
7 plt.xlabel('Город')
8 plt.ylabel('ЗП (руб)')
9 plt.title('Зависимость ЗП (руб) от Города')
10 plt.show()
```



ваши выводы здесь

- медианные уровни желаемой заработной платы и их размах в городах зависят от статуса и размера города
- признак города при прогнозировании заработной платы важен

1


6. Постройте **многоуровневую столбчатую диаграмму**, которая показывает зависимость медианной заработной платы ("**ЗП (руб)**") от признаков "**Готовность к переезду**" и "**Готовность к командировкам**". Проанализируйте график, сравнив уровень заработной платы в категориях.

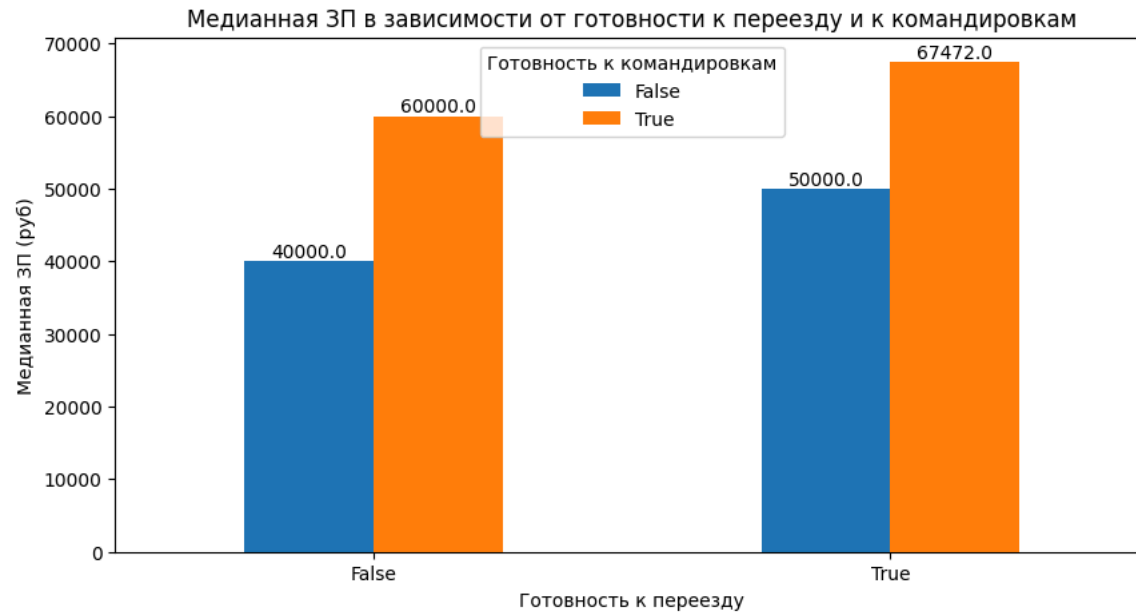
```
1 # Группировка данных для создания многоуровневой столбчатой диаграммы
2 pivot_df = hh.pivot_table(values='ЗП (руб)', index='Готовность к переезду', columns='Готовность к командировкам', aggfunc=np.median)
3
4 # Построение многоуровневой столбчатой диаграммы
5 ax = pivot_df.plot(kind='bar', figsize=(10, 5))
6
7 # Добавление меток с максимальными значениями
8 for p in ax.patches:
9     ax.annotate((p.get_height()), (p.get_x() + p.get_width() / 2, p.get_height()), ha='center', va='bottom')
10
11 plt.title('Медианная ЗП в зависимости от готовности к переезду и к командировкам')
12 plt.xlabel('Готовность к переезду')
13 plt.ylabel('Медианная ЗП (руб)')
14 plt.legend(title='Готовность к командировкам', labels=['False', 'True'], loc='upper center')
15 plt.xticks(rotation=0)
```

```

16 plt.show()
17
18 print('Ответ на Задание 4.6 \nЧему равна желаемая медианная заработная плата соискателей, готовых и к переезду, и к командировкам? \nОтвет приведите в тысячах, округлив до целого (например, 45):\n',
19       int(hh.groupby(['Готовность к переезду', 'Готовность к командировкам'], as_index=False)['ЗП (руб)'].median().max()['ЗП (руб)']/1000))

```

 <ipython-input-32-968f06104033>:2: FutureWarning: The provided callable <function median at 0x781738b5b400> is currently using [pivot_df = hh.pivot_table(values='ЗП (руб)', index='Готовность к переезду', columns='Готовность к командировкам', aggfunc=np.m



Ответ на Задание 4.6
 Чему равна желаемая медианная заработная плата соискателей, готовых и к переезду, и к командировкам?
 Ответ приведите в тысячах, округлив до целого (например, 45):
 67

ваши выводы здесь

- Зарплата выше у соискателей готовых к командировкам и ещё выше если соискатель готов к переезду

1

7. Постройте сводную таблицу, иллюстрирующую зависимость **медианной** желаемой заработной платы от возраста ("**Возраст**") и образования ("**Образование**"). На полученной сводной таблице постройте **тепловую карту**. Проанализируйте тепловую карту, сравнив показатели внутри групп.

```


1 # Создание сводной таблицы
2 pivot_table = hh.pivot_table(values='ЗП (руб)', index='Age', columns='Образование', aggfunc=np.median, fill_value=0)
3
4 # Построение тепловой карты
5 plt.figure(figsize=(10, 8))

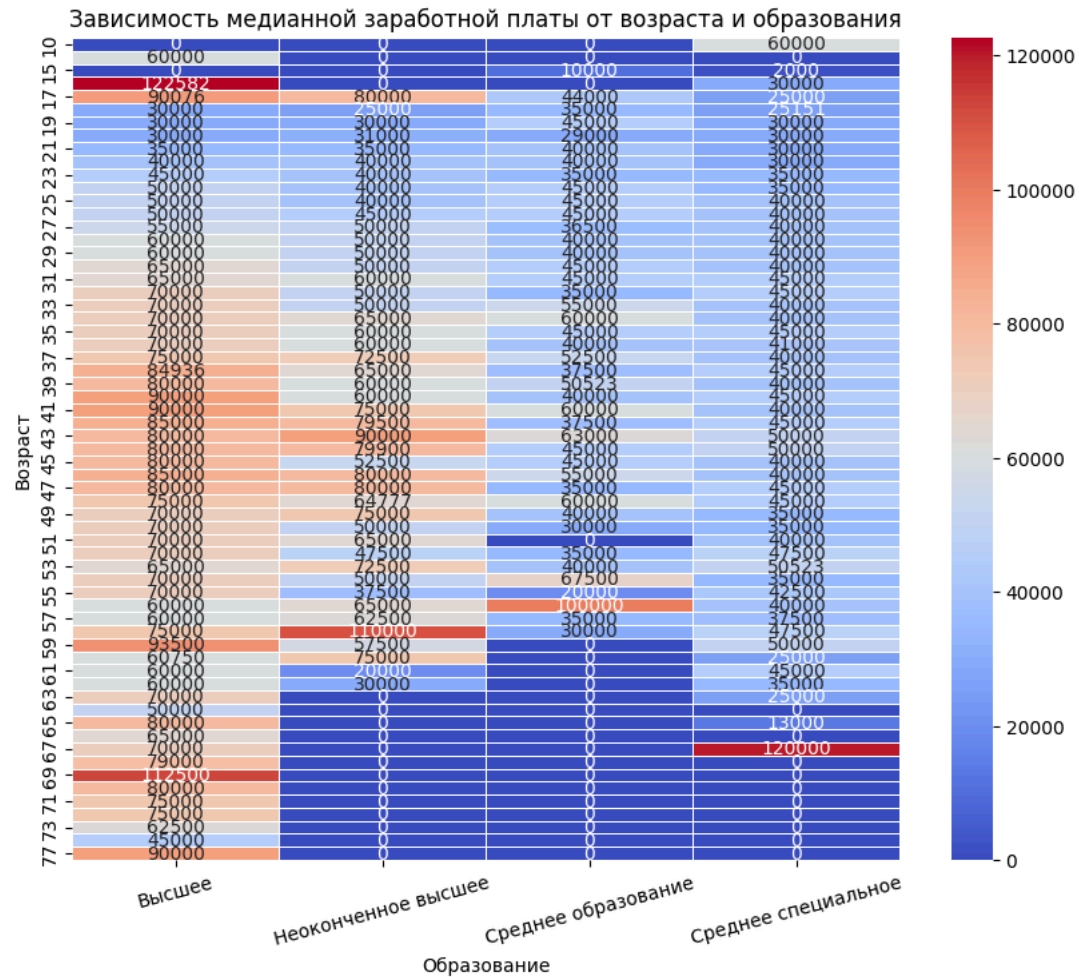
```

```

6 sns.heatmap(pivot_table, annot=True, fmt="0.0t", cmap="coolwarm", linewidths=.5)
7 plt.title('Зависимость медианной заработной платы от возраста и образования')
8 plt.ylabel('Возраст')
9 plt.xlabel('Образование')
10 plt.xticks(rotation=15)
11 plt.show()

```

 <ipython-input-33-dadfebcd2a08>:2: FutureWarning: The provided callable <function median at 0x781738b5b400> is currently using [pivot_table = hh.pivot_table(values='ЗП (руб)', index='Age', columns='Образование', aggfunc=np.median, fill_value=0)



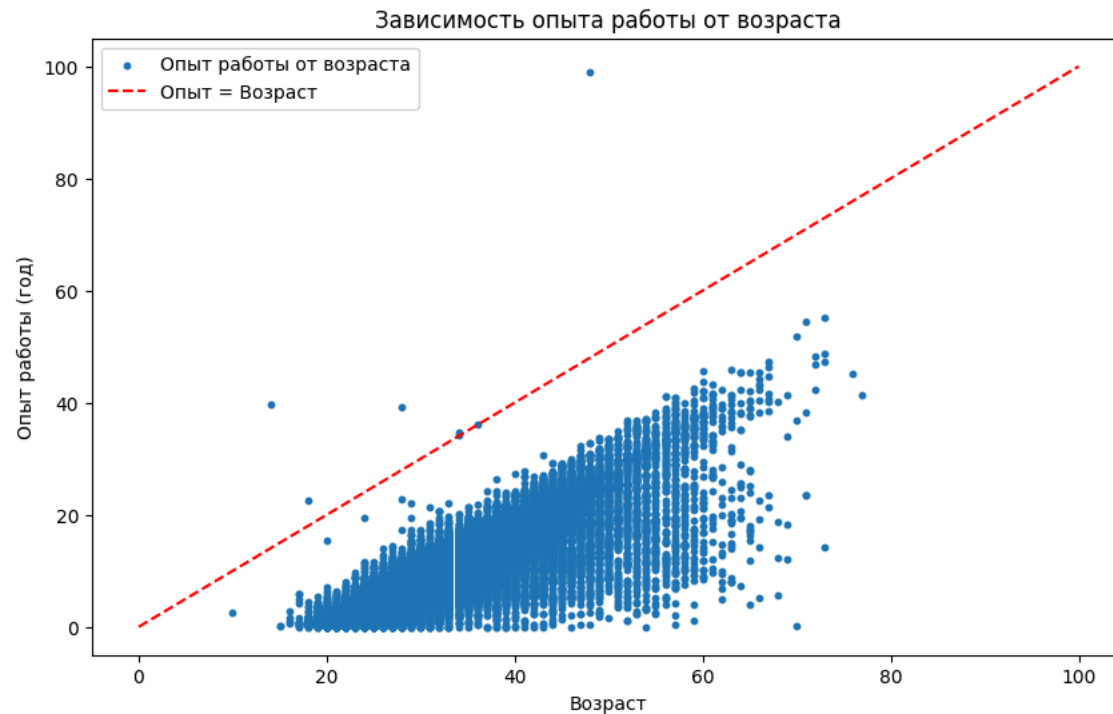
ваши выводы здесь

- Медианная желаемая заработная плата выше у соискателей с Высшим и неоконченным высшим образованием. Соискатели с Высшим образованием с возрастом старше 60 лет все еще катируются на рынке труда (или продолжают слать резюме)

8. Постройте **диаграмму рассеяния**, показывающую зависимость опыта работы ("**Опыт работы (месяц)**") от возраста ("**Возраст**").

Опыт работы переведите из месяцев в года, чтобы признаки были в едином масштабе. Постройте на графике дополнительно прямую, проходящую через точки (0, 0) и (100, 100). Данная прямая соответствует значениям, когда опыт работы равен возрасту человека. Точки, лежащие на этой прямой и выше нее - аномалии в наших данных (опыт работы больше либо равен возрасту соискателя)

```
1 # Переводим опыт работы из месяцев в года
2 hh['Опыт работы (год)'] = hh['Опыт работы (месяц)'] / 12
3
4 # Построение диаграммы рассеяния
5 plt.figure(figsize=(10, 6))
6 plt.scatter(hh['Age'], hh['Опыт работы (год)'].fillna(0),s=10,label='Опыт работы от возраста')
7
8 # Добавляем прямую, проходящую через точки (0, 0) и (100, 100)
9 plt.plot([0, 100], [0, 100], color='red', linestyle='--', label='Опыт = Возраст')
10
11 # Добавляем подписи и легенду
12 plt.xlabel('Возраст')
13 plt.ylabel('Опыт работы (год)')
14 plt.title('Зависимость опыта работы от возраста')
15 plt.legend()
16
17 plt.show()
18
19 print('Задание 4.8 Сколько точек лежат строго выше построенной прямой?',
20       sum(hh[['Age', 'Опыт работы (год)']].dropna()['Опыт работы (год)'] > hh[['Age', 'Опыт работы (год)']].dropna()['Age']))
```



Задание 4.8 Сколько точек лежат строго выше построенной прямой? 7

ваши выводы здесь

- после 60 количество желающих работать снижается
- тенденция накопления опыта выглядит вполне реально

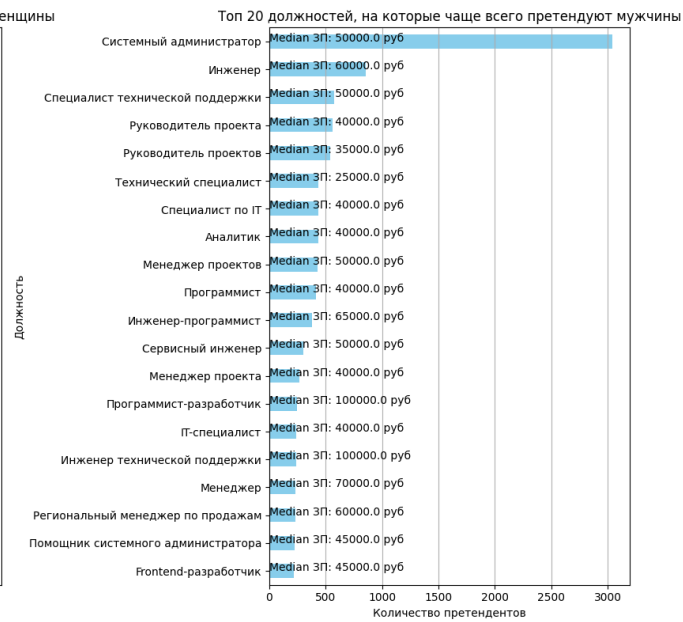
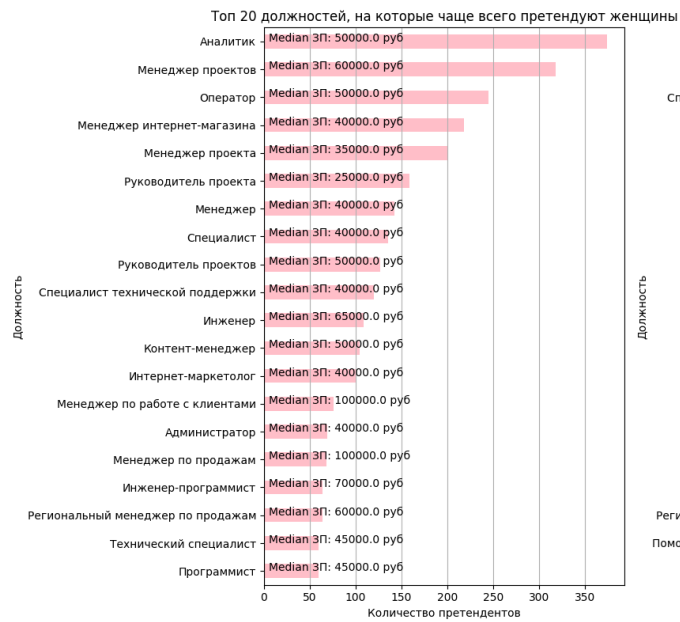
Дополнительные баллы

Для получения 2 дополнительных баллов по разведывательному анализу постройте еще два любых содержательных графика или диаграммы, которые помогут проиллюстрировать влияние признаков/взаимосвязь между признаками/распределения признаков. Приведите выводы по ним. Желательно, чтобы в анализе участвовали признаки, которые мы создавали ранее в разделе "Преобразование данных".

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 plt.figure(figsize=(16, 8))
5 # Подсчитываем количество претендентов на каждую должность для женщин
6 plt.subplot(1, 2, 1)
7 hh[hh['Sex'] == 'Ж']['Ищет работу на должность:'].value_counts().nlargest(20).iloc[::-1].plot(kind='barh', color='pink')
8 plt.xlabel('Количество претендентов')
9 plt.ylabel('Должность')
10 plt.title('Топ 20 должностей, на которые чаще всего претендуют женщины')
```



```
11 plt.grid(axis='x')
12
13 # Вставляем медианные значения 'ЗП (руб)' для каждой должности в топ 20
14 for idx, val in enumerate(hh['Ищет работу на должность:'].value_counts().nlargest(20).index):
15     median_salary = hh[hh['Ищет работу на должность:'] == val]['ЗП (руб)'].median()
16     plt.text(5, idx, f'Median ЗП: {median_salary} руб', color='black')
17
18 # Подсчитываем количество претендентов на каждую должность для мужчин
19 plt.subplot(1, 2, 2)
20 hh[hh['Sex'] == 'М']['Ищет работу на должность:'].value_counts().nlargest(20).iloc[::-1].plot(kind='barh', color='skyblue')
21 plt.xlabel('Количество претендентов')
22 plt.ylabel('Должность')
23 plt.title('Топ 20 должностей, на которые чаще всего претендуют мужчины')
24 plt.grid(axis='x')
25
26 # Вставляем медианные значения 'ЗП (руб)' для каждой должности в топ 20
27 for idx, val in enumerate(hh['Ищет работу на должность:'].value_counts().nlargest(20).index):
28     median_salary = hh[hh['Ищет работу на должность:'] == val]['ЗП (руб)'].median()
29     plt.text(5, idx, f'Median ЗП: {median_salary} руб', color='black')
30
31 plt.tight_layout()
32 plt.show()
```



ВАШИ ВЫВОДЫ ЗДЕСЬ

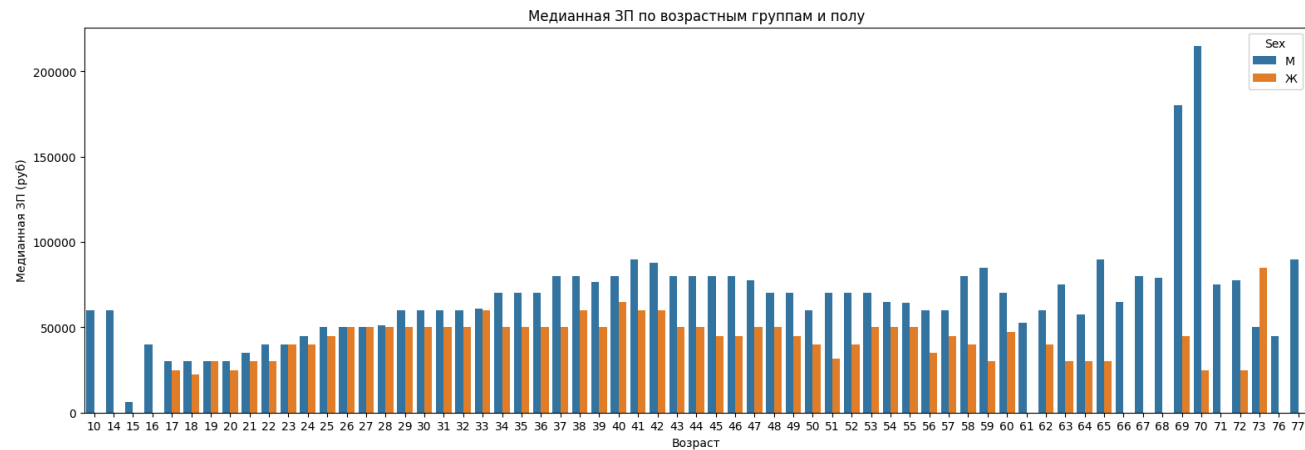
- Топ 20 должностей показывает, где хотят работать мужчины и женщины и сколько они хотят получать

```
1 # посмотрим как отличаются ЗП мужчин и женщин по годам
2 import seaborn as sns
3 import matplotlib.pyplot as plt
4
5
6 plt.figure(figsize=(19, 6))
7 sns.barplot(x='Age', y='ЗП (руб)', hue='Sex', data=hh, estimator=np.median, errorbar=None)
8 plt.title('Медианная ЗП по возрастным группам и полу')
9 plt.show()
```

```

9 plt.xlabel( 'возраст' )
10 plt.ylabel('Медианная ЗП (руб)')
11 plt.show()

```



ваши выводы здесь

- ЗП (руб) мужчин чаще всего больше чем у женщин

✓ Очистка данных

- Начнем с дубликатов в наших данных. Найдите **полные дубликаты** в таблице с резюме и удалите их.

```

1 # но сначала наверное нужно Задание 5.1 выполнить... Сразу напишу, у меня результат 158, но "правильный" ответ 155 либо 161. Это напрягает.
2 # Вывод информации о количестве дубликатов
3 print('Количество полных дубликатов в таблице:', hh.duplicated().sum())
4 # потом удаляем
5 hh = hh.drop_duplicates()
6 print('Количество полных дубликатов в таблице после удаления:', hh.duplicated().sum())

```



Количество полных дубликатов в таблице: 158
Количество полных дубликатов в таблице после удаления: 0

2. Займемся пропусками. Выведите информацию **о числе пропусков** в столбцах.

```
1 # Задание 5.2 Выведите информацию о числе пропусков в столбцах.
2 # ваш код здесь
3 print('Задание 5.2 Сколько пропусков в столбце «Опыт работы (месяц)»? ', hh['Опыт работы (месяц)'].isnull().sum())
```

➡ Задание 5.2 Сколько пропусков в столбце «Опыт работы (месяц)»? 168

3. Итак, у нас есть пропуски в 3ех столбцах: **"Опыт работы (месяц)", "Последнее/нынешнее место работы", "Последняя/нынешняя должность"**. Поступим следующим образом: удалите строки, где есть пропуск в столбцах с местом работы и должностью. Пропуски в столбце с опытом работы заполните **медианным** значением.

```
1 # ваш код здесь
2 # Удаление строк с пропусками в столбцах "Последнее/нынешнее место работы" и "Последняя/нынешняя должность"
3 hh=hh.dropna(subset=['Последнее/нынешнее место работы', 'Последняя/нынешняя должность'])
```

```
1 # Заполнение пропусков в столбце "Опыт работы (месяц)" медианным значением
2 hh['Опыт работы (месяц)'] = hh['Опыт работы (месяц)'].fillna(hh['Опыт работы (месяц)'].median())
```

```
1 print('Задание 5.3 Чему равно результирующее среднее значение в столбце «Опыт работы (месяц)» после заполнения пропусков? Ответ округлите до целых:',
2 int(hh['Опыт работы (месяц)'].mean()))
```

➡ Задание 5.3 Чему равно результирующее среднее значение в столбце «Опыт работы (месяц)» после заполнения пропусков? Ответ округлите до целых: 114

4. Мы добрались до ликвидации выбросов. Сначала очистим данные вручную. Удалите резюме, в которых указана заработная плата либо выше 1 млн. рублей, либо ниже 1 тыс. рублей.

```
1 # ваш код здесь
2 # Удаление строк, в которых указана заработная плата выше 1000000 или ниже 1000
3 print('Задание 5.4 - Количество выбросов где заработная плата либо выше 1 миллиона рублей, либо ниже 1 тысячи рублей:', hh[(hh['ЗП (руб)'] < 1000) | (hh['ЗП (руб)'] > 1000000)].shape[0] )
4 # Удаляю выбросы
5 hh = hh[(hh['ЗП (руб)'] > 1000) & (hh['ЗП (руб)'] < 1000000)]
```

➡ Задание 5.4 - Количество выбросов где заработная плата либо выше 1 миллиона рублей, либо ниже 1 тысячи рублей: 89

5. В процессе разведывательного анализа мы обнаружили резюме, в которых **опыт работы в годах превышал возраст соискателя**. Найдите такие резюме и удалите их из данных

```
1 print('Задание 5.5 - резюме, в которых опыт работы в годах превышал возраст соискателя:', len(hh[hh['Опыт работы (месяц)'] / 12 > hh['Age']].index))
2 hh = hh.drop(hh[hh['Опыт работы (месяц)'] / 12 > hh['Age']].index)
```

➡ Задание 5.5 - резюме, в которых опыт работы в годах превышал возраст соискателя: 6

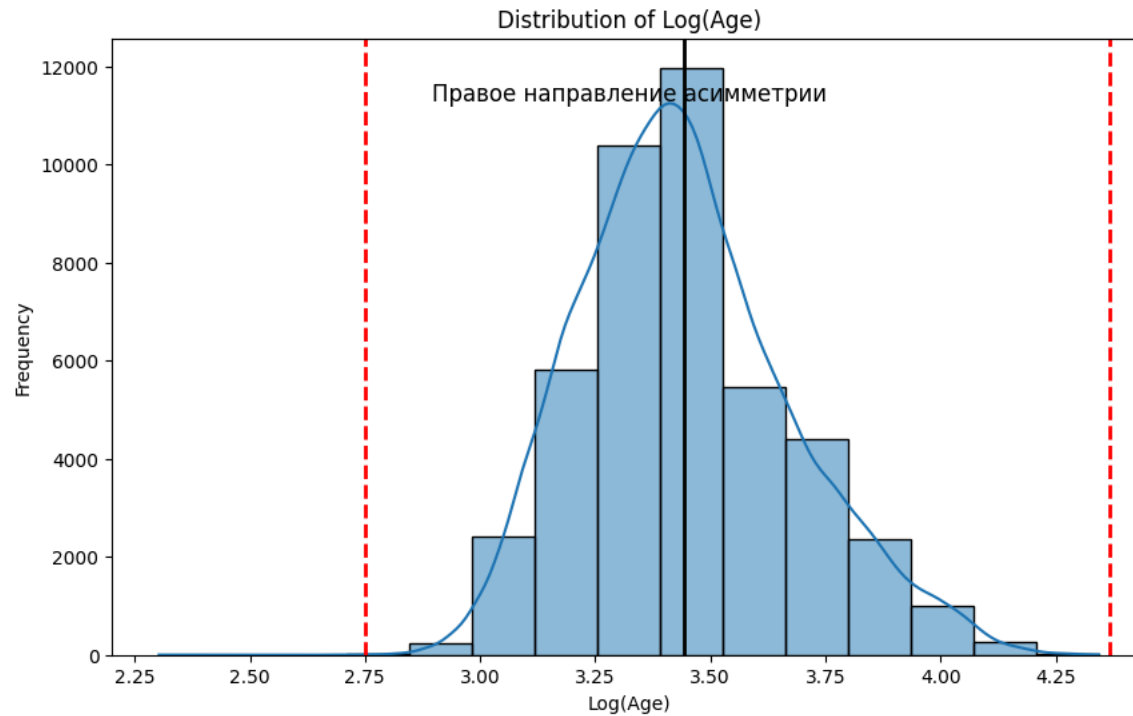
6. В результате анализа мы обнаружили потенциальные выбросы в признаке "**Возраст**". Это оказались резюме людей чересчур преклонного возраста для поиска работы. Попробуйте построить распределение признака в **логарифмическом масштабе**. Добавьте к графику линии, отображающие **среднее и границы интервала метода трех сигм**. Напомним, сделать это можно с помощью метода `axvline`. Например, для построение линии среднего будет иметь вид:

```
histplot.axvline(log_age.mean(), color='k', lw=2)
```

В какую сторону асимметрично логарифмическое распределение? Напишите об этом в комментарии к графику. Найдите выбросы с помощью метода z-отклонения и удалите их из данных, используйте логарифмический масштаб. Давайте сделаем послабление на **1 сигму** (возьмите 4 сигмы) в **правую сторону**.

Выведите таблицу с полученными выбросами и оцените, с каким возрастом соискатели попадают под категорию выбросов?

```
1 import seaborn as sns
2 import numpy as np
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Логарифмирование возраста
7 hh['Log_Age'] = np.log(hh['Age'])
8
9 # Построение распределение признака в логарифмическом масштабе
10 plt.figure(figsize=(10, 6))
11 histplot = sns.histplot(hh['Log_Age'], bins=15, kde=True)
12 plt.xlabel('Log(Age)')
13 plt.ylabel('Frequency')
14 plt.title('Distribution of Log(Age)')
15
16 # Вычисление среднего и границ трех сигм (4 сигмы в правую сторону согласно условию задания)
17 mean_log = hh['Log_Age'].mean()
18 std_log = hh['Log_Age'].std()
19 upper_bound_log = mean_log + 4 * std_log
20 lower_bound_log = mean_log - 3 * std_log
21
22
23 # Добавление линий среднего и границ трех сигм к графику
24 histplot.axvline(mean_log, color='k', lw=2, label='Mean')
25 histplot.axvline(lower_bound_log, color='r', lw=2, linestyle='--', label='Lower Bound (3 Sigma)')
26 histplot.axvline(upper_bound_log, color='r', lw=2, linestyle='--', label='Upper Bound (3 Sigma)')
27
28 # Определение направление асимметрии
29 if hh['Log_Age'].skew() > 0:
30     plt.annotate('Правое направление асимметрии', xy=(0.5, 0.9), xycoords='axes fraction', fontsize=12, ha='center')
31 elif hh['Log_Age'].skew() < 0:
32     plt.annotate('Левое направление асимметрии', xy=(0.5, 0.9), xycoords='axes fraction', fontsize=12, ha='center')
33 else:
34     plt.annotate('Симметрия', xy=(0.5, 0.9), xycoords='axes fraction', fontsize=12, ha='center')
35 print
36 plt.show()
37
```



```
1 # Вычисляю Число выбросов по методу z- отклонения
2 outliers = hh['Log_Age'][(hh['Log_Age'] < lower_bound_log) | (hh['Log_Age'] > upper_bound_log)]
3 print(f'Число выбросов по методу z- отклонения (логарифмический): {outliers.shape[0]}')
4
5 # удалите их из данных, используйте логарифмический масштаб
6 # hh = hh[~((np.log(hh['Age']) < lower_bound_log) | (np.log(hh['Age']) > upper_bound_log))]
7
8 # Вывести Таблицу с полученными выбросами (Но мы их удалили согласно заданию выше) и!!!
9 # !!! Логарифм не может показывать достоверные результаты
10 # Для сравнения: логарифмический
11 display('Для сравнения: логарифмический', hh[(hh['Log_Age'] < lower_bound_log) | (hh['Log_Age'] > upper_bound_log)])
12 # и Не логарифмический - покажет более достоверные данные
13 display('и Не логарифмический - покажет более достоверные данные', hh[(hh['Age'] < hh.Age.mean() - 3 * hh.Age.std()) | (hh['Age'] > hh.Age.mean() + 4 * hh.Age.std())])
14
15 # Оцениваю какой возраст попадает в категорию выбросов с учетом послабления
16 print('Возраст попадающий в категорию z-отклонения', int(hh.Age.mean() - 3 * hh.Age.std()), 'и', int(hh.Age.mean() + 4 * hh.Age.std()))
```



Число выбросов по методу z- отклонения (логарифмический): 3

'Для сравнения: логарифмический'

	Ищет работу на должность:	Город, переезд, командировки	Последнее/нынешнее место работы	Последняя/нынешняя должность	Обновление резюме	Авто	Образование	Sex	Age	Опыт работы (месяц)	...	ст
31137	Менеджер по работе с клиентами	Санкт-Петербург , не готов к переезду , не гот...	ООО "ФёрстКэшКомпани"	Менеджер по работе с клиентами	06.04.2019 09:13	Не указано	Среднее образование	M	15	2	...	
32950	Тестировщик игр	Канск , не готов к переезду , не готов к коман...	ООО ЖМЫХ	Тестировщик ПО	09.04.2019 16:02	Не указано	Среднее специальное	M	15	3	...	
33654	Frontend-разработчик	Санкт-Петербург , не готов к переезду , готов ...	Freelance	Frontend-разработчик	19.04.2019 23:27	Не указано	Среднее специальное	M	10	30	...	

3 rows × 26 columns

'и Не логарифмический - покажет более достоверные данные '

	Ищет работу на должность:	Город, переезд, командировки	Последнее/нынешнее место работы	Последняя/нынешняя должность	Обновление резюме	Авто	Образование	Sex	Age	p (м
152	Менеджер по работе с операторами связи	Талдом , не готова к переезду , не готова к ко...	УралКалий, ОАО	Начальник отдела движения персонала	07.04.2019 09:59	Не указано	Высшее	Ж	72	
850	постановщик задач, программист, руководитель I...	Москва , м. Кантемировская , не готов к перее...	ОАО "ЭКОС"	начальник лаборатории	14.05.2019 17:08	Не указано	Высшее	M	71	
1430	Руководитель проекта	Санкт-Петербург , м. Купчино , готов к переез...	ООО Сталт ЛТД	Руководитель проектов	21.04.2019 16:51	Не указано	Высшее	M	64	
2106	Разработчик. мультимедийных проектов. Организа...	Москва , м. Выхино , не готов к переезду , го...	ООО "Агентство"	Главный редактор	13.05.2019 10:30	Не указано	Высшее	M	68	
3271	Менеджер по продажам / Оператор на телефоне	Москва , м. Домодедовская , не готов к переез...	ООО "Гранд Евромобель" (удаленно)	Менеджер по работе с клиентами	11.05.2019 12:25	Имеется собственный автомобиль	Высшее	M	64	
...
40994	Программист, разработчик баз данных	Москва , м. Щукинская , не готов к переезду , ...	ООО "Геотрансинжиниринг"	Вед. специалист Технического отдела	07.09.2018 15:22	Не указано	Высшее	M	64	
		Москва м								

43381	Программист Navision	Славянский бульвар, не готов к п...	ООО ПРОФФИ-Стиль	Программист- разработчик	26.04.2019 11:54	Не указано	Высшее	М	68
44363	Эксперт (специалист) по информационной безопас...	Москва, м. Планерная, не готов к переезду, ...	ООО "КАБЕСТ", Группа компаний "Астерос"	Директор департамента, Руководитель службы Гла...	26.04.2019 15:40	Не указано	Высшее	М	70
44414	Инженер- электронщик Работы на дому у себя.	Челябинск, не готов к переезду, не готов к к...	Ч.П.	Инженер- конструктор- электронщик	08.04.2019 12:29	Не указано	Высшее	М	65

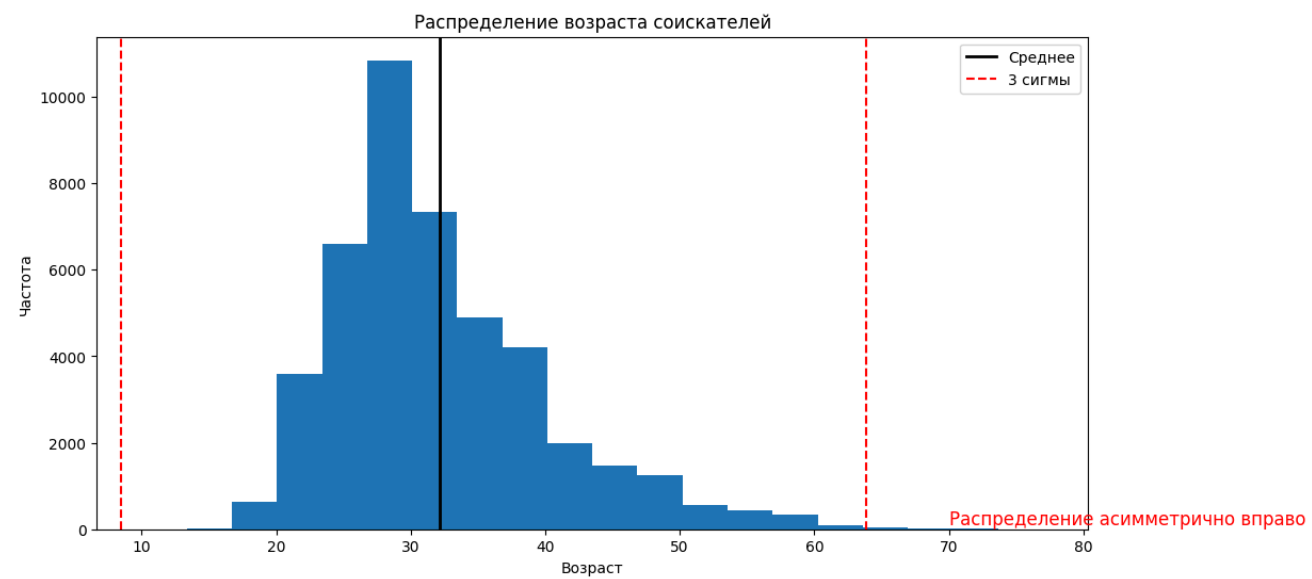
1

```

1 # P.S. Нагляднее изобразить график с реальным возрастом
2 # Построение распределения признака "Возраст" в реальных значениях
3 plt.figure(figsize=(12, 6))
4 plt.hist(hh['Age'], bins=20)
5 plt.xlabel('Возраст')
6 plt.ylabel('Частота')
7 plt.title('Распределение возраста соискателей')
8
9 # Добавление линий среднего и границ интервала метода трех сигм
10 mean_age = hh['Age'].mean()
11 std_age = hh['Age'].std()
12 plt.axvline(mean_age, color='k', lw=2, label='Среднее')
13 # Послабление до 4 сигм
14 plt.axvline(mean_age + 4*std_age, color='r', linestyle='--', label='3 сигмы')
15 plt.axvline(mean_age - 3*std_age, color='r', linestyle='--')
16 plt.legend()
17 print(mean_age + 4*std_age)
18
19 # Комментарий к графику: распределение асимметрично вправо
20 plt.text(70, 100, 'Распределение асимметрично вправо', fontsize=12, color='red')
21
22 # Нахождение и удаление выбросов с помощью z-отклонения (усиленного до 4 сигм)
23 outliers = hh[(hh['Age'] - mean_age).abs() > 4*std_age]['Age']
24 cleaned_data = hh[~hh['Age'].isin(outliers)]
25
26 plt.show()
27
28 # Вывод таблицы с полученными выбросами
29 print("Выбросы по возрасту:", list(set(outliers))[0], '-', list(set(outliers))[-1])

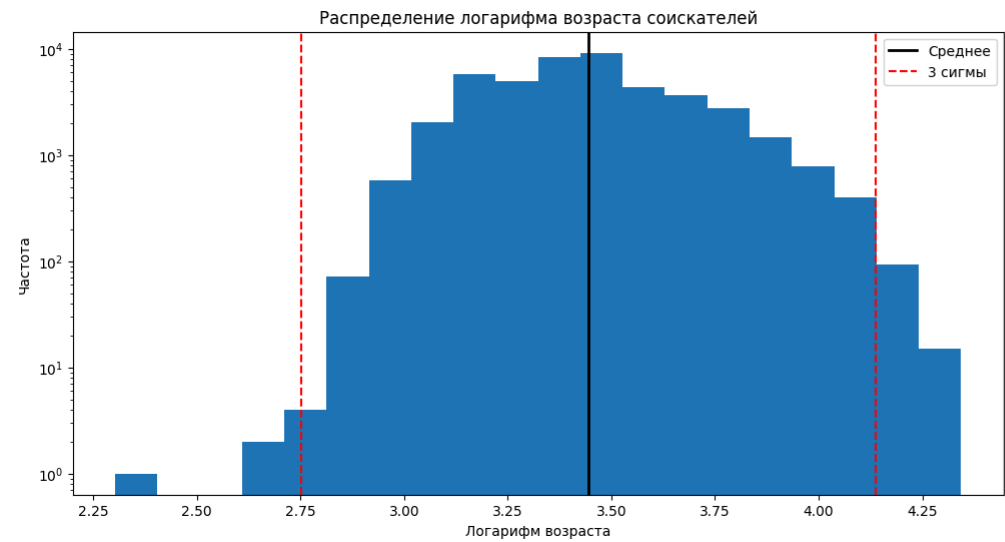
```


↕ 63.84362718131571



Выбросы по возрасту: 64 - 77

```
1 # И решение согласно заданию
2
3 import pandas as pd
4 import numpy as np
5 import matplotlib.pyplot as plt
6
7 # Построение распределения признака "Возраст" в логарифмическом масштабе
8 plt.figure(figsize=(12, 6))
9 log_age = np.log(hh['Age'])
10 plt.hist(log_age, bins=20)
11 plt.yscale('log')
12 plt.xlabel('Логарифм возраста')
13 plt.ylabel('Частота')
14 plt.title('Распределение логарифма возраста соискателей')
15
16 # Добавление линий среднего и границ интервала метода трех сигм
17 mean_age = log_age.mean()
18 std_age = log_age.std()
19 plt.axvline(mean_age, color='k', lw=2, label='Среднее')
20 plt.axvline(mean_age + 3*std_age, color='r', linestyle='--', label='3 сигмы')
21 plt.axvline(mean_age - 3*std_age, color='r', linestyle='--')
22 plt.legend()
23
24 # логарифмическое распределение асимметрично вправо
25 plt.text(4.5, 100, 'Распределение асимметрично вправо', fontsize=10, color='red')
26
27 # Нахождение и удаление выбросов с помощью z-отклонения (4 сигмы)
28 outliers = log_age[(log_age - mean_age).abs() > 4*std_age]
29 cleaned_data = hh.drop(outliers.index)
30
31 plt.show()
32
33 # Вывод таблицы с полученными выбросами
34 print("Выбросы по возрасту:")
35 display(hh.loc[outliers.index, 'Age'])
```



Распределение асимметрично вправо

Выбросы по возрасту:

Age	
33654	10

dtype: Int16

ваш коммментарий здесь

- Использование логарифмического метода в данном задании отдаляет от получения достоверного результата.
- Рекомендую использовать реальные данные для получения наглядного и достоверного результата

1

1

1

1

1 # Хвост, можно не читать)

1

```
1 # Преобразование даты и времени в столбце 'Обновление резюме' заранее
2 hh['Обновление резюме'] = pd.to_datetime(hh['Обновление резюме'], format='%d.%m.%Y %H:%M').dt.strftime('%d/%m/%Y')
3
4 def convert_salary(row):
5     salary, currency = row['ЭП'].split()
6     salary = int(salary)
7     currency = currency.rstrip('.').upper()
8
9     if currency == 'РУБ':
10         return pd.Series([salary, None, None, currency])
11
12     if currency == 'БЕЛ.РУБ':
13         currency = 'BYN'
14
15     date = row['Обновление резюме']
16     exchange_rate_row = hhe.loc[(hhe['date'] == date) & (hhe['currency'].str.upper() == currency)]
17
18     if not exchange_rate_row.empty:
19         return pd.Series([salary * exchange_rate_row['close'].iloc[0] / exchange_rate_row['proportion'].iloc[0], exchange_rate_row['close'].iloc[0], exchange_rate_row['proportion'].iloc[0], currency])
20     else:
21         return pd.Series([None, currency, None, None])
22
23 hh[['ЭП py6.', 'currency', 'close', 'proportion']] = hh.apply(convert_salary, axis=1)
24 hh[~hh['ЭП'].str.contains(' py6')]
25
26
27 1 hhdate = pd.to_datetime(hh['Обновление резюме'], format='%d.%m.%Y %H:%M').dt.strftime('%d/%m/%Y')
28 2 hhdate = pd.to_datetime(hhe['date'], format='%d/%m/%Y').dt.strftime('%d/%m/%Y')
29
30
31 1 print(set([i.split()[1].upper() for i in hh['ЭП'] if i != 'py6.']))
32
33 ➡ {'БЕЛ.РУБ.', 'KGS', 'EUR', 'KZT', 'AZN', 'ГРН.', 'CUM', 'USD'}
```

```

1 # Convert 'Обновление резюме' to datetime and remove the time for correct matching
2 hh['Обновление резюме'] = pd.to_datetime(hh['Обновление резюме'], format='%d.%m.%Y %H:%M').dt.date
3
4 # Split '3П' into salary and currency
5 hh[['salary', 'currency']] = hh['3П'].str.split().apply(pd.Series)
6 hh['salary'] = hh['salary'].astype(int)
7 hh['currency'] = hh['currency'].str.rstrip('.').str.upper()
8
9 # Replace 'БЕЛ.РУБ' with 'BYN'
10 hh['currency'] = hh['currency'].replace('БЕЛ.РУБ', 'BYN')
11
12 # Define conversion function
13 def convert_salary(row):
14     hhdate = row['Обновление резюме']
15     salary = row['salary']
16     currency = row['currency']
17
18     if currency == 'РУБ': return pd.Series([salary, currency, None])
19     else:
20         exchange_rate_row = hhe.loc[(hhe['date'] == hhdate) & (hhe['currency'].str.upper() == currency)]
21         if not exchange_rate_row.empty:
22             return pd.Series([salary * exchange_rate_row['close'].iloc[0] / exchange_rate_row['proportion'].iloc[0], currency, exchange_rate_row['proportion'].iloc[0]])
23         else:
24             return pd.Series([None, currency, None])
25
26 # Apply conversion function
27 hh[['3П py6.', 'currency', 'proportion']] = hh.apply(convert_salary, axis=1)
28
29 # Filter out rows where currency is 'РУБ'
30 hh = hh[hh['currency'] != 'РУБ']
31
32 hh

```

```
1 def convert_salary(row):
2 #####
3 hh['Обновление резюме'] = pd.to_datetime(hh['Обновление резюме'], format='%d.%m.%Y %H:%M').dt.date
4 def convert_salary(row):
5     hhdate = row['Обновление резюме']
6     salary, currency = row['ЗП'].split()
```