"Принятие решений "

Цель: провести анализ гипотез и результатов A/B теста с целью увеличения выручки.

Заказчик: интернет-магазин.

Предоставленные данные для анализа:

Данные с гипотезами:

краткое описание гипотезы; значения охвата по 10-бальной шкале; значения влияния на пользователей по 10-балльной шкале; значения уверенности в гипотезе по 10-балльной шкале; значения затрат ресурсов на проверку гипотезы по 10-балльной шкале.

Данные с заказами:

Уникальный номер заказа, уникальный номер покупателя, дату совершения заказа, дату совершения заказа, выручку, тестируемую группу.

Данные с посещениями:

Дата посещения, количество пользователей, группа теста

Ввод [1]:

```
import pandas as pd
import datetime as dt
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
```

Приоритизация гипотез:

Загрузка данных:

Ввод [2]:

```
data_hip = pd.read_csv('D:/АНАЛИТИКА/_6 Принятие решений в бизнесе/hypothesis.csv')
pd.set_option('max_colwidth', 120)
pd.set_option('display.width', 500)
display(data_hip)
data_hip.info()
```

	Hypothesis	Reach	Impact	Confidence	Efforts
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Hypothesis  9 non-null      object
 1   Reach       9 non-null      int64 
 2   Impact      9 non-null      int64 
 3   Confidence  9 non-null      int64 
 4   Efforts     9 non-null      int64 
dtypes: int64(4), object(1)
memory usage: 488.0+ bytes
```

В таблице с даннымми о гипотезах, данные записаны в корректном типе, пропусков нет.

Ввод [3]:

```
data_hip.set_axis(['hipothesis','reach','impact','confidence', 'efforts'], axis='columns',
print(data_hip['hipothesis'].unique())
```

```
['Добавить два новых канала привлечения трафика, что позволит привлекать на
30% больше пользователей'
 'Запустить собственную службу доставки, что сократит срок доставки заказов'
 'Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повыс
ить конверсию и средний чек заказа'
 'Изменить структура категорий, что увеличит конверсию, т.к. пользователи бы
стрее найдут нужный товар'
 'Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользов
ателей'
 'Добавить страницу отзывов клиентов о магазине, что позволит увеличить коли
чество заказов'
 'Показать на главной странице баннеры с актуальными акциями и распродажами,
чтобы увеличить конверсию'
 'Добавить форму подписки на все основные страницы, чтобы собрать базу клиен
тов для email-рассылок'
 'Запустить акцию, дающую скидку на товар в день рождения']
```

Приоритизация гипотез способом ICE:

Ввод [4]:

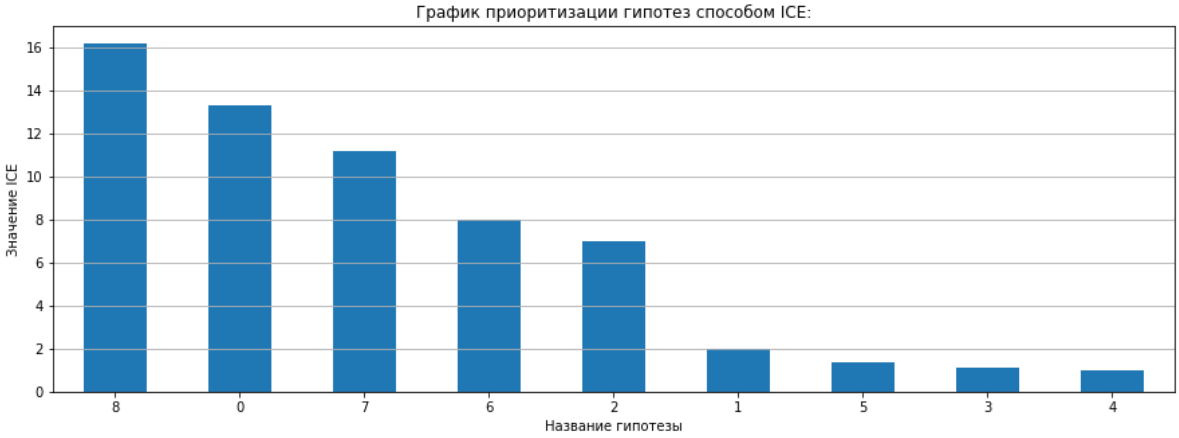
```
data_hip['ice']=(data_hip['impact']*data_hip['confidence'])/data_hip['efforts']
display(data_hip.sort_values(by='ice', ascending=False))
```

	hipothesis	reach	impact	confidence	efforts	ice
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200000
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333333
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200000
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000000
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000000
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000000
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333333
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125000
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000000

Ввод [5]:

```
data_hip[['hipothesis','ice']].sort_values(by='ice',ascending = False).plot( kind='bar', fi
plt.xticks(rotation=0)
plt.title('График приоритизации гипотез способом ICE:')
plt.xlabel('Название гипотезы')
plt.ylabel('Значение ICE')
plt.grid(axis='y')


plt.show()
```



Приоритизация гипотез способом RICE:

Ввод [6]:

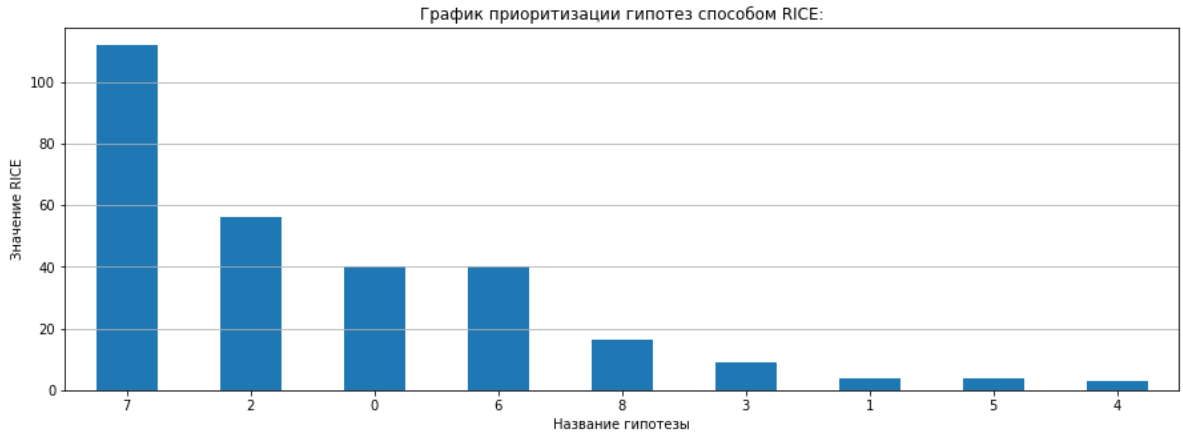
```
data_hip['rice']=(data_hip['reach']*data_hip['impact']*data_hip['confidence'])/data_hip['ef
display(data_hip.sort_values(by='rice', ascending=False))
```

	hipothesis	reach	impact	confidence	efforts	ice	rice
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200000	112.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000000	56.0
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333333	40.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000000	40.0
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200000	16.2
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125000	9.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000000	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333333	4.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000000	3.0

Ввод [7]:

```
data_hip[['hipothesis','rice']].sort_values(by='rice',ascending = False).plot( kind='bar',
plt.xticks(rotation=0)
plt.title('График приоритизации гипотез способом RICE:')
plt.xlabel('Название гипотезы')
plt.ylabel('Значение RICE')
plt.grid(axis='y')


plt.show()
```



Выводы по теме приоритизации гипотез:

Перспективные гипотезы:

К перспективным гипотезам по результатам приоритизации относятся гипотезы с индексами: 0, 2, 7, 6, 8

В способах ICE и RICE они занимают разные места в рейтингах, так происходит из-за значений охвата аудитории:

- **"Запустить акцию, дающую скидку на товар в день рождения"** - данная гипотеза занимает первую строчку метода ICE и последнее место в RICE, так как охват равен всего лишь единице, из-за качественного признака целевой аудитории. Но данные изменения сильно повлияют на выбранную аудиторию, значение - 9. Так как есть возможность посчитать пользователей, подверженных изменению, и прошлый опыт наверняка показывает, что скидки приуроченные к дню рождения повышают спрос, то и уверенность стремится к максимальной оценке и имеет значение - 9. Проверка этой гипотезы имеет среднее значение сложности, так как возникнет дополнительная нагрузка на отдел маркетинга и программистов.
- **"Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок"** - эта гипотеза занимаетпервую строчку в способе RICE и третью в ICE, также это произошло благодаря максимальному значению охвата, действительно всем посетителям сайта будет предложена форма регистрации, но не на всех это действие окажет сильное влияние, 30% проигнорируют заполнение формы регистрации, по сложности проверки, гипотеза занимает среднее значение.
- **"Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей"** - гипотеза находится на втором месте по ICE и на третьем месте по RICE снижение на строку произошло из-за вмешательства значения охвата - 3, но окажет максимальное 100% влияние на объём охваченных пользователей.

- **"Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа"** - гипотеза, изменение метрик в которой охватят многих посетителей сайта, с небольшим увеличением затратб данная гипотеза из-за отсутствия сильного воздействия на аудиторию располагается на поседнем месте среди перспективных гипотез по методу ICE и на втром по RICE, ситуацию выправляет боьшой охват.
- **"Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию"** - данная гипотеза по обоим методам приоритизации находится на четвёртой строке.

Гипотезы-аутсайдеры:

К гипотезам аутсайдерам относятся гипотезы с индексами: 1, 3, 5, 4

Они занимают проигрышные позиции либо из-за дороговизны и сложности проверки, либо из-за небольшого охвата, к которому прилагается низкий уровень влияния при изменении метрик

Анализ A/B теста:

Получение и обработка данных заказов и посещений:

Ввод [8]:

```
orders = pd.read_csv('D:/АНАЛИТИКА/_6 Принятие решений в бизнесе/orders.csv')
display(orders.head())
orders.info()
```

	transactionId	visitorId	date	revenue	group
0	3667963787	3312258926	2019-08-15	1650	B
1	2804400009	3642806036	2019-08-15	730	B
2	2961555356	4069496402	2019-08-15	400	A
3	3797467345	1196621759	2019-08-15	9759	B
4	2282983706	2322279887	2019-08-15	2308	B

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1197 entries, 0 to 1196
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   transactionId  1197 non-null   int64 
 1   visitorId      1197 non-null   int64 
 2   date           1197 non-null   object
 3   revenue        1197 non-null   int64 
 4   group          1197 non-null   object
dtypes: int64(3), object(2)
memory usage: 46.9+ KB
```

Ввод [9]:

```
visitors=pd.read_csv('D:/АНАЛИТИКА/_6 Принятие решений в бизнесе/visitors.csv')
display(visitors.head())
visitors.info()
```

	date	group	visitors
0	2019-08-01	A	719
1	2019-08-02	A	619
2	2019-08-03	A	507
3	2019-08-04	A	717
4	2019-08-05	A	756

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   date      62 non-null     object
 1   group     62 non-null     object
 2   visitors  62 non-null     int64 
dtypes: int64(1), object(2)
memory usage: 1.6+ KB
```

Приведение столбцов с датами к корректному типу в обоих датафреймах:

Ввод [10]:

```
orders['date'] = orders['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
visitors['date'] = visitors['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
display(orders.head())
display(visitors.head())
orders.info()
visitors.info()
```

	transactionId	visitorId	date	revenue	group
0	3667963787	3312258926	2019-08-15	1650	B
1	2804400009	3642806036	2019-08-15	730	B
2	2961555356	4069496402	2019-08-15	400	A
3	3797467345	1196621759	2019-08-15	9759	B
4	2282983706	2322279887	2019-08-15	2308	B

	date	group	visitors
0	2019-08-01	A	719
1	2019-08-02	A	619
2	2019-08-03	A	507
3	2019-08-04	A	717
4	2019-08-05	A	756

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1197 entries, 0 to 1196
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype         
---  ------         --------------  -----         
 0   transactionId  1197 non-null   int64         
 1   visitorId      1197 non-null   int64         
 2   date           1197 non-null   datetime64[ns]
 3   revenue        1197 non-null   int64         
 4   group          1197 non-null   object        
dtypes: datetime64[ns](1), int64(3), object(1)
memory usage: 46.9+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype         
---  ------    --------------  -----         
 0   date      62 non-null     datetime64[ns]
 1   group     62 non-null     object        
 2   visitors  62 non-null     int64         
dtypes: datetime64[ns](1), int64(1), object(1)
memory usage: 1.6+ KB
```

В таблицах с данными о заказах и посещениях данные в столбках с датами были приведены к корректному типу.

График и расчёт кумулятивной выручки по группам:

Ввод [11]:

```
orders[['date','group']].duplicated().sum()
```

Out[11]:

```
1135
```

Создание датафрейма с уникальными парами значений даты и групп, объявление переменной с датой, группой, числом уникальных значений заказов, пользователейи выручкой:

Ввод [12]:

```
datesGroups = orders[['date','group']].drop_duplicates()
ordersAggregated = datesGroups.apply(
    lambda x: orders[np.logical_and(orders['date'] <= x['date'], orders['group'] == x['grou
    .agg({'date' : 'max','group' : 'max','transactionId' : pd.Series.nunique,'visitorId' :
        ), axis=1).sort_values(by=['date','group'])
```

Объявление переменной с датой, группой и числом уникальных посетителей:

Ввод [13]:

```
visitorsAggregated = datesGroups.apply(
    lambda x: visitors[np.logical_and(visitors['date'] <= x['date'], visitors['group'] == x
    .agg({'date' : 'max', 'group' : 'max', 'visitors' : 'sum'}), axis=1).sort_values(by=['d
```

Создание датафрейма с накапливаемыми данными:

Ввод [14]:

```
cumulativeData = ordersAggregated.merge(visitorsAggregated, left_on=['date', 'group'], righ
cumulativeData.columns = ['date', 'group', 'orders', 'buyers', 'revenue', 'visitors']
display(cumulativeData.head())
```

	date	group	orders	buyers	revenue	visitors
0	2019-08-01	A	24	20	148579	719
1	2019-08-01	B	21	20	101217	713
2	2019-08-02	A	44	38	242401	1338
3	2019-08-02	B	45	43	266748	1294
4	2019-08-03	A	68	62	354874	1845

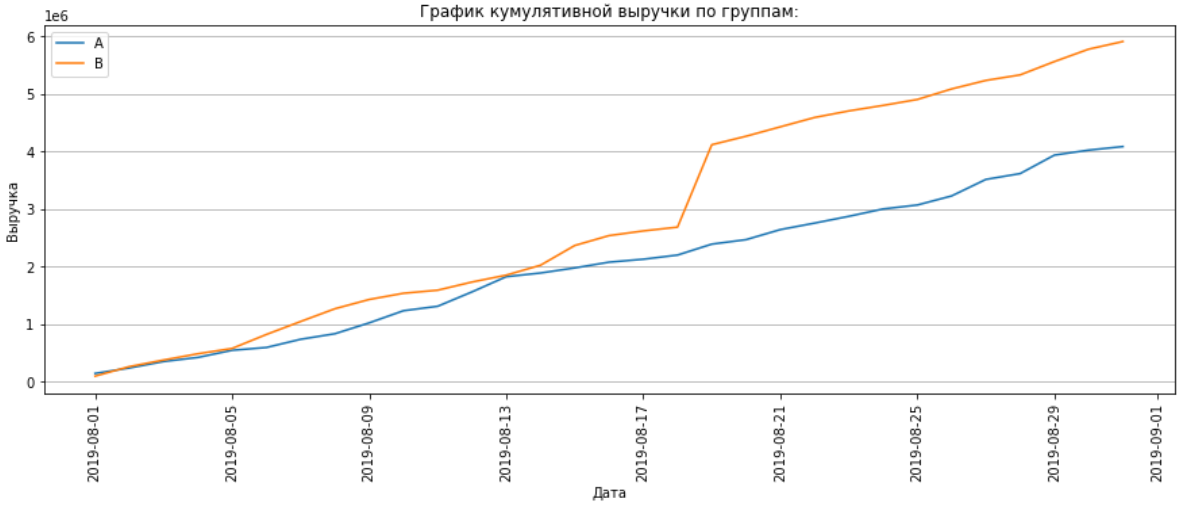
Построение графика кумулятивной выручки по группам:

Ввод [15]:

```
from pandas.plotting import register_matplotlib_converters
```

Ввод [16]:

```
cumulativeRevenueA = cumulativeData[cumulativeData['group']=='A'][['date','revenue', 'order
cumulativeRevenueB = cumulativeData[cumulativeData['group']=='B'][['date','revenue', 'order
plt.figure(figsize=(15,5))
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue'], label='A')
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue'], label='B')
plt.legend()
plt.xticks(rotation=90)
plt.title('График кумулятивной выручки по группам:')
plt.xlabel('Дата')
plt.ylabel('Выручка')
plt.grid(axis='y')
plt.show()
```

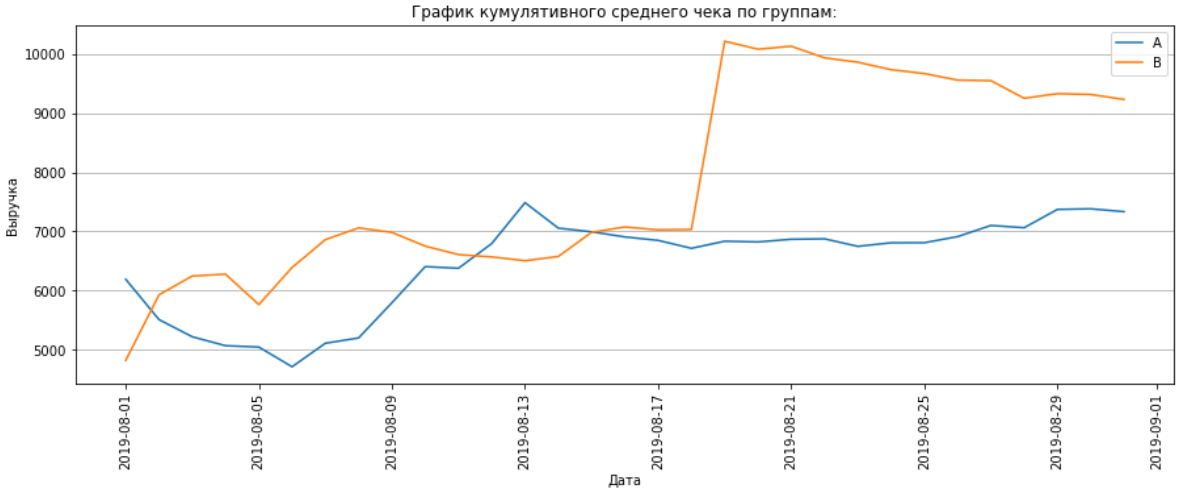


Графики кумулятивной выручки для обеих групп растут. Произошёл небольшой всплеск в выручке группы А 13 августа где А и В сравнялись в выручке. И также сильный всплеск в группе В 18-го числа, который вывел группу В в лидеры и оставил её на лидирующей позиции, возможно влияние аномальных заказов.

График кумулятивного среднего чека по группам:

Ввод [17]:

```
plt.figure(figsize=(15,5))
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue']/cumulativeRevenueA['orde
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue']/cumulativeRevenueB['orde
plt.legend()
plt.xticks(rotation=90)
plt.title('График кумулятивного среднего чека по группам:')
plt.xlabel('Дата')
plt.ylabel('Выручка')
plt.grid(axis='y')
plt.show()
```



Кумулятивный средний чек группы А снижается в первых числах, но после снижения происходит рост среднего чека, и к 13-му числу средний ченк группы А превосходит сегмент В, увеличение среднего чека сказалось на равенстве куммулятивной выручки обеих групп 13-го августа. После пика, средний кумулятвный чек снижается и остаётся стабильным. В группе В резкий скачек с 7000 до 10000 среднего кумулятивного чека произошёл примерно 18-го числа посленебольшого снижения значения стабилизировались.

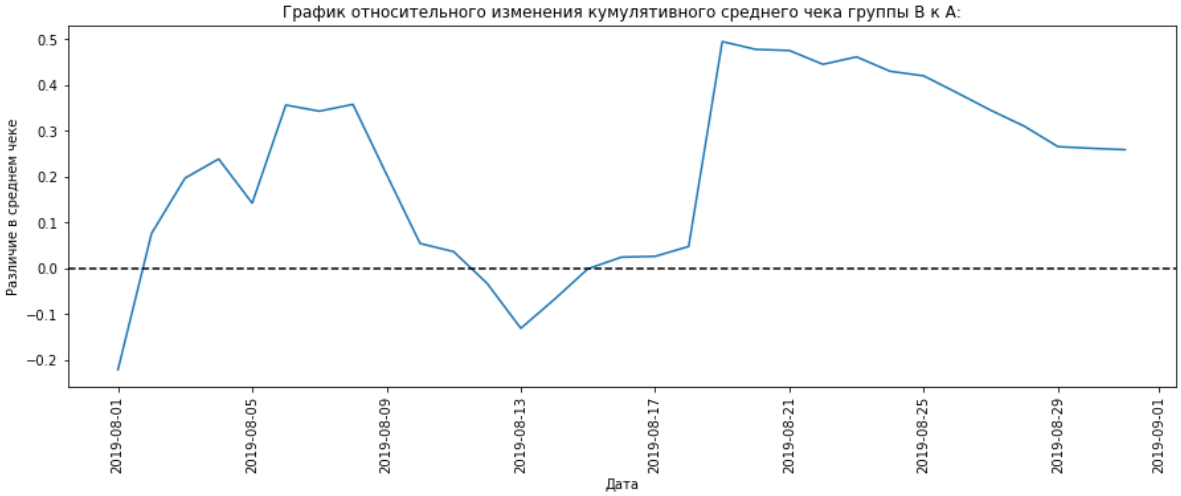
График относительного изменения кумулятивного среднего чека группы В к А:

Ввод [18]:

```
mergedCumulativeRevenue = cumulativeRevenueA.merge(
    cumulativeRevenueB, left_on='date', right_on='date', how='left', suffixes=['A', 'B'])
```

Ввод [19]:

```
plt.figure(figsize=(15,5))
plt.plot(mergedCumulativeRevenue['date'], (
    mergedCumulativeRevenue['revenueB']/mergedCumulativeRevenue['ordersB']
)/(mergedCumulativeRevenue['revenueA']/mergedCumulativeRevenue['ordersA'])-1)
plt.axhline(y=0, color='black', linestyle='--')
plt.xticks(rotation=90)
plt.title('График относительного изменения кумулятивного среднего чека группы B к A:')
plt.xlabel('Дата')
plt.ylabel('Различие в среднем чеке')
plt.show()
```



На графике относительного изменения среднего кумулятивного чека группы B к A заметны изменения в начале теста и 13-го числа, когда были сделаны анамольные заказы.

График кумулятивной конверсии по группам:

Расчёт кумулятивной конверсии и добавдение столбца с данными в таблицу:

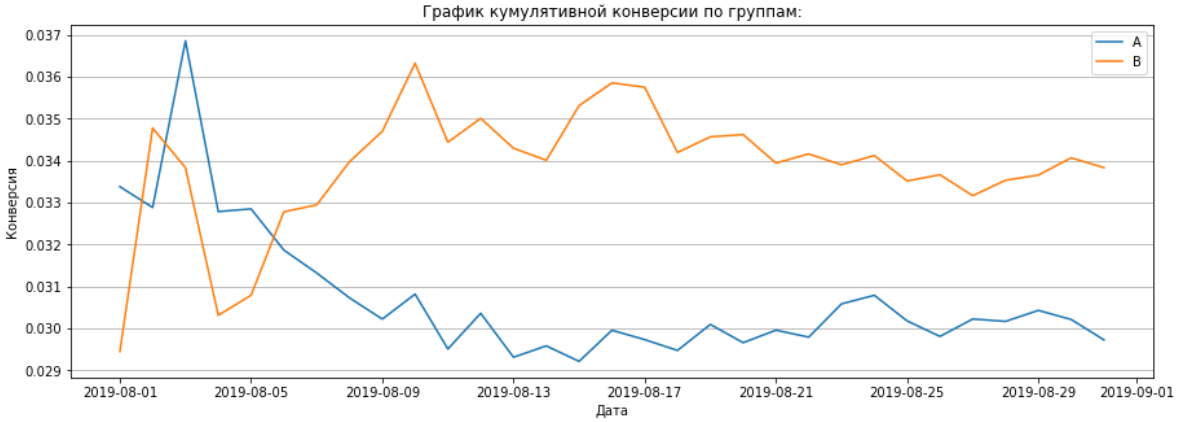
Ввод [20]:

```
cumulativeData['conversion'] = cumulativeData['orders']/cumulativeData['visitors']
cumulativeDataA = cumulativeData[cumulativeData['group']=='A']
cumulativeDataB = cumulativeData[cumulativeData['group']=='B']
```

Построение графиков кумулятивной конверсии по группам:

Ввод [21]:

```
plt.figure(figsize=(15,5))
plt.plot(cumulativeDataA['date'], cumulativeDataA['conversion'], label='A')
plt.plot(cumulativeDataB['date'], cumulativeDataB['conversion'], label='B')
plt.legend()
plt.title('График кумулятивной конверсии по группам:')
plt.xlabel('Дата')
plt.ylabel('Конверсия')
plt.grid(axis='y')
plt.show()
```



В первые шесть дней теста конверсия в целом была выше у сегмента A, затем у группы B произошёл рост конверсии и начиная со второй недели теста, конверсии выровнялись у обеих групп, но в группе B конверсия выше на протяжении остального времени теста.

График относительного изменения кумулятивной конверсии группы B к A:

Создание таблицы для построения графика изменения кумулятивной конверсии:

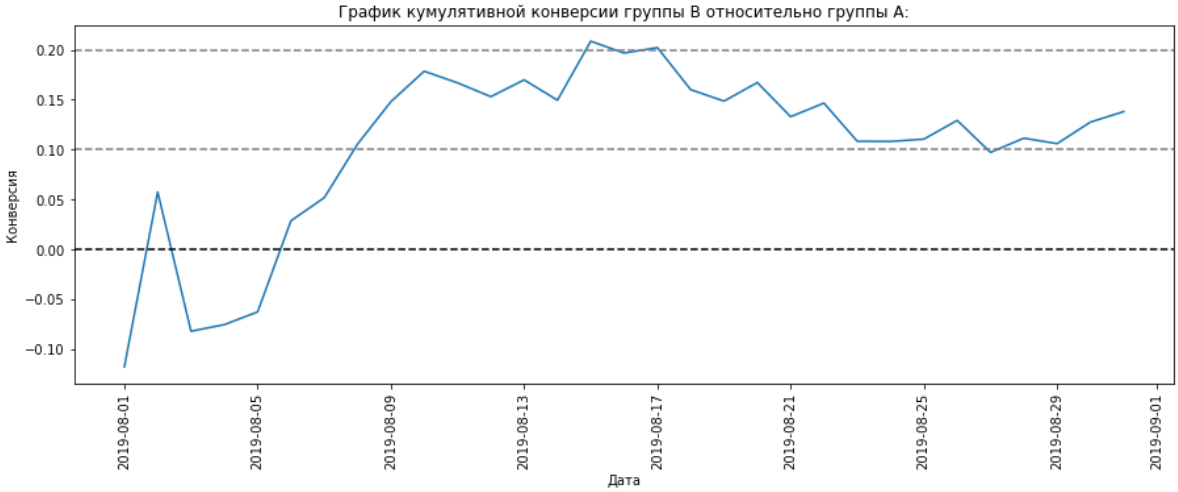
Ввод [22]:

```
mergedCumConvers = cumulativeDataA[['date', 'conversion']].merge(
    cumulativeDataB[['date', 'conversion']],left_on='date', right_on='date', how='left',suf
```

Построение графика изменения конверсии групы B к A:

Ввод [23]:

```
plt.figure(figsize=(15,5))
plt.plot(mergedCumConvers['date'],mergedCumConvers['conversionB']/mergedCumConvers['convers
plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=0.1, color='gray', linestyle='--')
plt.axhline(y=0.2, color='gray', linestyle='--')
plt.xticks(rotation=90)
plt.title('График кумулятивной конверсии группы B относительно группы A:')
plt.xlabel('Дата')
plt.ylabel('Конверсия')
plt.show()
```



После 6 дней теста конверсия группы B лидирует, стабильно растёт но после небольшого понижения почти зафиксировалась на отметке в 10% но последний день месяца опять намекает на небольшой рост.

График количества заказов по пользователям и поиск аномалий:

График количества заказов по пользователям:

Создание датафрейма с количеством заказов на покупателя:

Ввод [24]:

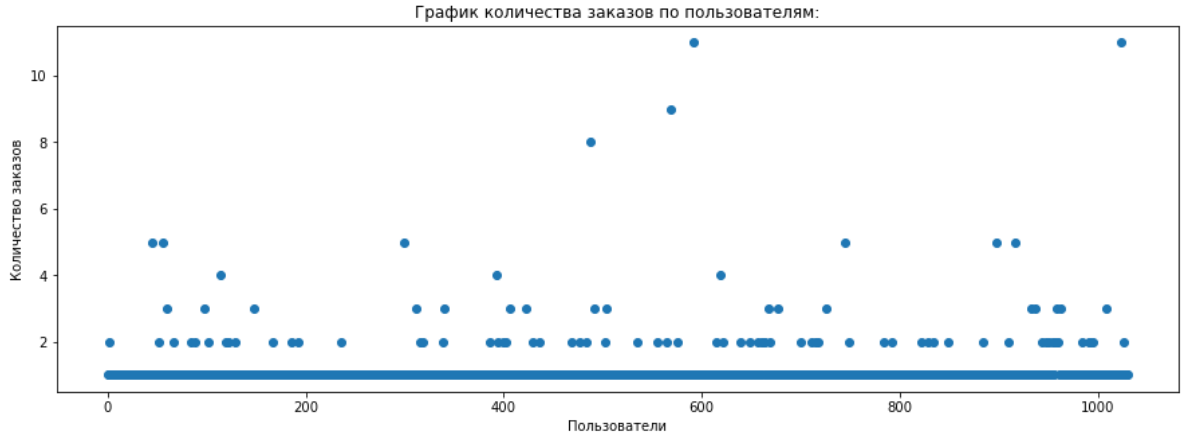
```
orders_by_users = orders.groupby('visitorId', as_index=False).agg({'transactionId':pd.Serie
orders_by_users.columns= ['userId','orders']
display(orders_by_users.sort_values(by='orders', ascending=False).head())
```

	userId	orders
1023	4256040402	11
591	2458001652	11
569	2378935119	9
487	2038680547	8
44	199603092	5

Построение графика количества заказов по пользователям:

Ввод [25]:

```
x_values=pd.Series(range(0,len(orders_by_users)))
plt.figure(figsize=(15,5))
plt.scatter(x_values,orders_by_users['orders'])
plt.title('График количества заказов по пользователям:')
plt.xlabel('Пользователи')
plt.ylabel('Количество заказов')
plt.show()
```



Мало кто из пользователей сделал более двух заказов, но и количество пользователей сделавших два заказа,также невелико. Заказы в количестве более двух возможно аномальные.

Вычисление аномалий в количестве заказов пользователями:

Ввод [26]:

```
orders_by_users_percentile_95 = np.percentile(orders_by_users['orders'], 95)
print('Количество заказов для 5% пользователей:')
print(orders_by_users_percentile_95)
print()
orders_by_users_percentile_99 = np.percentile(orders_by_users['orders'],99)
print('Количество заказов для 1% пользователей:')
print(orders_by_users_percentile_99)
```

```
Количество заказов для 5% пользователей:
2.0

Количество заказов для 1% пользователей:
4.0
```

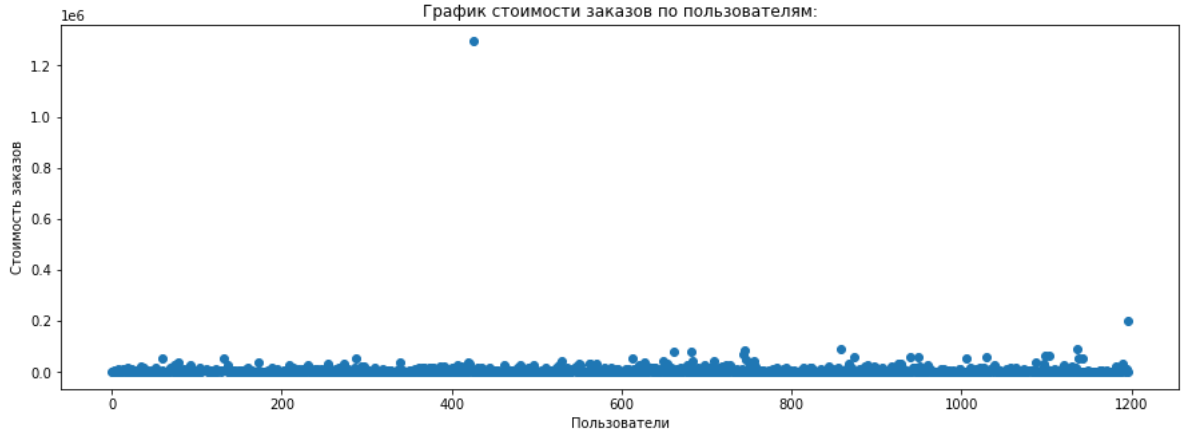
4 и более заказов было сделано 1% пользователей, похоже на аномалию.

График стоимости заказов и поиск анамалий:

График стоимости заказов:

Ввод [27]:

```
x_values_r = pd.Series(range(0, len(orders['revenue'])))
plt.figure(figsize=(15,5))
plt.scatter(x_values_r,orders['revenue'])
plt.title('График стоимости заказов по пользователям:')
plt.xlabel('Пользователи')
plt.ylabel('Стоимость заказов')
plt.show()
```



Сумма большинства заказов находится точно до 100 тысяч. Заметны выбросы после этих значений, но есть заказ на сумму 1,2млн.

Вычисление аномалий в стоимости заказов пользователями:

Ввод [28]:

```
orders_percentile_95 = np.percentile(orders['revenue'], 95)
print('Стоимость заказа для 5% пользователей:')
print(orders_percentile_95)
print()
orders_percentile_99 = np.percentile(orders['revenue'],99)
print('Стоимость заказа для 1% пользователей:')
print(orders_percentile_99)
```

```
Стоимость заказа для 5% пользователей:
28000.000000000004

Стоимость заказа для 1% пользователей:
58233.19999999999
```

Всего 1% заказов имеет сумму более 58тысяч, это явная граница для аномалий.

"Сырые" данные, статистическая значимость различий конверсий в группах:

Создание таблицы с накопительными данными по визитам для каждой группы.

Ввод [29]:

```
visitorsADaily = visitors[visitors['group'] == 'A'][['date', 'visitors']]
visitorsADaily.columns = ['date', 'visitorsPerDateA']
visitorsACummulative = visitorsADaily.apply(
    lambda x: visitorsADaily[visitorsADaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateA': 'sum'}),axis=1,)
visitorsACummulative.columns = ['date', 'visitorsCummulativeA']

visitorsBDaily = visitors[visitors['group'] == 'B'][['date', 'visitors']]
visitorsBDaily.columns = ['date', 'visitorsPerDateB']
visitorsBCummulative = visitorsBDaily.apply(
    lambda x: visitorsBDaily[visitorsBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateB': 'sum'}),axis=1,)
visitorsBCummulative.columns = ['date', 'visitorsCummulativeB']
display(visitorsACummulative.head())
display(visitorsBCummulative.head())
```

	date	visitorsCummulativeA
0	2019-08-01	719
1	2019-08-02	1338
2	2019-08-03	1845
3	2019-08-04	2562
4	2019-08-05	3318

	date	visitorsCummulativeB
31	2019-08-01	713
32	2019-08-02	1294
33	2019-08-03	1803
34	2019-08-04	2573
35	2019-08-05	3280

Создание таблицы с накопительными данными по заказам для каждой группы.

Ввод [30]:

```
ordersADaily = (
    orders[orders['group'] == 'A'][['date', 'transactionId', 'visitorId', 'revenue']]
    .groupby('date', as_index=False)
    .agg({'transactionId': pd.Series.nunique, 'revenue': 'sum'}))
ordersADaily.columns = ['date', 'ordersPerDateA', 'revenuePerDateA']

ordersACummulative = ordersADaily.apply(
    lambda x: ordersADaily[ordersADaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateA': 'sum', 'revenuePerDateA': 'sum'}),axis=1,)
ordersACummulative.columns = ['date','ordersCummulativeA','revenueCummulativeA',]

ordersBDaily = (
    orders[orders['group'] == 'B'][['date', 'transactionId', 'visitorId', 'revenue']]
    .groupby('date', as_index=False)
    .agg({'transactionId': pd.Series.nunique, 'revenue': 'sum'}))
ordersBDaily.columns = ['date', 'ordersPerDateB', 'revenuePerDateB']

ordersBCummulative = ordersBDaily.apply(
    lambda x: ordersBDaily[ordersBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateB': 'sum', 'revenuePerDateB': 'sum'}),axis=1,)
ordersBCummulative.columns = ['date','ordersCummulativeB','revenueCummulativeB',]

display(ordersACummulative.head())
display(ordersBCummulative.head())
```

	date	ordersCummulativeA	revenueCummulativeA
0	2019-08-01	24	148579
1	2019-08-02	44	242401
2	2019-08-03	68	354874
3	2019-08-04	84	425699
4	2019-08-05	109	549917

	date	ordersCummulativeB	revenueCummulativeB
0	2019-08-01	21	101217
1	2019-08-02	45	266748
2	2019-08-03	61	380996
3	2019-08-04	78	489567
4	2019-08-05	101	581995

Создание объединённой таблицы:

Ввод [31]:

```
df = (
    ordersADaily.merge(
        ordersBDaily, left_on='date', right_on='date', how='left'
    )
    .merge(ordersACummulative, left_on='date', right_on='date', how='left')
    .merge(ordersBCummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsADaily, left_on='date', right_on='date', how='left')
    .merge(visitorsBDaily, left_on='date', right_on='date', how='left')
    .merge(visitorsACummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsBCummulative, left_on='date', right_on='date', how='left'))
display(df.head())
```

	date	ordersPerDateA	revenuePerDateA	ordersPerDateB	revenuePerDateB	ordersCummulat
0	2019-08-01	24	148579	21	101217	
1	2019-08-02	20	93822	24	165531	
2	2019-08-03	24	112473	16	114248	
3	2019-08-04	16	70825	17	108571	
4	2019-08-05	25	124218	23	92428	

Создание таблиц для пользователей групп совершивших хотя бы одну покупку:

Ввод [32]:

```
ordersByUsersA = (
    orders[orders['group'] == 'A']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique}))

ordersByUsersA.columns = ['userId', 'orders']
ordersByUsersB = (
    orders[orders['group'] == 'B']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique})
)
ordersByUsersB.columns = ['userId', 'orders']
display(ordersByUsersA.head())
display(ordersByUsersB.head())
```

	userId	orders
0	8300375	1
1	11685486	1
2	54447517	1
3	66685450	1
4	78758296	1

	userId	orders
0	5114589	1
1	6958315	1
2	8300375	1
3	39475350	1
4	47206413	1

Переменные sampleA и sampleB, в которых пользователям с заказами соответствует число заказов пользователя, а пользователям без заказов — нули.

Ввод [33]:

```
sampleA = pd.concat([ordersByUsersA['orders'],pd.Series(
    0, index=np.arange(df['visitorsPerDateA'].sum() - len(ordersByUsersA['orders'])), name=
sampleB = pd.concat([ordersByUsersB['orders'],pd.Series(
    0, index=np.arange(df['visitorsPerDateB'].sum() - len(ordersByUsersB['orders'])), name=
```

По тесту Манна-Уитни проведвение проверки гипотез:

Но:"В конверсии между группой B и группой A по сырым данным нет статистически значимых различий."

H1:"В конверсии между группой B и группой A по сырым данным есть статистически значимые различия."

Ввод [34]:

```
# p-value для сравнения конверсии между группами:
p_value = stats.mannwhitneyu(sampleA, sampleB)[1]
print("{0:.5f}".format(p_value))

# относительное различие в конверсии между группами
a_b_value = (df['ordersPerDateB'].sum()/df['visitorsPerDateB'].sum())/(
    df['ordersPerDateA'].sum()/df['visitorsPerDateA'].sum())-1
print("{0:.3f}".format(a_b_value))
```

```
0.00840
0.138
```

Проверка гипотезы об отсутствии различий в конверсии по "сырым" данным мжду группами A и B опревергнута так, ка полученной значение меньше 0,05. В конверсии различия есть! При этом наблюдается прирост конверсии групы B к группе A в 13,8%

"Сырые" данные, статистическая значимость различий в среднем чеке в группах:

Проверки гипотезы:

Но:"В значениях среднего чека между группой B и группой A по сырым данным нет статистически значимых различий."

H1:"В значениях среднего чека между группой B и группой A по сырым данным есть статистически значимые различия."

Ввод [35]:

```
# p-value для сравнения различий в среднем чеке между группами:
p_value_ch=stats.mannwhitneyu(orders[orders['group']=='A']['revenue'], orders[orders['group
print('{0:.3f}'.format(p_value_ch))

# относительное различие в среднем чеке между группами
a_b_value_ch = orders[orders['group']=='B']['revenue'].mean()/orders[orders['group']=='A'][
print('{0:.3f}'.format(a_b_value_ch))
```

```
0.365
0.259
```

p-value больше 0,05 значит, что статистически значимых различий по сырым данным в среднем чеке между группами A и B нет. Но относительное различие среднего чека при этом 26%.

"Очищенные" данные, статистическая значимость различий в конверсии между группами:

Получение группы с очищенными данными:

Ввод [36]:

```
usersWithManyOrders = pd.concat(
    [
        ordersByUsersA[ordersByUsersA['orders'] > 2]['userId'],
        ordersByUsersB[ordersByUsersB['orders'] > 2]['userId'],
    ],
    axis=0,)
usersWithExpensiveOrders = orders[orders['revenue'] > 28100]['visitorId']
abnormalUsers = (
    pd.concat([usersWithManyOrders, usersWithExpensiveOrders], axis=0)
    .drop_duplicates())
print(abnormalUsers.head())
```

```
18      199603092
23      237748145
68      611059232
146    1230306981
189    1614305549
dtype: int64
```

Расчёт статистической значимости отсутсвия различий конверсии между группами A и B по тесту Манна-Уитни по очищенным данным:

Ho:"В конверсии между группой B и группой A по сырым данным нет статистически значимых различий."

H1:"В конверсии между группой B и группой A по сырым данным есть статистически значимые различия."

Ввод [37]:

```
sampleAFiltered = pd.concat(
    [ordersByUsersA[np.logical_not(ordersByUsersA['userId'].isin(abnormalUsers))]['orders'],
        pd.Series(0,index=np.arange(df['visitorsPerDateA'].sum() - len(ordersByUsersA['orde

sampleBFiltered = pd.concat(
    [ordersByUsersB[np.logical_not(ordersByUsersB['userId'].isin(abnormalUsers))]['orders'],
        pd.Series(0,index=np.arange(df['visitorsPerDateB'].sum() - len(ordersByUsersB['orde

print('{0:.5f}'.format(stats.mannwhitneyu(sampleAFiltered, sampleBFiltered)[1]))
print('{0:.3f}'.format(sampleBFiltered.mean()/sampleAFiltered.mean()-1))
```

```
0.00714
0.170
```

Как и в случае с сырыми данными статистическая разница достигнута. Различия в конверсии есть. Группа B лучше группы A. Прирост конверсии к группе A 17%

"Очищенные" данные, статистическая значимость различий в средних чеках между группами:

Проверки гипотезы:

Но:"В значениях среднего чека между группой B и группой A по очищенным данным нет статистически значимых различий."

H1:"В значениях среднего чека между группой B и группой A по очищенным данным есть статистически значимые различия."

Ввод [38]:

```
print('{0:.3f}'.format(stats.mannwhitneyu(
    orders[np.logical_and(orders['group'] == 'A',np.logical_not(orders['visitorId'].isin(ab
            orders[np.logical_and(orders['group'] == 'B',np.logical_not(
                orders['visitorId'].isin(abnormalUsers)),)]['revenue'],)[1]))
print("{0:.3f}".format(orders[np.logical_and(orders['group'] == 'B',np.logical_not(
    orders['visitorId'].isin(abnormalUsers)),
            )]['revenue'].mean()
        / orders[np.logical_and(
            orders['group'] == 'A',np.logical_not(orders['visitorId'].isin(abnormalUsers)),
```

```
0.348
-0.029
```

p-value по очищенным данным незначительно снизился, но значение больше 0,05 а значит, что статистически значимых различий по очищенным данным в среднем чеке между группами A и B не появилось. Но появилось относительное различие среднего чека в 2,9% не в пользу группы B.

Выводы:

Не смотря на то, что относительное отличие в среднем чеке по очищенным данным не в пользу группы B, всёже кумулятивный средний чек группы B выше. Также значения кумулятивной выручки выше и происходит её рост. Значения кумулятивной конверсии у группы B также выше и стабилизировались после повышения. Поэтому по результатам анализа тест можно остановить и признать победу группы B.

Ввод [ ]:

```

```