Приоритизация гипотез по увеличению выручки интернет-магазина и анализ результатов проведённого A/B-теста

Содержание

Вместе с отделом маркетинга крупного интернет-магазина подготовлен список гипотез для увеличения выручки.

Требуется:

1. Приоритизировать гипотезы в списке.
2. Запустить A/B-тест и проанализировать его результаты.

Данные хранятся в следующих файлах:

- В файле `hypothesis.csv` перечислены характеристики 9 гипотез по увеличению выручки интернет-магазина.
- Результаты проведённого A/B-теста описаны в файлах `orders.csv` и `visitors.csv`.

Импорт библиотек и определение функций проекта

Импорт библиотек

In [1]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import datetime as dt
from scipy import stats as st
import seaborn as sns
```

Функции загрузки и обзора данных

In [2]:
```
# определение функции обзора данных
# ================================================
# на вход подаётся датафрейм df
# на выходе:
#     - 10 случайных строк df
#     - информация df.info()
#     - количество явных дубликатов в строках df
#     - процент пропусков данных в столбцах df
# ================================================
def data_observe(df):
    row_num = 5    # количество отображаемых строк таблицы

    print('Произвольные строки таблицы:')
    print('============================')
    if len(df) >= row_num:
        display(df.sample(row_num))
    else:
        display(df)

    print('\nИнформация о таблице:')
```

```
    print('====================')
    df.info()

    print('\nКоличество явных дубликатов в таблице:')
    print('====================================')
    print(df.duplicated().sum())

    print('\nПроцент пропусков в столбцах:')
    print('============================')
    display(pd.DataFrame(
                round((df.isna().mean()*100),2), columns=['NaNs, %'])
                .sort_values(by='NaNs, %', ascending=False
            )
        .style.format('{:.2f}')
        .background_gradient('coolwarm')
    )
```

In [3]:
```
# определение кроссплатформенной функции загрузки данных
# ======================================================
# на вход подаётся:
#     file_name - имя файла
# на выходе - датафрейм с загруженными данными
# ======================================================
def open_file(file_name, sep=','):
    pth1 = '/datasets/' + file_name             # яндексовский путь
    pth2 = os.path.join('datasets', file_name)  # мой путь

    if os.path.exists(pth1):
        return pd.read_csv(pth1, sep=sep)
    elif os.path.exists(pth2):
        return pd.read_csv(pth2, sep=sep)
    else:
        print("ERROR: Neither Yandex nor local path is reacable...")

    return pd.DataFrame() # в случае ошибки чтения вернём пустой DataFrame
```

Функция расчёта кумулятивных метрик

In [4]:
```
# определение кроссплатформенной функции расчёта
# кумулятивных метрик A/B-теста
# ======================================================
# на вход подаются:
#     orders - информация о заказах
#     visitors - информация о посетителях интернет-магазина
# на выходе - датафрейм с посчитанными метриками:
#     кумулятивный средний чек по группам тестирования
#     кумулятивная конверсия по группам тестирования
# ======================================================
def get_cummulative_data(orders, visitors):
    # выделим массив уникальных пар значений дат и групп для последующей
    # индексации при подсчёте кумулятивной выручки
    dates_and_groups = orders[['date', 'group']].drop_duplicates()

    # агрегируем значения с накоплением
    orders_agg = (
        dates_and_groups
        .apply(
            # для каждой строки dates_and_groups агрегируем orders по условию
            lambda x : orders[np.logical_and(
                orders['date'] <= x['date'],
                orders['group'] == x['group']
            )]
            .agg(
                {
                    'date' : 'max',
                    'group': 'max',
                    'transaction_id' : 'nunique',
                    'visitor_id' : 'nunique',
                    'revenue' : 'sum'
                }
            ),
            axis=1
        )
        .sort_values(by=['date', 'group'])
    )

    # переименуем колонки
    orders_agg.columns = ['date', 'group', 'num_orders',
                          'num_buyers', 'revenue']

    # агрегируем количество посетителей с накоплением
    visitors_agg = (
        dates_and_groups
        .apply(
            # для каждой строки dates_and_groups агрегируем orders по условию
            lambda x : visitors[np.logical_and(
                visitors['date'] <= x['date'],
                visitors['group'] == x['group']
            )]
            .agg(
                {
                    'date' : 'max',
                    'group': 'max',
                    'visitors' : 'sum'
                }
            ),
            axis=1
        )
        .sort_values(by=['date', 'group'])
    )

    # переименуем колонки
    visitors_agg.columns = ['date', 'group', 'num_visitors']

    # объединяем с orders_agg
    cummulative_data = orders_agg.merge(
        visitors_agg, left_on=['date', 'group'], right_on=['date', 'group']
    )
```

```
    # вычисляем сумулятивную конверсию по группам
    cummulative_data['conversion'] = (
        cummulative_data['num_buyers'] / cummulative_data['num_visitors']
    )

    return cummulative_data
```

Приоритизация гипотез по увеличению выручки

На первом этапе необходимо расставить приоритеты для 9 гипотез по увеличению выручки интернет-магазина.

Обзор данных

Откроем и изучим файл `hypothesis.csv`, в котором перечислены характеристики 9 гипотез по увеличению выручки:

In [5]:
```
hypothesis = open_file('hypothesis.csv', sep=',')
if not hypothesis.empty:
    data_observe(hypothesis)
```

```
Произвольные строки таблицы:
============================
```

	Hypothesis	Reach	Impact	Confidence	Efforts
5	Добавить страницу отзывов клиентов о магазине,...	3	2	2	3
1	Запустить собственную службу доставки, что сок...	2	5	4	10
8	Запустить акцию, дающую скидку на товар в день...	1	9	9	5
7	Добавить форму подписки на все основные страни...	10	7	8	5
2	Добавить блоки рекомендаций товаров на сайт ин...	8	3	7	3

```
Информация о таблице:
=====================
<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

Количество явных дубликатов в таблице:
======================================
0

Процент пропусков в столбцах:
=============================
```

	NaNs, %
Hypothesis	0.00
Reach	0.00
Impact	0.00
Confidence	0.00
Efforts	0.00

In [ ]:
```

```

Таблица `hypothesis` состоит из 9 строк, 5 столбцов. Типы данных в столбцах - object, int64.

Согласно описанию данных, столбцы хранят следующую информацию:

- `Hypothesis` — краткое описание гипотезы;
- `Reach` — охват пользователей по 10-балльной шкале;
- `Impact` — влияние на пользователей по 10-балльной шкале;
- `Confidence` — уверенность в гипотезе по 10-балльной шкале;
- `Efforts` — затраты ресурсов на проверку гипотезы по 10-балльной шкале. Чем больше значение `Efforts`, тем дороже проверка гипотезы.

Столбцы поименованы в смешаном стиле - целесообразно привести названия к нижнему регистру.

Для оценок, хранимых в столбцах `Reach`, `Impact`, `Confidence`, `Efforts` тип int64, очевидно, избыточен. Тем не менее, ввиду малого размера таблицы не будем менять тип столбцов для экономии памяти.

В таблице отсутствуют явные дубликаты и пропуски данных.

Предварительная обработка

По результатам обзора данных таблицы `hypothesis` приведём названия столбцов к нижнему регистру:

In [6]:
```
hypothesis.columns = hypothesis.columns.str.lower()
hypothesis.columns
```

Out[6]:
```
Index(['hypothesis', 'reach', 'impact', 'confidence', 'efforts'], dtype='object')
```

Столбцы поименованы в хорошем стиле.

Приоритизация гипотез

Таблица `hypothesis` содержит данные об охвате пользователей, влиянии на них, уверенности в гипотезе и затратах ресурсов на её проверку. Воспользуемся фреймворками `ICE` и `RICE` для расстановки приоритетов.

Приоритизация с помощью фреймворка ICE

Посчитаем оценку гипотез в рамках фреймворка `ICE` по следубщей формуле:

$$ICE = \frac{Impact \times Confidence}{Efforts}.$$

In [7]:
```
# посчитаем метрику ice
hypothesis['ice'] = (
    round(
        hypothesis['impact'] * hypothesis['confidence'] / hypothesis['efforts'],
        1
    )
)
# ранжируем гипотезы
pd.options.display.max_colwidth = 150
hypothesis[['hypothesis','ice']].sort_values(by='ice', ascending=False)
```

Out[7]:

	hypothesis	ice
8	Запустить акцию, дающую скидку на товар в день рождения	16.2
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	13.3
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	11.2
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	8.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	7.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	1.3
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	1.1
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	1.0

В рамках фреймворка `ICE` наиболее перспективными с точки зрения увеличения выручки представляются гипотезы (в порядке убывания приоритета):

1. Запустить акцию, дающую скидку на товар в день рождения
1. Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей
1. Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок
1. Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию
1. Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа

Остальные гипотезы получили ранг, более чем в 3 раза меньший по сравнению с перечисленными.

Учтём при расчёте приоритетов параметр охвата пользователей `Reach`.

Приоритизация с помощью фреймворка RICE

Рассчитаем оценку гипотез в рамках фреймворка `RICE` по следубщей формуле:

$$RICE = \frac{Reach \times Impact \times Confidence}{Efforts}.$$

In [8]:
```
# посчитаем метрику rice
hypothesis['rice'] = (
    round(
        hypothesis['reach'] * hypothesis['impact'] * hypothesis['confidence'] /
        hypothesis['efforts'],
        1
    )
)

# ранжируем гипотезы
pd.options.display.max_colwidth = 150
hypothesis[['hypothesis','ice','rice']].sort_values(by='rice', ascending=False)
```

Out[8]:

	hypothesis	ice	rice
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	11.2	112.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	7.0	56.0
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	13.3	40.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	8.0	40.0
8	Запустить акцию, дающую скидку на товар в день рождения	16.2	16.2
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	1.1	9.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2.0	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	1.3	4.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	1.0	3.0

С учётом охвата пользователей ситуация изменилась:

1. Ярким лидером стала занимавшая в оценке `ICE` 3-ю позицию гипотеза "Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок" (лидирует с 2-кратным перевесом относительно 2 позиции)
1. На втроре место поднялась замыкавшая top-5 по версии `ICE` гипотеза "Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа"

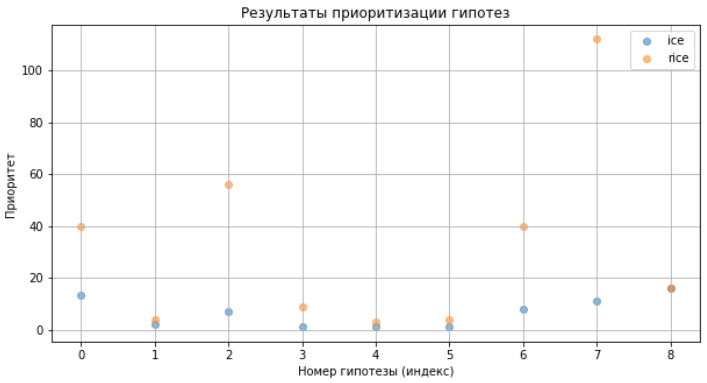
1. Гипотезы "Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей" и "Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию" по-прежнему занимают средние строчки top-5

1. Замыкает top-5 по версии `RICE` прошлый лидер - гипотеза "Запустить акцию, дающую скидку на товар в день рождения" (очевидно, из-за низкого мгновенного охвата пользователей, родившихся в конкретный день)

1. Прочие гипотезы имеют низкий приоритет в рамках обоих фреймворков

Отобразим приоритеты на точечной диаграмме:

In [9]:
```
# отобразим распределение в виде точечной диаграммы
x_values = hypothesis.index # порядковые номера (замена id)
plt.figure(figsize=(10, 5))
plt.scatter(x_values, hypothesis['ice'], label='ice', alpha=.5)
plt.scatter(x_values, hypothesis['rice'], label='rice', alpha=.5)
plt.title('Результаты приоритизации гипотез')
plt.xlabel('Номер гипотезы (индекс)')
plt.ylabel('Приоритет')
plt.grid()
plt.legend()
plt.show()
```



Выводы

По итогу приоритизации бизнес-гипотез по увеличению выручки интернет-магазина можно заключить:

1. Состав top-5 гипотез в рамках фреймворков `ICE` и `RICE` совпадает.

1. Введение в рассмотрение охвата пользователей существенно перераспределяет приоритеты.

1. В рамках фреймворка `ICE` сложнее определить, с какой именно гипотезы начать проверку - первые три гипотезы имеют близкие приоритеты.

1. Фреймворк `RICE` исправляет эту ситуацию, выделив ярко лидирующую гипотезу под номером 7.

1. Проверку представляется целесообразным начинать с гипотезы **"Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок".**

Анализ результатов A/B-теста

Теперь необходимо проанализировать результаты проведённого A/B-теста гипотезы "Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок", описанные в файлах `orders.csv` и `visitors.csv`, содержащих данные о заказах и количестве посетителей интернет-магазина пользователями.

Основные статистические гипотезы, которые проверялись в ходе теста:

1. О различии конверсии пользователей групп A и B (статистическая гипотеза H0 - конверсии групп A и B одинаковы).
2. О различии среднего чека пользователей групп A и B (статистическая гипотеза H0 - средние чеки групп A и B одинаковы).

Начнём с изучения данных.

Обзор данных

Откроем и изучим файл `orders.csv`:

In [10]:
```
orders = open_file('orders.csv', sep=',')
if not orders.empty:
    data_observe(orders)
```

```
Произвольные строки таблицы:
============================
```

	transactionId	visitorId	date	revenue	group
487	2786000406	2744017357	2019-08-19	1490	A
848	1063646926	102981183	2019-08-05	14489	B
846	4256972642	3281496343	2019-08-05	1990	A
455	2781850870	2954449915	2019-08-06	3070	B
730	3335803766	477780734	2019-08-30	7300	A

```
Информация о таблице:
====================
<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

Количество явных дубликатов в таблице:
======================================
0

Процент пропусков в столбцах:
=============================
```

	NaNs, %
transactionId	0.00
visitorId	0.00
date	0.00
revenue	0.00
group	0.00

Таблица `orders` состоит из 1197 строк, 5 столбцов. Типы данных в столбцах - object, int64.

Согласно описанию данных, столбцы хранят следующую информацию:

- `transactionId` — идентификатор заказа;
- `visitorId` — идентификатор пользователя, совершившего заказ;
- `date` — дата, когда был совершён заказ;
- `revenue` — выручка заказа;
- `group` — группа A/B-теста, в которую попал заказ.

Столбцы поименованы в смешаном стиле - целесообразно привести названия к стилю "snake_case".

Для значений, хранимых в столбцах `transactionId`, `visitorId`, `revenue` тип int64, очевидно, достаточен.

На этапе предварительной обработки целесообразно достоверно обеспечить формат даты в столбце `date` путём преобразования типов.

В таблице отсутствуют явные дубликаты и пропуски данных.

Откроем и изучим файл `visitors.csv`:

In [11]:
```
visitors = open_file('visitors.csv', sep=',')
if not visitors.empty:
    data_observe(visitors)
```

```
Произвольные строки таблицы:
============================
```

	date	group	visitors
19	2019-08-20	A	575
58	2019-08-28	B	654
46	2019-08-16	B	413
24	2019-08-25	A	621
22	2019-08-23	A	468

```
Информация о таблице:
====================
<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

Количество явных дубликатов в таблице:
======================================
0

Процент пропусков в столбцах:
=============================
```

	NaNs, %
date	0.00
group	0.00
visitors	0.00

Таблица `visitors` состоит из 62 строк, 3 столбцов. Типы данных в столбцах - object, int64.

Согласно описанию данных, столбцы хранят следующую информацию:

- `date` — дата;
- `group` — группа A/B-теста;
- `visitors` — количество пользователей в указанную дату в указанной группе A/B-теста.

Столбцы поименованы в хорошем стиле.

Для значений, хранимых в столбце `visitors` тип int64, очевидно, избыточен. Тем не менее, ввиду малого размера таблицы не будем менять тип столбца для экономии памяти.

На этапе предварительной обработки целесообразно достоверно обеспечить формат даты в столбце `date` путём преобразования типов.

В таблице отсутствуют явные дубликаты и пропуски данных.

Предварительная обработка данных проведённого A/B-теста

По результатам обзора данных таблицы `orders` приведём названия столбцов к стилю "snake_case":

In [12]:
```
orders.rename(
    columns={'transactionId' : 'transaction_id',
             'visitorId' : 'visitor_id'},
    inplace=True
)
orders.columns
```

Out[12]:
```
Index(['transaction_id', 'visitor_id', 'date', 'revenue', 'group'], dtype='object')
```

Столбцы поименованы в хорошем стиле.

Обеспечим формат даты в соответствующих столбцах таблиц `orders` и `visitors`:

In [13]:
```
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')
)
```

Предварительная обработка завершена.

Исследование аномалий данных A/B-теста

Прежде чем проверять и интерпретировать результаты проведённого A/B-теста выясним, не закрались ли ошибки в итоговые данные. Обзор данных показал, что явные дубликаты и пропуски отсутствуют, однако ***возможны*** следующие проблемы в данных:

1. Для одинаковых уникальных комбинаций даты и группы в таблице `visitors` указаны различные значения количества посетителей.
2. Для одинаковых уникальных комбинаций `transaction_id`, `visitor_id`, `date` и `group` в таблице `orders` указаны различные значения `revenue`.
3. Одни и те же пользователи попали в обе группы теста.

Проверим данные на наличие этих проблем:

In [14]:
```
# 1 проблема
(
    visitors.groupby(by=['date', 'group'])
    .agg({'visitors':'count'}) # считаем количество visitors в группировке
    .reset_index()
    .query('visitors > 1')
    .shape[0] # считаем количество строк
)
```

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

In [15]:
```
# 2 проблема
(
    orders.groupby(by=['transaction_id', 'visitor_id', 'date', 'group'])
    .agg({'revenue':'count'}) # считаем количество revenue в группировке
    .reset_index()
    .query('revenue > 1')
    .shape[0] # считаем количество строк
)
```

Out[15]:
```
0
```

Первая и вторая потенциальные проблемы в данных не обнаружены.

Проверим наличие пользователей, попавших в обе группы:

In [16]:
```
# 3 проблема
group_a_users = list(orders.query('group == "A"')['visitor_id'].unique())
group_b_users = list(orders.query('group == "B"')['visitor_id'].unique())

intersection = []
for user in group_a_users:
    if user in group_b_users:
        intersection.append(user)
# вывод результата
print(f'{len(intersection)} пользователей попали в обе группы:')
print(
    round(len(intersection) * 100 / len(group_a_users), 2),
    '% от объёма группы A'
)
print(
    round(len(intersection) * 100 / len(group_b_users), 2),
    '% от объёма группы B'
)
```

```
58 пользователей попали в обе группы:
11.53 % от объёма группы A
9.9 % от объёма группы B
```

Действительно, 58 посетителей интернет-магазина по какой-то причине попали в обе группы теста. Данные по ним могут исказить результат A/B-теста, поэтому представляется целесообразным удалить их из таблицы `orders`:

In [17]:
```
orders = orders.query('visitor_id not in @intersection')
```

Перейдём к проверке результатов A/B-теста.

Определение динамики кумулятивной выручки по группам

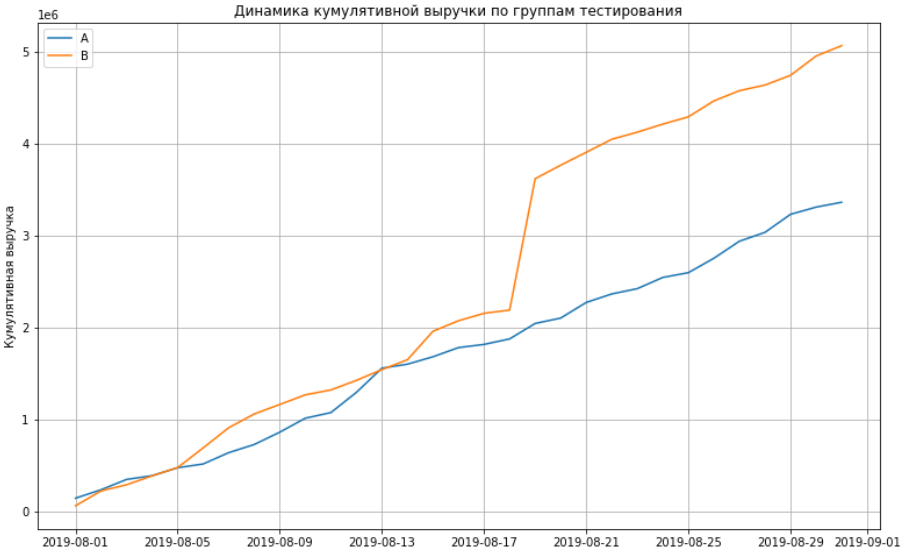
Проследим динамику кумулятивной выручки по группам за период тестирования. Данная метрика напрямую связана с целевой метрикой "размер среднего чека".

Для нахождения кумулятивной выручки просумминуем на каждую дату теста выручку по группам за текущую и предыдущие даты.

Определим и визуализируем кумулятивную выручку по группам теста:

In [18]:
```
# агрегируем значения с накоплением и посчитаем кумулятивные метрики
cummulative_data = get_cummulative_data(orders, visitors)

# построим графики кумулятивной выручки по группам тестирования
plt.figure(figsize=(13, 8))
# для группы A
plt.plot(
    cummulative_data.query('group == "A"')['date'],
    cummulative_data.query('group == "A"')['revenue'],
    label='A'
)
# для группы B
plt.plot(
    cummulative_data.query('group == "B"')['date'],
    cummulative_data.query('group == "B"')['revenue'],
    label='B'
)
plt.legend()
plt.title('Динамика кумулятивной выручки по группам тестирования')
plt.ylabel('Кумулятивная выручка')
plt.grid()
plt.show()
```



В течение первых 5 дней теста выручка групп практически не отличалась. Затем с 6 по 12 августа выручка группы B стабильно превышала группу A. 13 августа значения метрики у групп сравнялись, а в дальнейшем выручка группы B стабильно превышает группу A.

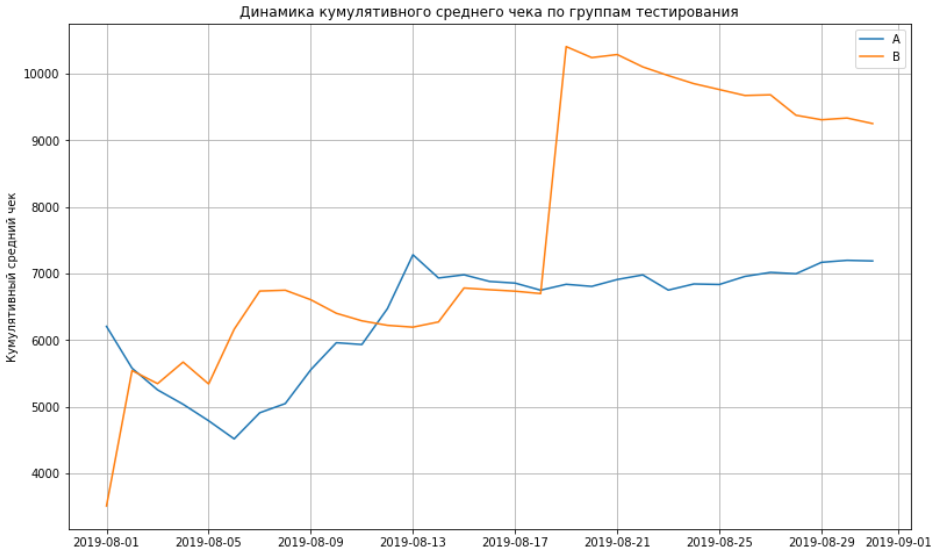
Отметим, что в периоды 12-13 и 18-19 августа наблюдаются скачки метрики для групп A и B соответственно. Это может свидетельствовать об аномальных покупках (с высокой выручкой) в эти дни.

Определение динамики кумулятивного среднего чека по группам

Посчитаем целевую метрику теста - размер кумулятивного среднего чека по группам. Для этого разделим кумулятивную выручку на агрегированное количество заказов на каждую дату.

Визуализируем динамику кумулятивного среднего чека по группам теста:

In [19]:
```
# построим графики кумулятивного среднего чека по группам тестирования
plt.figure(figsize=(13, 8))
# для группы A
plt.plot(
    cummulative_data.query('group == "A"')['date'],
    cummulative_data.query('group == "A"')['revenue'] /
    cummulative_data.query('group == "A"')['num_orders'],
    label='A'
)
# для группы B
plt.plot(
    cummulative_data.query('group == "B"')['date'],
    cummulative_data.query('group == "B"')['revenue'] /
    cummulative_data.query('group == "B"')['num_orders'],
    label='B'
)
plt.legend()
plt.title('Динамика кумулятивного среднего чека по группам тестирования')
plt.ylabel('Кумулятивный средний чек')
plt.grid()
plt.show()
```



Как видно из графика, кумулятивный средний чек контрольной группы A после падения в начале тестирования растёт, начиная с 7 августа, до отметки 7000, возле которой колеблется с 13 августа и до окончания тестирования.

Средний чек в группе B с самого начала бурно растёт, затем слегка снижается, колеблется в районе 6800-6900 (около установившегося среднего чека группы A).

Затем, 19 августа, наблюдается резкий скачкообразный рост среднего чека группы B до значения более чем 10000, после чего средний чек группы B до конца среза данных плавно снижался в направлении установившегося значения группы A.

Это подтверждает наши предположения о наличии 19 августа одного или более заказов с аномально высокой выручкой.

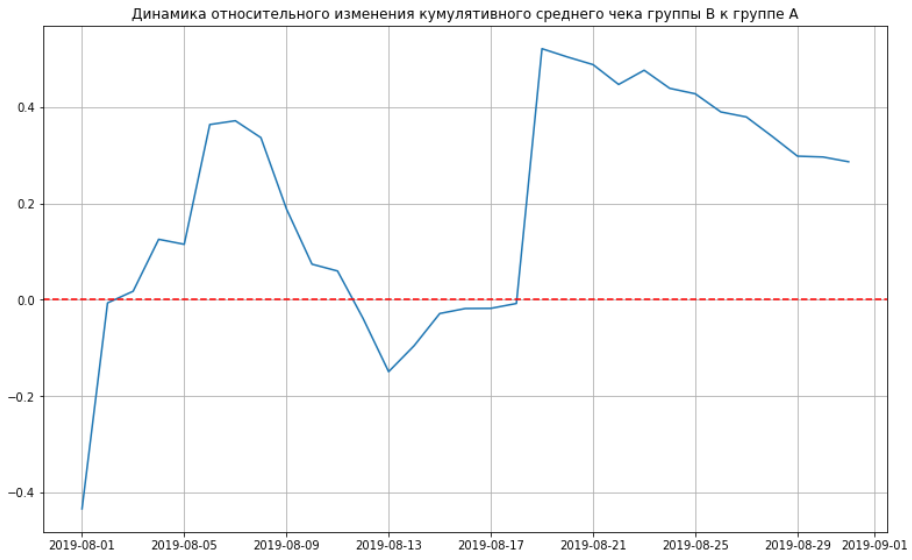
Определение динамики относительного изменения кумулятивного среднего чека группы B к группе A

Посмотрим на отношение кумулятивного среднего чека группы B к группе A в динамике по ходу тестирования. Для этого разделим кумулятивный средний чек группы B на кумулятивный средний чек группы A:

In [20]:
```
# собираем данные по группам в один датафрейм
total_cumulative_revenue = (
    cummulative_data.query('group == "A"')
    .merge(
        cummulative_data.query('group == "B"'),
        left_on='date',
        right_on='date',
        how='left',
        suffixes=['_a', '_b']
    )
)

# строим график отношения средних чеков
plt.figure(figsize=(13, 8))

plt.plot(
    total_cumulative_revenue['date'],
    (total_cumulative_revenue['revenue_b'] / total_cumulative_revenue['num_orders_b']) /
    (total_cumulative_revenue['revenue_a'] / total_cumulative_revenue['num_orders_a']) - 1
    # вычитаем 1, чтобы получить превышение / занижение
)
plt.title('Динамика относительного изменения кумулятивного среднего чека группы B к группе A')
plt.grid()
plt.axhline(y=0, color='red', linestyle='--')
plt.show()
```



На рисунке красным пунктиром отмечена условная граница кумулятивного среднего чека контрольной группы A. График отражает динамику относительного превышения среднего чека группы B над средним чеком группы A.

Можно видеть, что в первый день теста группа B показала себя на примерно 40% хуже группы A. Затем к концу первой недели уверенно обгоняла контрольную группу. Затем всю вторую неделю превышение среднего чека группы B снижалось, и к 13 августа она проигрывала контрольной группе 15%. Вероятно, это связано с наличием в этот период аномальных покупок в группе A, или опережающей конверсией в ней и увеличением количества заказов.

С 13 по 18 день теста средний чек группы B стабилизируется и приближается к группе A.

19 августа в группе B происходит резкий скачок, отмеченный нами ранее, после чего превышение среднего чека начинает стабилизацию в направлении границы группв A.

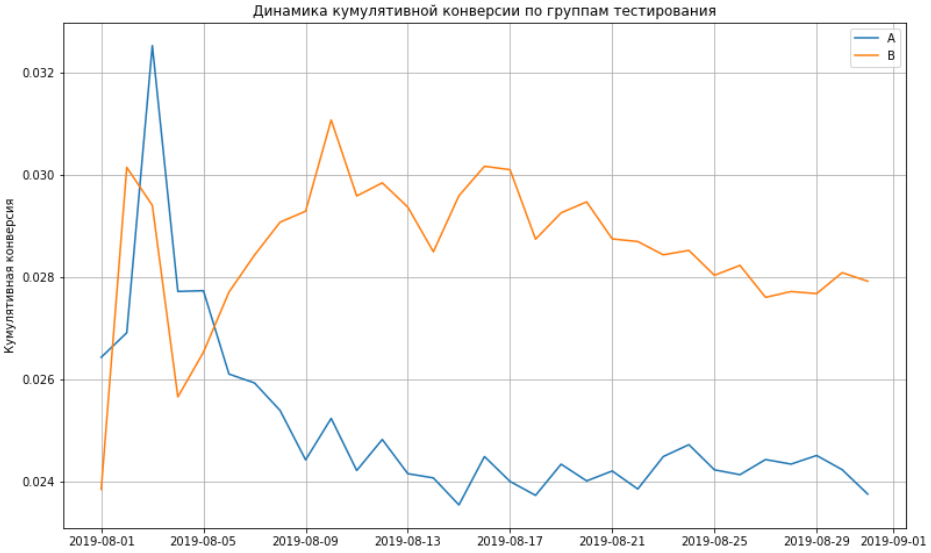
Вероятно, 19 августа имели место один или несколько очень дорогих заказов в группе B.

Определение динамики кумулятивной конверсии по группам

Конверсией будем считать превращение посетителей в уникальных покупателей. Кумулятивные метрики, в том числе и конверсию, мы рассчитали ранее.

Визуализируем динамику кумулятивной конверсии по группам тестирования:

In [21]:
```
# построим графики кумулятивной выручки по группам тестирования
plt.figure(figsize=(13, 8))
# для группы A
plt.plot(
    cummulative_data.query('group == "A"')['date'],
    cummulative_data.query('group == "A"')['conversion'],
    label='A'
)
# для группы B
plt.plot(
    cummulative_data.query('group == "B"')['date'],
    cummulative_data.query('group == "B"')['conversion'],
    label='B'
)
plt.legend()
plt.title('Динамика кумулятивной конверсии по группам тестирования')
plt.ylabel('Кумулятивная конверсия')
plt.grid()
plt.show()
```



В начале теста кумулятивная конверсия группы B растёт в среднем медленнее, чем в контрольной группе A. При этом 5-6 августа конверсии обеих групп примерно сравниваются, после чего наблюдается достаточно резкий рост кумулятивной конверсии группы B на фоне постепенного спада и стабилизации конверсии группы A.

Кумулятивная конверсия контрольной группы A к концу 2 недели теста стабилизируется на уровне 2.4%.

Конверсия в группе B в течение второй и третьей недель теста переживает скачкообразные колебания в районе отметки 3%, после чего к концу среза тестирования наблюдаются признаки её стабилизации в районе 2.8%.

Предварительно можно заключить, что конверсия группы B по результатам теста превышает конверсию группы A.

Определение динамики относительного изменения кумулятивной конверсии группы B к группе A

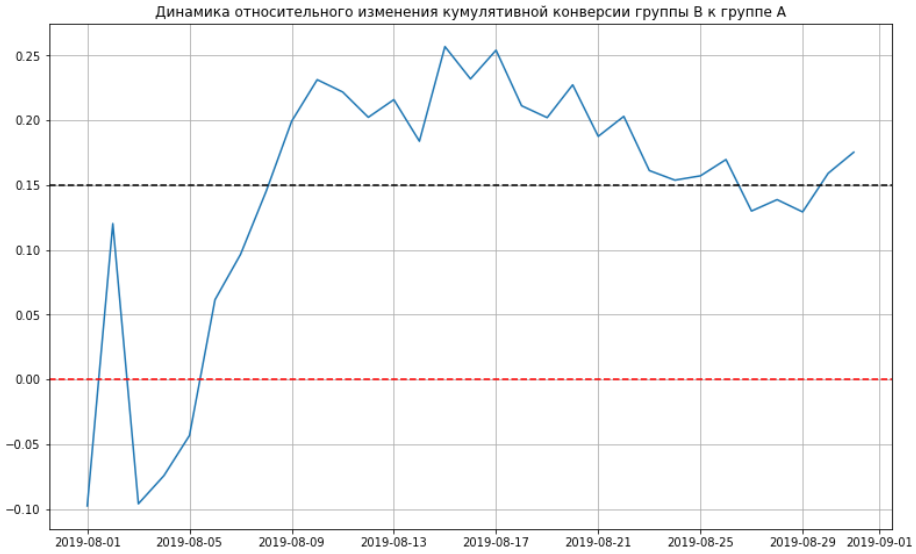
Аналогично среднему чеку, определим и визуализируем динамику относительного изменения кумулятивной конверсии группы B к группе A:

In [22]:
```
# собираем данные по группам в один датафрейм
cumulative_conversion = (
    cummulative_data.query('group == "A"')
    .merge(
        cummulative_data.query('group == "B"'),
        left_on='date',
        right_on='date',
        how='left',
        suffixes=['_a', '_b']
    )
)

# строим график отношения средних чеков
plt.figure(figsize=(13, 8))

plt.plot(
    cumulative_conversion['date'],
    (cumulative_conversion['conversion_b'] / cumulative_conversion['conversion_a']) - 1
    # вычитаем 1, чтобы получить превышение / занижение
)
plt.title('Динамика относительного изменения кумулятивной конверсии группы B к группе A')
```

```
plt.grid()
plt.axhline(y=0, color='red', linestyle='--')
plt.axhline(y=.15, color='black', linestyle='--')
plt.show()
```



Динамика относительного изменения конверсии группы B к группе A подтверждает сделанные выше выводы:

- в первую неделю тестирования, не считая однодневного выброса, конверсия группы B ниже группы A на 5-10%;
- начиная с 6 августа, конверсия группы B стабильно превышает конверсию группы A;
- при этом к концу второй недели она пытается закрепиться на уровне 20%, на третьей неделе увеличивается до 25% и начинает плавно снижаться;
- на четвёртой неделе тестирования превышение конверсии группы B над конверсией группы A стабилизируется на уровне 15%.

Вывод: кумулятивная конверсия тестовой группы превышает конверсию контрольной в среднем на 15%.

Определение статистических выбросов в A/B-тесте

Выше мы определили, что на имеющемся срезе данных A/B-теста средний чек и конверсия для тестовой группы B превышают значения этих метрик для контрольной группы. Вместе с тем, была зафиксирована возможность наличия аномальных покупок.

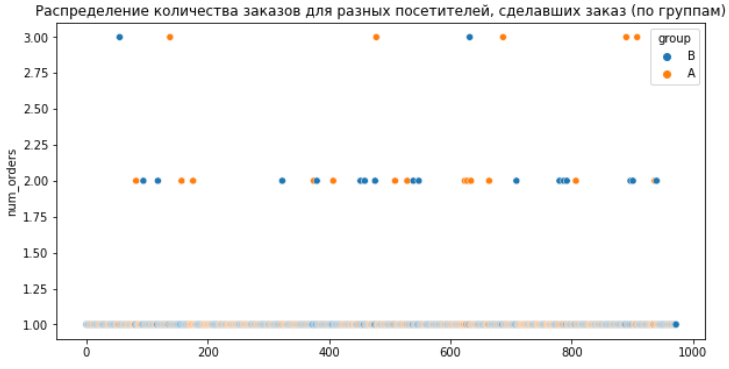
Определим статистические выбросы в результатах теста. Рассмотрим, как распределено количество заказов для уникальных покупателей:

In [23]:
```
# посчитаем количество заказов для каждого покупателя
orders_by_visitors = (
    orders
    .groupby(by='visitor_id', as_index=False)
    .agg({'transaction_id':'nunique',
          'group':'max'})
)
orders_by_visitors.columns = ['visitor_id', 'num_orders', 'group']

# отобразим распределение в виде точечной диаграммы
plt.figure(figsize=(10, 5))

x_values = orders_by_visitors.index
plt.title('Распределение количества заказов для разных посетителей, сделавших заказ (по группам)')
sns.scatterplot(
    x=x_values,
    y='num_orders',
    hue="group",
    data=orders_by_visitors
)
plt.show()

# выведем перцентили 90, 95, 99
print('Перцентили 90, 95, 99:', np.percentile(orders_by_visitors['num_orders'], [90, 95, 99]))
```




```
Перцентили 90, 95, 99: [1. 1. 2.]
```

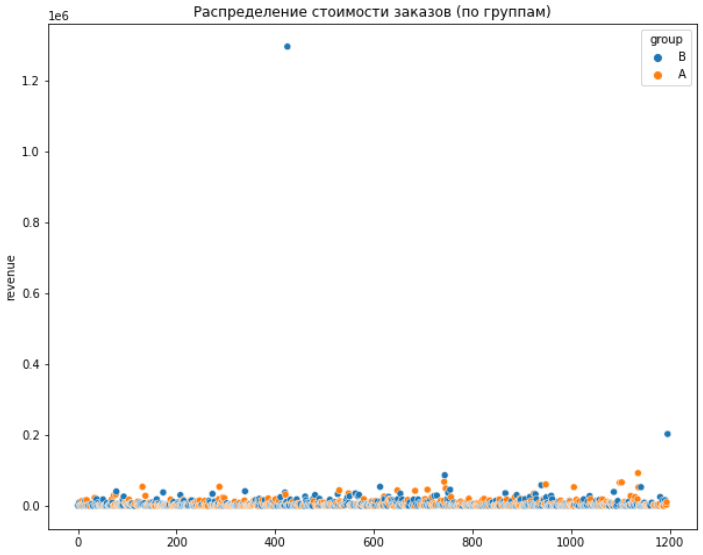
Очевидно, даже 2 заказа - достаточно редкое событие в данном тесте. будем учитывать это при оценке статистической значимости полученных результатов. А именно - будем считать выбросами 2 заказа и более.

Рассмотрим, как распределена стоимость заказов:

In [24]:
```
# отобразим распределение в виде точечной диаграммы
plt.figure(figsize=(10, 8))
```

```
x_values = orders.index
plt.title('Распределение стоимости заказов (по группам)')
sns.scatterplot(
    x=x_values,
    y='revenue',
    hue="group",
    data=orders
)
plt.show()

# выведем перцентили 90, 95, 99
print('Перцентили 90, 95, 99:', np.percentile(orders['revenue'], [90, 95, 99]))
```




```
Перцентили 90, 95, 99: [17990. 26785. 53904.]
```

А вот и аномальные заказы в группе B - на 200000 и на 1.2 млн. При этом 95% заказов были сделаны на сумму до 27000, а 99% - на сумму до 54000. Будем считать выбросами заказы на сумму 27000 и более и проверим повлияет ли их исключение на статистическую значимость результатов теста.

Определение статистической значимости различий в конверсии между группами по «сырым» данным

Для начала определим статистическую значимость различий в конверсии между группами по исходным, "сырым" данным. Для этого воспользуемся непараметрическим статистическим тестом Манна-Уитни.

Напомним, проверяется статистическая гипотеза о равенстве конверсии между группами.

In [25]:
```
# рассчитаем статистическую значимость различий в конверсии
print(
    'raw conversion p-value:',
    '{0:.9f}'
    .format(
        st.mannwhitneyu(cummulative_data.query('group == "A"')['conversion'],
                        cummulative_data.query('group == "B"')['conversion'])[1]
    )
)
```

```
raw conversion p-value: 0.000000012
```

Напомним, что мы проверяем гипотезу о различии конверсии пользователей групп A и B:

- статистическая гипотеза H0 - конверсии групп A и B одинаковы;
- альтернатива H1 - конверсии групп A и B не одинаковы.

Очевидно, полученное значение p-value заведомо меньше любого разумного уровня статистической значимости, и нам следует отвергнуть гипотезу о равенстве конверсий в группах.

Поскольку ранее мы выяснили, что **конверсия целевой группы B установилась на уровне, на 15% превышающем конверсию группы A, то можно утверждать, что это статистически обоснованный результат**.

Определение статистической значимости различий в среднем чеке заказа между группами по «сырым» данным

Другой метрикой, которую мы оценивали в тесте, является значение среднего чека. Для начала определим статистическую значимость различий в чеке между группами по исходным, "сырым" данным. Для этого воспользуемся непараметрическим статистическим тестом Манна-Уитни.

Напомним, проверяется статистическая гипотеза о равенстве средних чеков между группами.

In [26]:
```
# рассчитаем статистическую значимость различий в чеках
print(
    'raw orders mean check p-value:',
    '{0:.3f}'
    .format(
        st.mannwhitneyu(orders.query('group == "A"')['revenue'],
                        orders.query('group == "B"')['revenue'])[1]
    )
)
```

```
raw orders mean check p-value: 0.829
```

При таком уровне статистической значимости мы не можем отвергнуть нулевую гипотезу о равенстве средних чеков. Возможно, велико влияние выбросов. Очистим данные и повторно оценим статистическую значимость.

Определение статистической значимости различий в конверсии между группами по «очищенным» данным

Очистим данные от выбросов - удалим пользователей, сделавших 2 и более заказов, а также тех, кто заказывал на сумму 27000 и более:

In [27]:
```
# выделим идентификаторы анормальных пользователей
abnormal_visitors = pd.concat(
    [
        # аномальные конверсии
        orders_by_visitors.query('num_orders >= 2')['visitor_id'],
        # аномальные суммы заказов
        orders.query('revenue >= 27000')['visitor_id']
    ],
    axis=0
).drop_duplicates().sort_values()

# выведем количество аномальных покупателей
abnormal_visitors.shape[0]
```

Out[27]: 86

Итак, в разряд аномальных покупателей попали 86 человек.

In [28]:
```
# выведем количество заказов в "сырых" данных
print(orders.shape[0])

clean_orders = orders.query('visitor_id not in @abnormal_visitors')

# выведем количество заказов в очищенных данных
print(clean_orders.shape[0])
```

```
1016
887
```

После очистки количество заказов сократилось с 1016 до 887. Проверим, как очистка повлияла на уровни статистической значимости:

In [29]:
```
# агрегируем значения с накоплением и посчитаем кумулятивные метрики
# для "чистых" данных
clean_cummulative_data = get_cummulative_data(clean_orders, visitors)

# рассчитаем статистическую значимость различий в конверсии
print(
    'clean conversion p-value:',
    '{0:.10f}'
    .format(
        st.mannwhitneyu(clean_cummulative_data.query('group == "A"')['conversion'],
                        clean_cummulative_data.query('group == "B"')['conversion'])[1]
    )
)
```

```
clean conversion p-value: 0.0000000004
```

Вероятность признать конверсии одинаковыми еще более сократилась!

Определение статистической значимости различий в среднем чеке заказа между группами по «очищенным» данным

Определим статистическую значимость различий в чеке между группами по очищенным данным:

In [30]:
```
# рассчитаем статистическую значимость различий в чеке
print(
    'clean orders mean check p-value:',
    '{0:.3f}'
    .format(
        st.mannwhitneyu(clean_orders.query('group == "A"')['revenue'],
                        clean_orders.query('group == "B"')['revenue'])[1]
    )
)
```

```
clean orders mean check p-value: 0.727
```

Вероятность, с которой можно считать чеки групп A и B одинаковым, сократилась с 82.9% до 72.7%. Однако при таком уровне статистической значимости мы по-прежнему не можем считать чеки групп A и B различными.

Решение по результатам теста

Основной целью гипотезы, проверяемой с помощью A/B-теста, являлось увеличение выручки. В ходе анализа среза результатов мы выяснили что:

1. С достаточным уровнем статистической значимости установившаяся конверсия в группе B превышает установившуюся конверсию в группе A на 15%.
1. Средний чек в группе B к концу рассматриваемого среза теста превышает средний чек в группе A на порядка 30%.
1. Однако графики демонстрируют продолжающуюся тенденцию к снижению как непосредственно среднего чека в группе B, так и его превышения по отношению к установившемуся на уровне 7000 среднему чеку группы A.
1. Оцененный уровень статистической значимости не позволяет считать средние чеки групп A и B различными.

В этих условиях представляется целесообразным продолжить тест еще по крайней мере на 1 неделю.

Общий вывод

В рамках проведённого исследования мы:

- приоритизировали и выбрали наиболее приоритетную гипотезу для A/B-теста;
- провели A/B-тест и проанализировали его результаты.

1. С учётом охвата пользователей (фреймворк `RICE`) наиболее приоритетной признана следующая гипотеза о путях увеличения выручки интернет-магазина: **"Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок"**.
1. Проведённый A/B-тест гипотезы показал:
   A. С достаточным уровнем статистической значимости установившаяся конверсия в группе B превышает установившуюся конверсию в группе A на 15%.

B. Средний чек в группе B к концу рассматриваемого среза теста превышает средний чек в группе A на 30%.
C. Однако графики демонстрируют продолжающуюся тенденцию к снижению как непосредственно среднего чека в группе B, так и его превышения по отношению к установившемуся на уровне 7000 среднему чеку группы A.
D. Оцененный уровень статистической значимости не позволяет считать средние чеки групп A и B различными.

1. **В этих условиях представляется целесообразным продолжить A/B-тест еще по крайней мере на 1 неделю.**