

ВСТУПЛЕНИЕ

Добрый день, уважаемые коллеги! Рада вас приветствовать на презентации хода исследования и его результатов.

Важность данного исследования определяется необходимостью разобраться в поведении пользователей при использовании мобильного приложения для продажи продуктов питания.

Описание проекта:

В ходе работы изучим воронку продаж. Узнаем, как пользователи доходят до покупки и сколько пользователей доходит до покупки, а сколько — «застревает» на предыдущих шагах. Выясним какие именно шаги становятся самыми провальными, так как на них "отваливается" наибольшее число потенциальных покупателей. Также проверим как именно повлияет на поведение пользователей изменение используемых шрифтов в приложении.

Цель проекта:

Изучить поведение пользователей мобильного приложения, выявить слабые места и потенциально увеличить продажи компании.

Ожидаемые результаты проекта:

- Увеличение количества покупателей дошедших до совершения покупки.
- Выявление наиболее эффективных дизайнов шрифтов приложения.
- Улучшение понимания поведения пользователей.
- Повышение эффективности и продаж кампаний.

ПЛАН ИССЛЕДОВАНИЯ

1. Загрузка и подготовка данных

- 1.1 Предварительное знакомство с данными. Переименование столбцов.
- 1.2 Обработка пропусков.
- 1.3 Изменение таблицы, добавление необходимых полей.

2. Изучение и проверка данных

- 2.1 Количественное изучение данных.
- 2.2 Изучение временных рамок исследования:
 - 2.2.1 Проверка процента сохраненных данных и состава экспериментальных групп

3. Изучение воронки событий

3.1 Изучение логов событий.

3.2 Формирование воронки:

3.2.1 Подсчёт количества пользователей совершили каждое из этих событий.

3.2.2 Выявление плана порядка событий.

3.3 Подсчёт долей пользователей проходящие через воронку.

4. Изучение результатов эксперимента

4.1 Контрольные проверки:

4.1.1 Подсчёт количества пользователей в каждой экспериментальной группе

4.2 Определение самого популярного события для контрольных групп

4.3 Определение самого популярного события для экспериментальной групп

4.4 Сравнение контрольных и экспериментальных групп

4.5 Аргументация выбранной стратегии, подсчёт количества статистических проверок гипотез.

Загрузка и подготовка данных

Предварительное знакомство с данными.
Переименование столбцов.

Переменные используемы в работе:

1. **logs_exp** - таблица с логами событий

2. **count_logs_of_date** - таблица числа событий по дням

3. **logs_by_user** - таблица с числом каждого события по пользователям

4. **logs_counter** - таблица с количеством каждого события

5. **logs_by_user_counter** - таблица с количеством уникальных событий совершенное каждым пользователем

```
In [1]: # Ипорты необходимых библиотек
import pandas as pd
pd.options.display.max_colwidth = 100
from matplotlib import pyplot as plt
from datetime import timedelta, date
import numpy as np
```

```
import math as mth
import seaborn as sns
import scipy.stats as stats
import statsmodels.api as sm
import warnings
warnings.simplefilter('ignore')
```

In [2]: *# Объявление необходимых функций потребующихся в процессе обработки и анализа да*

In [3]: *# Вычисление доли события*

```
def users_per_event_group(group, data):
    event_counter = logs_exp[logs_exp['exp_id'] == group].groupby('event_name')[
    fraction = proportion_per_event = round(event_counter / logs_exp[logs_exp['e
    return event_counter, fraction, logs_exp[logs_exp['exp_id'] == group]['device
```

In [4]:

```
def users_per_event_group_2(group, data):
    event_counter = logs_exp[logs_exp['exp_id'] != group].groupby('event_name')[
    return event_counter, logs_exp[logs_exp['exp_id'] != group]['device_ID_hash'
```

In [5]:

```
def determination_stat_sign(x, y, n_1, n_2):
    alpha = 0.05
    for i in range(len(x.values)):
        print(f'Для события {x.index[i]}')
        purchases = np.array([x.values[i], y.values[i]])
        leads = np.array([n_1, n_2])

        p1 = purchases[0] / leads[0]
        p2 = purchases[1] / leads[1]

        p_combined = (purchases[0] + purchases[1]) / (leads[0] + leads[1])
        difference = p1 - p2

        z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1 / lea
        distr = stats.norm(0, 1)
        p_value = (1 - distr.cdf(abs(z_value))) * 2

        print('p-значение: ', p_value)

        if p_value < alpha:
            print('Отвергаем нулевую гипотезу: между долями есть значимая разниц
        else:
            print('Не получилось отвергнуть нулевую гипотезу, нет оснований счит
```

In [6]: *# Проверка стат. значимости и отличий средних значений для объем контр. группы*

```
def determination_stat_sign2(group_1, event_name, logs_exp):
    sample_1 = logs_exp[(logs_exp['exp_id'] != group_1) & (logs_exp['event_name'
    sample_2 = logs_exp[(logs_exp['exp_id'] == group_1) & (logs_exp['event_name'
    print(f'Прверка статистически достоверных отличий между контрольной и экспе
    result = stats.ttest_ind(sample_1, sample_2, equal_var=False)
    print(f"p-value: {result.pvalue}")
    print(f'Относительное изменение средних = {round(sample_2.mean() / sample_1.
```

In [7]:

```
logs_exp = pd.read_csv('logs_exp.csv', sep='\t')
logs_exp.head(5)
```

Out[7]:

	EventName	DeviceIDHash	EventTimestamp	ExpId
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248
3	CartScreenAppear	3518123091307005509	1564054127	248
4	PaymentScreenSuccessful	6217807653094995999	1564055322	248

In [8]: `logs_exp.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244126 entries, 0 to 244125
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  -
0   EventName       244126 non-null object
1   DeviceIDHash    244126 non-null int64
2   EventTimestamp  244126 non-null int64
3   ExpId           244126 non-null int64
dtypes: int64(3), object(1)
memory usage: 7.5+ MB
```

In [9]: `logs_exp = logs_exp.rename(columns = {logs_exp.columns[0]: 'event_name', logs_exp.columns[1]: 'device_id_hash', logs_exp.columns[2]: 'event_time', logs_exp.columns[3]: 'exp_id'})`

Out[9]:

	event_name	device_id_hash	event_time	exp_id
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246

В данных обнаружены не соответствие типов поле с датой представлено числовым типом int64, так же не очень удобочитаемые наименования полей верблюдным стилем, а так же в поле с датой сама дата представлена секундами с начала эпохи, такое время сложно воспринимать и делать с ним какие-либо расчёты.

Наименование полей приведено к удобочитаемому формату. Обработка поля содержащего дату обработки предусмотрена в следующем разделе. По выводу info(), что в данных нет пропусков.

Обработка пропусков

In [10]: `# Проверка на наличие явных дубликатов`
`logs_exp.duplicated().sum()`

Out[10]: `np.int64(413)`

In [11]: `# Удаление явных дубликатов`
`logs_exp = logs_exp.drop_duplicates().reset_index(drop=True)`

In [12]: `# Повторная проверка на наличие явных дубликатов`
`logs_exp.duplicated().sum()`

```
Out[12]: np.int64(0)
```

```
In [13]: logs_exp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243713 entries, 0 to 243712
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  -
0   event_name      243713 non-null object
1   device_ID_hash  243713 non-null int64
2   event_time      243713 non-null int64
3   exp_id          243713 non-null int64
dtypes: int64(3), object(1)
memory usage: 7.4+ MB
```

Удаление явных дубликатов прошло успешно. Доля удаленных данных составила менее 0.2 % от всего числа данных.

Изменение таблицы, добавление необходимых полей.

```
In [14]: # Переводим формат времени эпохи unix в удобочитаемый формат "year-month-day time"
logs_exp['event_time'] = pd.to_datetime(logs_exp['event_time'], unit = 's')
logs_exp.head(5)
```

```
Out[14]:
```

	event_name	device_ID_hash	event_time	exp_id
0	MainScreenAppear	4575588528974610257	2019-07-25 04:43:36	246
1	MainScreenAppear	7416695313311560658	2019-07-25 11:11:42	246
2	PaymentScreenSuccessful	3518123091307005509	2019-07-25 11:28:47	248
3	CartScreenAppear	3518123091307005509	2019-07-25 11:28:47	248
4	PaymentScreenSuccessful	6217807653094995999	2019-07-25 11:48:42	248

```
In [15]: logs_exp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243713 entries, 0 to 243712
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  -
0   event_name      243713 non-null object
1   device_ID_hash  243713 non-null int64
2   event_time      243713 non-null datetime64[ns]
3   exp_id          243713 non-null int64
dtypes: datetime64[ns](1), int64(2), object(1)
memory usage: 7.4+ MB
```

```
In [16]: # Добавим поле "date", в котором будет содержаться только дата события
logs_exp['date'] = logs_exp['event_time'].dt.normalize()
logs_exp.head(2)
```

```
Out[16]:
```

	event_name	device_ID_hash	event_time	exp_id	date
0	MainScreenAppear	4575588528974610257	2019-07-25 04:43:36	246	2019-07-25
1	MainScreenAppear	7416695313311560658	2019-07-25 11:11:42	246	2019-07-25

```
In [17]: logs_exp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243713 entries, 0 to 243712
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   event_name      243713 non-null object
1   device_ID_hash  243713 non-null int64
2   event_time      243713 non-null datetime64[ns]
3   exp_id          243713 non-null int64
4   date            243713 non-null datetime64[ns]
dtypes: datetime64[ns](2), int64(2), object(1)
memory usage: 9.3+ MB
```

```
In [18]: sum(logs_exp.groupby('exp_id')['device_ID_hash'].nunique().values) - (logs_exp['
```

```
Out[18]: np.int64(0)
```

Для быстрой проверки наличия пользователей, которые оказались в обеих группах, я собрала все уникальные идентификаторы по группам и вычла из них общее количество уникальных идентификаторов в датафрейме. Если бы результат вычислений оказался положительным, это свидетельствовало бы о наличии повторяющихся пользователей, поскольку пользователи с идентичным device_ID_hash могли быть зафиксированы в разных группах и, соответственно, учитываться несколько раз. Однако в случае, когда итоговая сумма равна нулю, это указывает на то, что одни и те же пользователи не пересекались в различных группах. Такой результат подтверждает отсутствие дублирующихся записей, что позволяет с уверенностью утверждать о четком разделении пользователей между исследуемыми группами.

Все данные находятся в нужном формате. Добавлены необходимые поля для дальнейшей работы. Так же проведена предварительная обработка данных удалены дубликаты.

Пропусков в данных нет (видно по выводам метода info т.к. совпадает общее количество строк и количество строк в каждом поле таблицы)

Изучение и проверка данных

Количественное изучение данных

```
In [19]: print('В данном датасете всего:\n\t', logs_exp['event_name'].nunique(), 'событий')
print('Такие как: ')
for log_name in logs_exp['event_name'].unique():
```

```
print('\t', log_name)
logs_exp
```

В данном датасете всего:
5 событий

Такие как:

```
MainScreenAppear
PaymentScreenSuccessful
CartScreenAppear
OffersScreenAppear
Tutorial
```

Out[19]:

	event_name	device_ID_hash	event_time	exp_id	date
0	MainScreenAppear	4575588528974610257	2019-07-25 04:43:36	246	2019-07-25
1	MainScreenAppear	7416695313311560658	2019-07-25 11:11:42	246	2019-07-25
2	PaymentScreenSuccessful	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25
3	CartScreenAppear	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25
4	PaymentScreenSuccessful	6217807653094995999	2019-07-25 11:48:42	248	2019-07-25
...
243708	MainScreenAppear	4599628364049201812	2019-08-07 21:12:25	247	2019-08-07
243709	MainScreenAppear	5849806612437486590	2019-08-07 21:13:59	246	2019-08-07
243710	MainScreenAppear	5746969938801999050	2019-08-07 21:14:43	246	2019-08-07
243711	MainScreenAppear	5746969938801999050	2019-08-07 21:14:58	246	2019-08-07
243712	OffersScreenAppear	5746969938801999050	2019-08-07 21:15:17	246	2019-08-07

243713 rows × 5 columns

In [20]: `print('В данном датасете всего:', logs_exp['device_ID_hash'].nunique(), 'пользов`

В данном датасете всего: 7551 пользователь

In [21]: `print('Распределение уникальных пользователей по группам тестирования:\n', logs_e`

Распределение уникальных пользователей по группам тестирования:

exp_id

246 2489

247 2520

248 2542

Name: device_ID_hash, dtype: int64

In [22]: `print('В среднем на каждого пользователя приходится:', round(len(logs_exp) / log`

В среднем на каждого пользователя приходится: 32 события

Изучение временных рамок исследования

```
In [23]: print('Самой поздней датой является:', max(logs_exp['date']).date())  
print('Самой ранней датой является:', min(logs_exp['date']).date())
```

Самой поздней датой является: 2019-08-07

Самой ранней датой является: 2019-07-25

```
In [24]: count_logs_of_date = logs_exp.groupby('date')['event_name'].count().reset_index(  
grouped_df = logs_exp.groupby(['date', 'exp_id'])['event_name'].count().unstack(  
grouped_df.index = grouped_df.index.strftime('%Y-%m-%d')  
grouped_df
```

Out[24]:

exp_id	246	247	248
--------	-----	-----	-----

date			
2019-07-25	4	1	4
2019-07-26	14	8	9
2019-07-27	24	23	8
2019-07-28	33	36	36
2019-07-29	55	58	71
2019-07-30	129	138	145
2019-07-31	620	664	746
2019-08-01	11561	12306	12274
2019-08-02	10946	10990	13618
2019-08-03	10575	11024	11683
2019-08-04	11514	9942	11512
2019-08-05	12368	10949	12741
2019-08-06	11726	11720	12342
2019-08-07	10612	10091	10393

```
In [25]: # Визуализация  
fig, ax = plt.subplots(figsize=(12, 6))  
grouped_df.plot(kind='bar', ax=ax, width=0.8)  
  
plt.title('Количество событий по времени и группам')  
plt.xlabel('Дата')  
plt.ylabel('Сумма событий')  
plt.xticks(rotation=45)  
plt.legend(title='Группа')  
plt.grid()  
plt.tight_layout()  
plt.show()
```




Как видно из графика, количество зарегистрированных событий до 31 июля оставалось незначительным. Однако начиная с 1 августа наблюдается резкий, многократный рост активности в приложении.

На столбчатой диаграмме, иллюстрирующей количество событий в зависимости от времени и групп, заметно, что в целом объем событий повторяется, без резких различий между группами. Часто выявляется преобладание событий в экспериментальной группе, в то время как контрольные группы оказываются в меньшинстве. Однако разрыв в данных не является значительным. Такое положение дел также может быть связано с тем, что в экспериментальной группе представлено немного больше пользователей по сравнению с контрольными группами. Сравнительный анализ не выявляет ощутимых контрастов, указывая на стабильность результатов в различных условиях. В целом, данная визуализация помогает углубить понимание динамики событий и их зависимости от экспериментальных условий, подчеркивая, что несмотря на небольшие колебания, общий тренд остается неизменным, что свидетельствует о корректности проведенного анализа.

```
In [26]: # Суммируем значения в столбце 'events' для отфильтрованных строк
total_events_before = count_logs_of_date[count_logs_of_date['date'] <= pd.to_datetime('2019-07-31')]
print(f'Число событий до взлета показателя после 2019-07-31: {total_events_before.sum()}')

total_events_after = count_logs_of_date[count_logs_of_date['date'] > pd.to_datetime('2019-08-01')]
print(f'Число событий после взлета показателя 2019-08-01: {total_events_after.sum()}')

print(f'Число событий до взлёта составляет {round((total_events_before.sum() / (total_events_before.sum() + total_events_after.sum()) * 100, 2)} % от всего числа данных')
```

Число событий до взлета показателя после 2019-07-31: 2826

Число событий после взлета показателя 2019-08-01: 240887

Число событий до взлёта составляет 1.16 % от всего числа данных

Данные логи сильно выбиваются из общей картины и их значения ниже в несколько десятков раз в последующем анализе исказят информацию, считая необходимым данные до 31 июля включительно удалить и использовать начальную

дату 1 августа. Данными за период с 1 августа по 7 августа мы располагаем на самом деле

```
In [27]: size_initial_table = len(logs_exp)
logs_exp = logs_exp[logs_exp['date'] >= pd.to_datetime('2019-08-01')].reset_index(drop=True)
```

```
Out[27]:
```

	event_name	device_ID_hash	event_time	exp_id	date
0	Tutorial	3737462046622621720	2019-08-01 00:07:28	246	2019-08-01
1	MainScreenAppear	3737462046622621720	2019-08-01 00:08:00	246	2019-08-01
2	MainScreenAppear	3737462046622621720	2019-08-01 00:08:55	246	2019-08-01
3	OffersScreenAppear	3737462046622621720	2019-08-01 00:08:58	246	2019-08-01
4	MainScreenAppear	1433840883824088890	2019-08-01 00:08:59	247	2019-08-01
...
240882	MainScreenAppear	4599628364049201812	2019-08-07 21:12:25	247	2019-08-07
240883	MainScreenAppear	5849806612437486590	2019-08-07 21:13:59	246	2019-08-07
240884	MainScreenAppear	5746969938801999050	2019-08-07 21:14:43	246	2019-08-07
240885	MainScreenAppear	5746969938801999050	2019-08-07 21:14:58	246	2019-08-07
240886	OffersScreenAppear	5746969938801999050	2019-08-07 21:15:17	246	2019-08-07

240887 rows × 5 columns

Проверка процента сохраненных данных и состава экспериментальных групп

```
In [28]: print('Удалось сохранить', round(len(logs_exp) / size_initial_table * 100, 1), '%', 'Удалось сохранить 98.8 % от исходной информации')
```

```
In [29]: logs_exp.groupby('exp_id')['device_ID_hash'].nunique()
```

```
Out[29]: exp_id
246      2484
247      2513
248      2537
Name: device_ID_hash, dtype: int64
```

По итогам удаления выбросов сохранились все 3 экспериментальные группы, количество уникальных пользователей по группам сократилось не значительно.

Изучение воронки событий

Изучение логов событий

```
In [30]: logs_counter = logs_exp.groupby('event_name')['device_ID_hash'].count().sort_val
logs_counter
```

```
Out[30]:
```

	event_name	device_ID_hash
0	Tutorial	1005
1	PaymentScreenSuccessful	33918
2	CartScreenAppear	42303
3	OffersScreenAppear	46333
4	MainScreenAppear	117328

Формирование воронки

Подсчёт количества пользователей совершили каждое из этих событий.

```
In [31]: logs_by_user = logs_exp.groupby(['device_ID_hash', 'event_name'])['event_name'].
logs_by_user
```

```
Out[31]:
```

device_ID_hash	event_name	
33381960957324102	CartScreenAppear	1
	PaymentScreenSuccessful	1
6909561520679493	CartScreenAppear	1
	OffersScreenAppear	1
	PaymentScreenSuccessful	1
...		
4623191541214045580	CartScreenAppear	778
197027893265565660	PaymentScreenSuccessful	865
	CartScreenAppear	931
6304868067479728361	PaymentScreenSuccessful	1085
	CartScreenAppear	1100

Name: event_name, Length: 20125, dtype: int64

Изучение руководства в воронку не входит, так как пользователь может совершать покупку и другие действия из воронки вне зависимости от того, прочёл ли он руководство. Уберем данное действие из последующего анализа.

```
In [32]: logs_exp = logs_exp[logs_exp.event_name != 'Tutorial']
```

```
In [33]: # Посчитать количество пользователей, совершивших каждое событие контрольные гру
users_per_event = logs_exp.groupby('event_name')['device_ID_hash'].nunique().sor
print("События, отсортированные по числу пользователей:")
print(users_per_event)

# Посчитать долю пользователей, совершивших каждое событие
```

```
proportion_per_event = round(users_per_event / logs_exp['device_ID_hash'].nunique)

print("\nДоля пользователей, совершивших каждое событие:")
print(proportion_per_event)
```

События, отсортированные по числу пользователей:

```
event_name
MainScreenAppear          7419
OffersScreenAppear        4593
CartScreenAppear          3734
PaymentScreenSuccessful    3539
Name: device_ID_hash, dtype: int64
```

Доля пользователей, совершивших каждое событие:

```
event_name
MainScreenAppear          0.99
OffersScreenAppear        0.61
CartScreenAppear          0.50
PaymentScreenSuccessful    0.47
Name: device_ID_hash, dtype: float64
```

По долям пользователей совершивших каждое действие видим, что почти все (99%) открыли главное меню (MainScreenAppear) и почти половина дошла до успешной оплаты 47 %, но на событии OffersScreenAppear имеет гораздо ниже долю от MainScreenAppear это может говорить о проблемах в привлечении пользователей к разделу с предложениями или о том, что этот раздел не так удобен, на данном этапе теряется почти 38 % пользователей. Дальнейшие потери не так велики и к корзине переходят на 81 % пользователей после OffersScreenAppear, а оплачивают и вовсе 94% (47 % от 50 %) зашедших в корзину (CartScreenAppear). Событие Tutorial имеет низкую популярность, возможно, пользователям не требуется обучающий материал или он не так удобен. Конверсию подробнее рассмотрим в следующем разделе.

Выявление плана порядка событий.

```
In [34]: logs_counter.sort_values(by='device_ID_hash', ascending=False)
```

```
Out[34]:
```

	event_name	device_ID_hash
4	MainScreenAppear	117328
3	OffersScreenAppear	46333
2	CartScreenAppear	42303
1	PaymentScreenSuccessful	33918
0	Tutorial	1005

```
In [35]: round(logs_counter.loc[0, 'device_ID_hash'] / logs_counter['device_ID_hash'].sum
```

```
Out[35]: np.float64(0.4)
```

В данном наборе данных наибольшую популярность занимает действие "ОТКРЫТЬ ГЛАВНЫЙ ЭКРАН" (MainScreenAppear), которое зафиксировано 117,328 раз. Самым редким событием выступает "РУКОВОДСТВО" (Tutorial) 1005 событий. Из таблицы

распределения частоты событий вырисовывается определенная логика последовательности действий пользователей. После открытия главной страницы, по убывающей частоте, пользователи перемещаются на страницу с предложениями, затем переходят в корзину, заполняя ее товарами, и завершающим действием становится успешная оплата. Эти четыре события образуют четкую логическую цепь.

**Главный экран → Экран предложений → Экран корзины → Оплата
произведена успешно**

**MainScreenAppear → OffersScreenAppear → CartScreenAppear →
PaymentScreenSuccessful**

Тем не менее, порой пользователи прибегают к "РУКОВОДСТВУ", что, возможно, свидетельствует о возникших трудностях на одном из этапов. На эту страницу приходится лишь 0.4% от общего числа событий, что подчеркивает редкость этого действия.

Подсчет долей пользователей проходящие через воронку.

```
In [36]: for ind, count in enumerate(proportion_per_event):
          if ind == len(proportion_per_event.values) - 1:
              print(f'\nОт первого {proportion_per_event.index[0]} действия к последнему
                  break
          else:
              print(f'После {proportion_per_event.index[ind]} действия к следующему {pr
```

После MainScreenAppear действия к следующему OffersScreenAppear переходит 62 % пользователей

После OffersScreenAppear действия к следующему CartScreenAppear переходит 82 % пользователей

После CartScreenAppear действия к следующему PaymentScreenSuccessful переходит 94 % пользователей

От первого MainScreenAppear действия к последнему PaymentScreenSuccessful доходит 47 % пользователей

```
In [37]: proportion_per_event = proportion_per_event.reset_index()
```

```
In [38]: proportion_per_event
```

```
Out[38]:
```

	event_name	device_ID_hash
0	MainScreenAppear	0.99
1	OffersScreenAppear	0.61
2	CartScreenAppear	0.50
3	PaymentScreenSuccessful	0.47

```
In [39]: proportion_per_event['conversion'] = round(proportion_per_event['device_ID_hash'
```

```
In [40]: proportion_per_event
```

```
Out[40]:
```

	event_name	device_ID_hash	conversion
0	MainScreenAppear	0.99	NaN
1	OffersScreenAppear	0.61	0.62
2	CartScreenAppear	0.50	0.82
3	PaymentScreenSuccessful	0.47	0.94

Изучение результатов эксперимента

Контрольные проверки

```
In [41]: logs_exp.head(15)
```

Out[41]:

	event_name	device_ID_hash	event_time	exp_id	date
1	MainScreenAppear	3737462046622621720	2019-08-01 00:08:00	246	2019-08-01
2	MainScreenAppear	3737462046622621720	2019-08-01 00:08:55	246	2019-08-01
3	OffersScreenAppear	3737462046622621720	2019-08-01 00:08:58	246	2019-08-01
4	MainScreenAppear	1433840883824088890	2019-08-01 00:08:59	247	2019-08-01
5	MainScreenAppear	4899590676214355127	2019-08-01 00:10:15	247	2019-08-01
6	OffersScreenAppear	3737462046622621720	2019-08-01 00:10:26	246	2019-08-01
7	MainScreenAppear	3737462046622621720	2019-08-01 00:10:47	246	2019-08-01
8	MainScreenAppear	3737462046622621720	2019-08-01 00:11:10	246	2019-08-01
9	MainScreenAppear	3737462046622621720	2019-08-01 00:11:20	246	2019-08-01
10	MainScreenAppear	4899590676214355127	2019-08-01 00:11:28	247	2019-08-01
11	OffersScreenAppear	4899590676214355127	2019-08-01 00:11:30	247	2019-08-01
12	MainScreenAppear	1182179323890311443	2019-08-01 00:11:57	246	2019-08-01
14	MainScreenAppear	3737462046622621720	2019-08-01 00:12:34	246	2019-08-01
15	OffersScreenAppear	4899590676214355127	2019-08-01 00:12:36	247	2019-08-01
16	MainScreenAppear	4613461174774205834	2019-08-01 00:14:31	248	2019-08-01

Подсчёт количества пользователей в каждой экспериментальной группе

```
In [42]: logs_exp.groupby('exp_id')['device_ID_hash'].nunique()  
logs_exp.groupby('exp_id')['device_ID_hash'].nunique().values[0]
```

Out[42]: np.int64(2483)

В данном исследовании я буду использовать уровень значимости $\alpha = 0.05$. Это достаточно строгий уровень значимости, который дает хороший баланс между вероятностью обнаружить статистически значимые различия (если они есть) и вероятностью ошибочно отвергнуть верную нулевую гипотезу. В нашем случае с

большим объемом выборки (суммарно 7551) использование уровня значимости 0.1 может привести к увеличению количества ложноположительных результатов.

Нулевая гипотеза (H0): Распределения различий в среднем количестве событий на посетителя по данным в контрольных группах 246 и 247 не отличаются. Это означает, что нет статистически значимой разницы между группами по этой метрике.

Альтернативная гипотеза (H1) Распределения значений различий в среднем количестве событий на посетителя по данным в контрольных группах 246 и 247 отличаются. Это означает, что существует статистически значимая разница между группами по этой метрике.

```
In [43]: x = logs_exp.groupby('exp_id')['device_ID_hash'].nunique().values[0]
y = logs_exp.groupby('exp_id')['device_ID_hash'].nunique().values[1]
z = int(x + y)
alpha = 0.05
purchases = np.array([x, y])
leads = np.array([z, z])

p1 = purchases[0] / leads[0]
p2 = purchases[1] / leads[1]

p_combined = (purchases[0] + purchases[1]) / (leads[0] + leads[1])
difference = p1 - p2

z_value = difference / math.sqrt(p_combined * (1 - p_combined) * (1 / leads[0] +
distr = stats.norm(0, 1)
p_value = (1 - distr.cdf(abs(z_value))) * 2

print('p-значение: ', p_value)

if p_value < alpha:
    print('Отвергаем нулевую гипотезу: между долями есть значимая разница\n')
else:
    print('Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли
```

p-значение: 0.5617189231075914

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Результаты теста на контрольных группах не показали статистически значимой разницы между группами, что говорит о соблюдении технологии деления на группы.

Проверка контрольных групп для А/А-эксперимента 246 и 247 показала отсутствие статистически значимых различий в распределении количества событий на пользователя между двумя экспериментальными группами, это показывает проверку гипотезы о равенстве долей.

**Статистически значимых различий между выборками установить не удалось.
Нулевую гипотезу не отвергаем**

Определение самого популярного события для контрольных групп

```
In [44]: control_groups = logs_exp[logs_exp['exp_id'] != 248].groupby('event_name')['device_ID_hash'].nunique().sort_values(ascending=False)
```

```
Out[44]:
```

	event_name	device_ID_hash
0	MainScreenAppear	76766
1	OffersScreenAppear	29946
2	CartScreenAppear	27124
3	PaymentScreenSuccessful	21833

Самым популярным событием для контрольных групп становится MainScreenAppear "ПРОСМОТР ГЛАВНОЙ СТРАНИЦЫ"

```
In [45]: event_246, fraction_246, num_246 = users_per_event_group(246, logs_exp)
print("События, отсортированные по числу пользователей:")
print(event_246)
print("\nДоля пользователей, совершивших каждое событие:")
print(fraction_246)
```

События, отсортированные по числу пользователей:

```
event_name
MainScreenAppear      2450
OffersScreenAppear    1542
CartScreenAppear      1266
PaymentScreenSuccessful 1200
Name: device_ID_hash, dtype: int64
```

Доля пользователей, совершивших каждое событие:

```
event_name
MainScreenAppear      0.99
OffersScreenAppear    0.62
CartScreenAppear      0.51
PaymentScreenSuccessful 0.48
Name: device_ID_hash, dtype: float64
```

```
In [46]: event_247, fraction_247, num_247 = users_per_event_group(247, logs_exp)
print("События, отсортированные по числу пользователей:")
print(event_247)
print("\nДоля пользователей, совершивших каждое событие:")
print(fraction_247)
```

События, отсортированные по числу пользователей:

```
event_name
MainScreenAppear      2476
OffersScreenAppear    1520
CartScreenAppear      1238
PaymentScreenSuccessful 1158
Name: device_ID_hash, dtype: int64
```

Доля пользователей, совершивших каждое событие:

```
event_name
MainScreenAppear      0.99
OffersScreenAppear    0.61
CartScreenAppear      0.49
PaymentScreenSuccessful 0.46
Name: device_ID_hash, dtype: float64
```

Можно сказать, что разбиение на группы работает корректно так как для всех действий разбитым по группам размер и процент доли схож. Найти сильных отличий в данных не получилось поэтому можно с уверенностью утверждать корректность деления на группы.

Нулевая гипотеза (H0): Распределения различий в среднем количестве распределений событий в контрольных группах 246 и 247 не отличаются. Это означает, что нет статистически значимой разницы между группами по этой метрике.

Альтернативная гипотеза (H1): Распределения значений различий в среднем количестве распределений событий в контрольных группах 246 и 247 отличаются. Это означает, что существует статистически значимая разница между группами по этой метрике.

```
In [47]: determination_stat_sign(event_246, event_247, num_246, num_247)
```

Для события MainScreenAppear
p-значение: 0.7526703436483038
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события OffersScreenAppear
p-значение: 0.24786096925282264
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события CartScreenAppear
p-значение: 0.22867643757335676
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события PaymentScreenSuccessful
p-значение: 0.11446627829276612
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Нулевую гипотезу отвергнуть нельзя. Гипотеза о равенстве пропорций событий в этих двух сегментах верна.

Определение самого популярного события для экспериментальной групп

```
In [48]: event_248, fraction_248, num_248 = users_per_event_group(248, logs_exp)
print("События, отсортированные по числу пользователей:")
print(event_248)
print("\nДоля пользователей, совершивших каждое событие:")
print(fraction_248)
```

События, отсортированные по числу пользователей:

```
event_name
MainScreenAppear      2493
OffersScreenAppear    1531
CartScreenAppear      1230
PaymentScreenSuccessful 1181
Name: device_ID_hash, dtype: int64
```

Доля пользователей, совершивших каждое событие:

```
event_name
MainScreenAppear      0.98
OffersScreenAppear    0.60
CartScreenAppear      0.49
PaymentScreenSuccessful 0.47
Name: device_ID_hash, dtype: float64
```

Для экспериментальной группы самым популярным действием так же становится MainScreenAppear.

Сравнение контрольных и экспериментальной групп

```
In [49]: determination_stat_sign(event_246, event_248, num_246, num_248)
```

Для события MainScreenAppear

p-значение: 0.3387114076159288

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события OffersScreenAppear

p-значение: 0.21442476639710506

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события CartScreenAppear

p-значение: 0.08067367598823139

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события PaymentScreenSuccessful

p-значение: 0.21693033984516674

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Не отвергаем нулевую гипотезу для всех действий кроме MainScreenAppear, как видим по результатам t-теста есть статистически значимые отличия при p-value = 0.05 можем отвергнуть нулевую гипотезу. Можно предположить что изменение шрифтов влияет на изменение количества посещения главной страницы.

Для остальных действий выявить статистически значимых событий выявить не удалось. Нулевую гипотезу для них не отвергаем.

```
In [50]: determination_stat_sign(event_247, event_248, num_247, num_248)
```

Для события mainScreenAppear
p-значение: 0.5194964354051703

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события OffersScreenAppear
p-значение: 0.9333751305879443

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события CartScreenAppear
p-значение: 0.5878284605111943

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события PaymentScreenSuccessful
p-значение: 0.7275718682261119

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

После проверки статистической значимости достоверных отличий выявить не удалось. Относительное изменение средних колеблется в пределах от 3 % до 22,8 %.

Нулевую гипотезу не отвергаем для всех событий.

```
In [51]: event_united, num_united = users_per_event_group_2(248, logs_exp)
num_united
```

Out[51]: 4995

```
In [52]: determination_stat_sign(event_united, event_248, num_united, num_248)
```

Для события mainScreenAppear
p-значение: 0.3486684291093256

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события OffersScreenAppear
p-значение: 0.44582745409482394

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события CartScreenAppear
p-значение: 0.18683558686831558

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Для события PaymentScreenSuccessful
p-значение: 0.6107918742187335

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

Всего провели 16 тестов. При уровне значимости 0.05, в данном случае необходимо применить провести коррекцию уровня значимости. Один из вариантов это сделать - поправка Бонферрони, что бы вероятность совершить ошибку первого рода была минимальна.

Скорректированный уровень значимости = $0.05 / 16 = 0.003125$ При уровне значимости 0.003125 не было выявлено статистически значимой разницы в доле пользователей, совершивших одно из исследуемых действий.

Аргументация выбранной стратегии, подсчёт количества статистических проверок гипотез.

В данном исследовании я буду использовать уровень значимости $\alpha = 0.05$. Это достаточно строгий уровень значимости, который дает хороший баланс между вероятностью обнаружить статистически значимые различия (если они есть) и вероятностью ошибочно отвергнуть верную нулевую гипотезу. В нашем случае с большим объемом выборки (суммарно 7551) использование уровня значимости 0.1 может привести к увеличению количества ложноположительных результатов.

Таким образом, можно с уверенностью утверждать, что смена шрифта на новый не оказала никакого влияния на воронку продаж, что, в свою очередь, свидетельствует о том, что количество продаж останется неизменным.

In []: