Анализ пользовательского поведения в мобильном приложении (анализ воронки продаж и результатов A/A/B теста)

Заказчик: отдел маркетинга стартапа, продающий продукты питания через мобильное приложение.

Цель исследования: 1) разобраться, как ведут себя пользователи, изучив воронку продаж;

2) исследовать результаты A/A/B-эксперимента о влиянии нового шрифта на продажи(пользователей разбили на 3 группы: 2 контрольные со старыми шрифтами и одну экспериментальную — с новыми)

Входные данные: логи с действиями пользователей

Описание данных:

- EventName — название события;
- DeviceIDHash — уникальный идентификатор пользователя;
- EventTimestamp — время события;
- ExpId — номер эксперимента: 246 и 247 — контрольные группы, а 248 — экспериментальная.

Содержание проекта

Шаг 1. Загрузка данных

In [5]:
```
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats as st # второй способ написания import scipy.stats as stats
import numpy as np
import math as mth
import plotly.express as px # для графика воронки продаж
```

Out[6]:

	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

```
<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
```

Шаг 2. Подготовка данных

In [8]:
```
#переименуем столбцы
```

```
df = df.rename(columns={'EventName':'event_name','DeviceIDHash': 'user_id', 'EventTimestamp':'date_time', 'ExpId'
df.head(3)
```

Out[8]:

	event_name	user_id	date_time	group
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248

In [9]:

```
# приведём значение времени в столбце date к стандартному виду
df.date_time = pd.to_datetime(df.date_time, unit='s')
df.head(3)
```

Out[9]:

	event_name	user_id	date_time	group
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

In [10]:

```
# добавим столбец с датой
df['date'] = df['date_time'].dt.date
df.head(3)
```

Out[10]:

	event_name	user_id	date_time	group	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

```
Пропусков в столбце event_name: 0
Пропусков в столбце user_id: 0
Пропусков в столбце date_time: 0
Пропусков в столбце group: 0
Пропусков в столбце date: 0
```

In [13]:

```
# проверим данные на дубликаты:
df.duplicated().sum()
```

Out[13]:

```
413
```

413 составляют незначительную часть от 244126 записей таблицы, поэтому можем их удалить:

In [14]:

```
df = df.drop_duplicates().reset_index(drop=True)
```

Шаг 3. Изучение и проверка данных

In [15]:

```
# посчитаем количество событий
cnt_event = df['event_name'].count()
cnt_event
```

Out[15]:

```
243713
```

In [16]:

```
# посчитаем количество уникальных пользователей
cnt_user = df['user_id'].nunique()
cnt_user
```

Out[16]:

```
7551
```

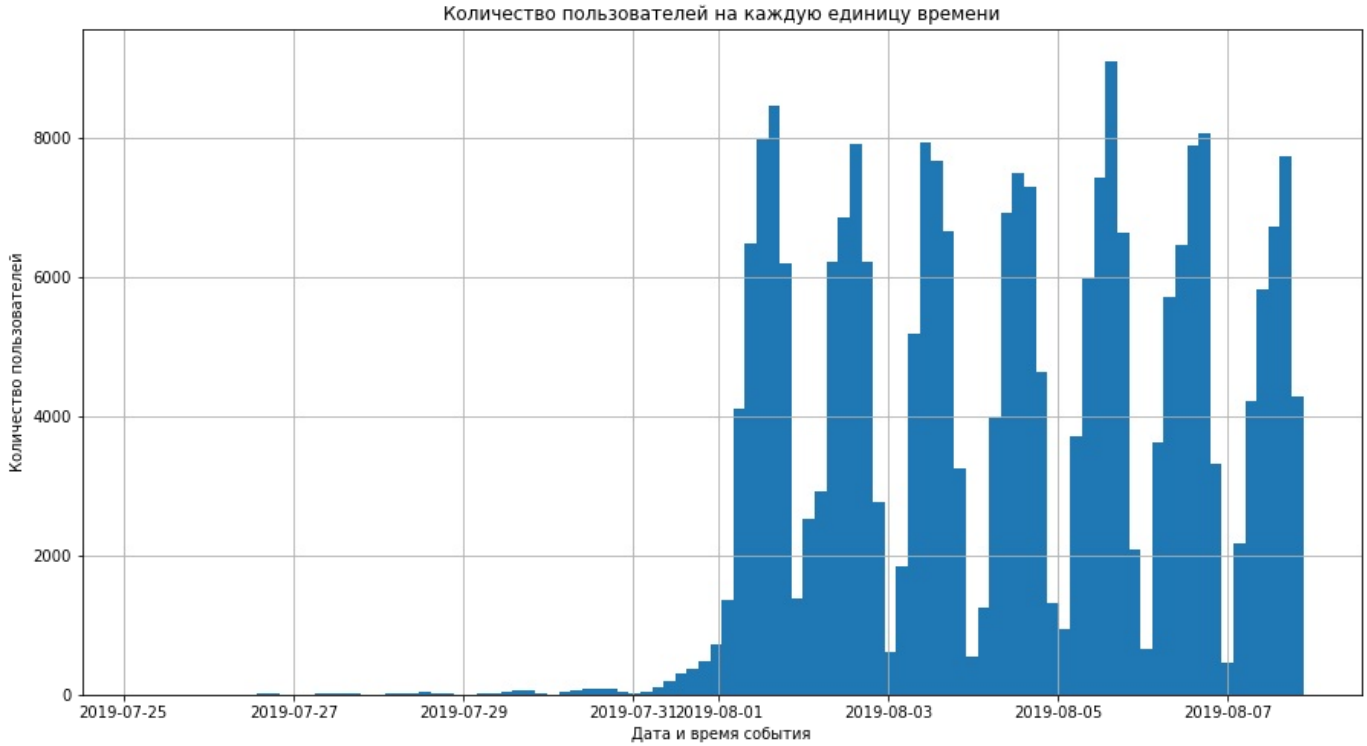
In [17]:
```
# посчитаем, сколько в среднем событий приходится на пользователя
ev_user = round(cnt_event/cnt_user, 1)
ev_user
```

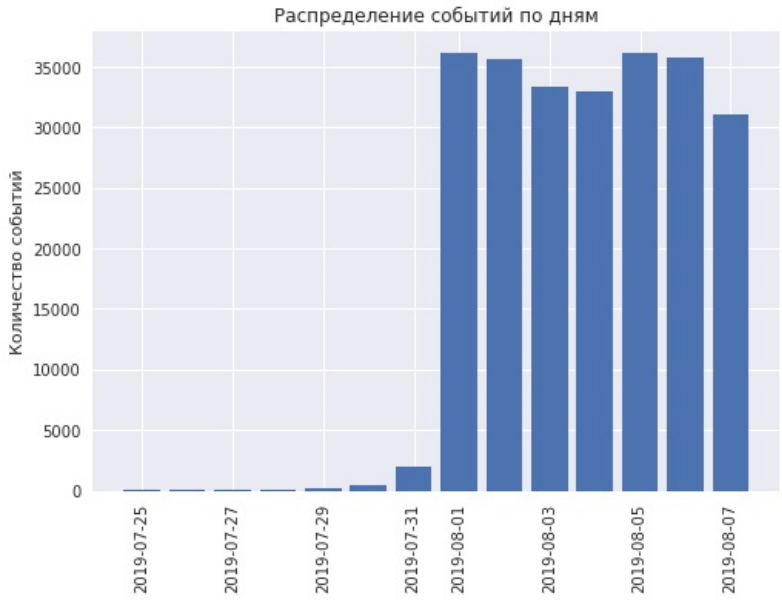
Out[17]:
```
32.3
```

In [18]:
```
#дайдём минимальную и максимальную дату записи в таблицу:

print('Минимальная дата записи в таблицу:',df['date'].min())
print('Максимальная дата записи в таблицу:',df['date'].max())
```

```
Минимальная дата записи в таблицу: 2019-07-25
Максимальная дата записи в таблицу: 2019-08-07
```





По графику видим, что до 2019-07-31 событий почти не было из-за малого кол-ва пользователей. Отбросим временной период до этой даты, а также учтём то, что в логи новых дней по некоторым пользователям могут «доезжать» события из прошлого — это может «перекашивать данные» и отбросим время до 02:00 дня 2019-07-31, чтобы не попали те пользователи, которые начали что-то делать поздно вечером предыдущего дня.

Таким образом, располагаем данными за период с 2019-07-31 (не полный день) по 2019-08-07.

In [21]:
```
# обросим период, за который данные были не полные
data = df[df['date_time'] > '2019-07-31 02:00:00']
```

In [22]:
```
df[df['date_time'] > '2019-07-31 02:00:00']
```

Out[22]:

	event_name	user_id	date_time	group	date
799	MainScreenAppear	4293428370257583636	2019-07-31 02:19:18	248	2019-07-31
800	MainScreenAppear	4567464647598975872	2019-07-31 02:33:30	247	2019-07-31
801	MainScreenAppear	416669255233170069	2019-07-31 03:07:07	247	2019-07-31
802	MainScreenAppear	6983610287457587320	2019-07-31 03:15:43	246	2019-07-31
803	MainScreenAppear	6983610287457587320	2019-07-31 03:16:09	246	2019-07-31
...	...	...	...	...	...
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

242914 rows × 5 columns

In [23]:
```
len(df)
```

Out[23]:
```
243713
```

In [24]:
```
# определим, какой % данных был отфильтрован:
print('Было отфильтровано данных,%:', (round(100 -len(data)/len(df)*100,1)))
```

```
Было отфильтровано данных,%: 0.3
```

In [25]:
```
data.groupby(['group']).agg({'date_time':'count'}).reset_index().rename(columns ={'date_time': 'number'})
```

Out[25]:

	group	number
0	246	79922
1	247	77684
2	248	85308

По результатам расчётов видим, что после фильтрации есть пользователи из всех трёх экспериментальных групп.

Шаг 4. Изучение воронки событий

In [26]:
```
# Посмотрим, какие события есть в логах, как часто они встречаются.Отсортируем события по частоте.
events = data.groupby('event_name').agg({'group':'count'}).reset_index()
events = events.rename(columns = {'group':'number'}).sort_values(by='number', ascending = False)
events
```

Out[26]:

	event_name	number
1	MainScreenAppear	118577
2	OffersScreenAppear	46706
0	CartScreenAppear	42560
3	PaymentScreenSuccessful	34058
4	Tutorial	1013

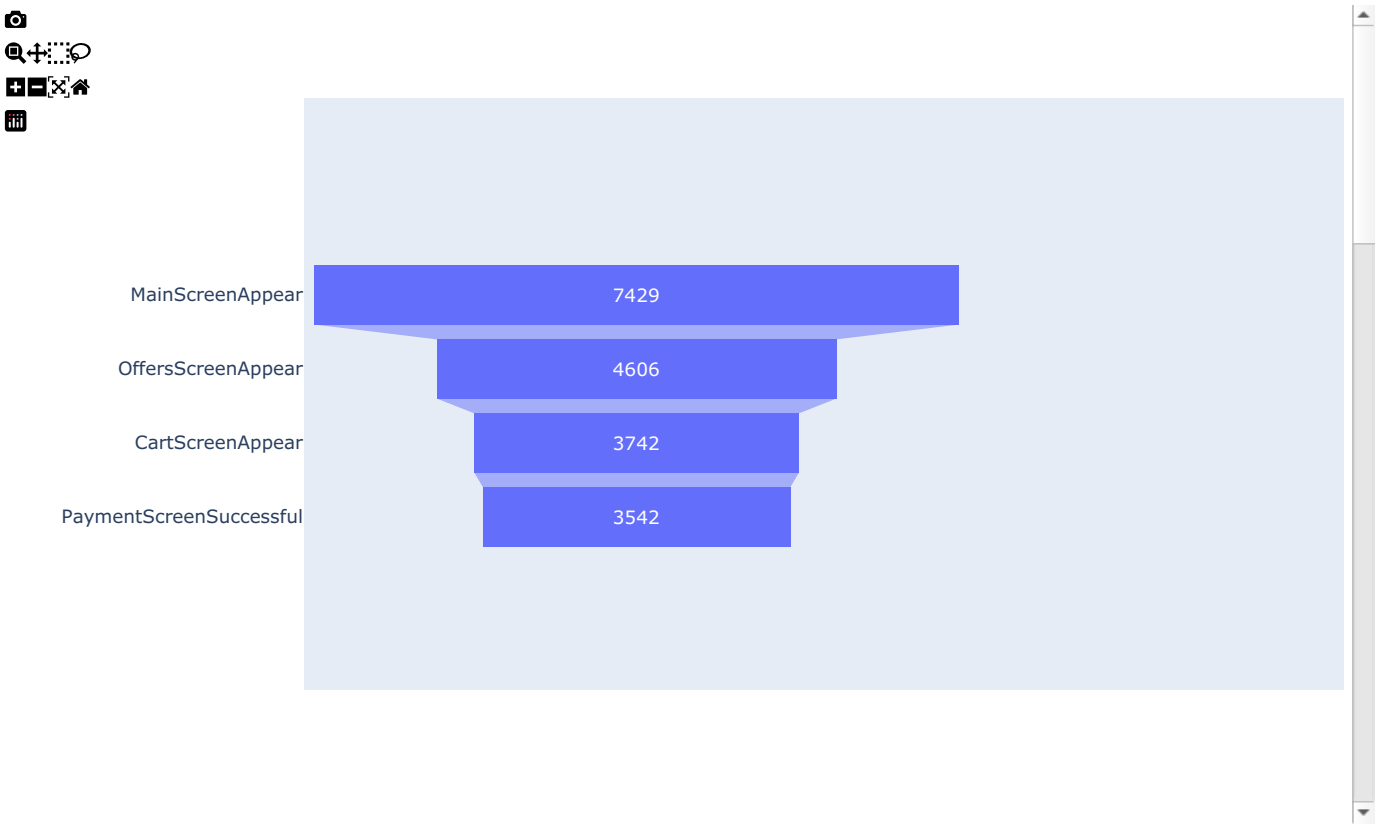
In [27]:

In [27]:
```
# Посчитаем,сколько пользователей совершали каждое из этих событий. События отсортируем по числу пользователей
events_users = data.groupby(['event_name']).agg({'user_id':'nunique' }).reset_index()
events_users  = events_users.rename(columns = {'user_id':'cnt_user' }).sort_values(by='cnt_user', ascending = Fal
events_users
```

Out[27]:

	event_name	cnt_user
1	MainScreenAppear	7429
2	OffersScreenAppear	4606
0	CartScreenAppear	3742
3	PaymentScreenSuccessful	3542
4	Tutorial	845

Отобразим рассчитанные данные по количеству пользователей на графике воронки:



In [29]:
```
# найдём долю (в процентах) пользователей, которые хоть раз совершали событие
events_users['persent,%'] = round(events_users['cnt_user']/cnt_user*100, 1)
events_users
```

Out[29]:

	event_name	cnt_user	persent,%
1	MainScreenAppear	7429	98.4
2	OffersScreenAppear	4606	61.0
0	CartScreenAppear	3742	49.6
3	PaymentScreenSuccessful	3542	46.9
4	Tutorial	845	11.2

После сотрировки событий видим по 4-м первым событиям логичную цепочку событий, которую проходит пользователь при покупке товара:

1. MainScreenAppear = пользователь начинает использование приложения с его главной страницы
2. OffersScreenAppear = переходит на экран с предложениями для выбора товаров
3. CartScreenAppear = переходит в корзину с выбранными товарами
4. PaymentScreenSuccessful = оплачивает свой заказ

Страница с Руководством пользователя в анализе воронки продаж учитывать не будем.

In [30]:

In [30]:
```
# посчитаем, какая доля пользователей переходит на следующий шаг относительно предыдущего:

main = data[data['event_name'] == 'MainScreenAppear']
offers = data[data['event_name'] == 'OffersScreenAppear']
cart = data[data['event_name'] == 'CartScreenAppear']
payment = data[data['event_name'] == 'PaymentScreenSuccessful']


# посчитаем количество уникальных пользователей
cnt_user_main = main['user_id'].nunique()
cnt_user_offers = offers['user_id'].nunique()
cnt_user_cart = cart['user_id'].nunique()
cnt_user_payment = payment['user_id'].nunique()

A_B = round(cnt_user_offers/cnt_user_main*100,1)
B_C = round(cnt_user_cart/cnt_user_offers*100,1)
C_D = round(cnt_user_payment/cnt_user_cart*100,1)

A_D= round(cnt_user_payment/cnt_user_main*100,1)



print('% от кол-ва пользователей перешедших с гл.стр. на выбор товаров::', A_B)
print('% от кол-ва польователей перешедших со стр. выбора в корзину:', B_C)
print('% от кол-ва пользователей перешедших из корзины на стр. оплаты:', C_D)
print()
print('% от кол-ва пользователей дошедших с первого события до оплаты:', A_D)
```

```
% от кол-ва пользователей перешедших с гл.стр. на выбор товаров:: 62.0
% от кол-ва польователей перешедших со стр. выбора в корзину: 81.2
% от кол-ва пользователей перешедших из корзины на стр. оплаты: 94.7

% от кол-ва пользователей дошедших с первого события до оплаты: 47.7
```

По результатам расчётов видим, что на шаге перехода с главной страницы на страницу выбора товаров теряется больше всего пользователей: только 61,7% от первоначального кол-ва зашедших на гл. страницу перешло к выбору товаров.

А также видим, что 47,7 % пользователей проходят все шаги от главного экрана до покупки.

Шаг 5. Изучение результатов эксперимента

Перед началом изуучения результатов проверим следующие моменты:

- сколько у нас групп в АВ-тесте и сколько пользователей в каждой группе;
- даты начала и окончания теста в каждой группе;
- не попадают ли какие-то пользователи в обе группы.

Расчитаем сколько пользователей в каждой из групп, а также даты начала и окончания теста в каждой группе:

Out[31]:

	group	number	min_date
0	246	2485	2019-07-31
1	247	2517	2019-07-31
2	248	2540	2019-07-31

Out[32]:

	group	number	max_date
0	246	2485	2019-08-07
1	247	2517	2019-08-07
2	248	2540	2019-08-07

По результам расчётов видим, что у всех групп дата начала и окончания теста совпадают.

Проверим, есть ли пользователи попавшие в обе группы, для этого создадим массив уникальных пар значений пользователей и групп теста:

In [33]:
```
user_group =  data[['user_id', 'group']].drop_duplicates()
```

Теперь полученный массив проверим на дубликаты идентификторов пользователей:

Out[34]: 0

По результату проверки видим, что пересечений пользователей по группам нет.

Расчитаем, какое количество пользователей совершивших открытие главной страницы: event_name = 'MainScreenAppear' в группе 246 и группе 247

Out[35]:

	group	number	part,%
0	246	2452	33.0
1	247	2479	33.4

Out[36]:

	group	event_name	number	part,%
1	246	MainScreenAppear	2452	33.0
2	246	OffersScreenAppear	1544	20.8
0	246	CartScreenAppear	1267	17.1
3	246	PaymentScreenSuccessful	1200	16.2
4	246	Tutorial	278	3.7

Out[37]:

	group	event_name	number	part,%
1	247	MainScreenAppear	2479	33.4
2	247	OffersScreenAppear	1526	20.5
0	247	CartScreenAppear	1239	16.7
3	247	PaymentScreenSuccessful	1159	15.6
4	247	Tutorial	285	3.8

Для того, чтобы проверить является ли отличие между группами в конверсии статистически значимым, воспользуемся z-критерем Фишера, т.е. будем проверять гипотезу о равенстве долей клиентов перешедших с одного шага на другой на каждом этапе воронки продаж. Сравнение долей групп, а не абсолютных величин позволяет более точно оценить результаты A/A/B теста не зависимо от размеров групп.

Для проверки гипотезы о равенстве долей выберем 5% уровень статистической значимости

Проверим, является ли отличие в группах 246 и 247 статистически значимым:

Проверим, есть ли стистическая разница между группами 246 и 247.

H0 = нет статистически значимого различия групп 246 и 247 по кол-ву пользователей по каждому из действий

H1 = есть статистически значимое различие групп 246 и 247 по кол-ву пользователей по каждому из действий

Событие: MainScreenAppear в группах A1/A2

In [38]:

```
#alpha = .05 # критический уровень статистической значимости первоначальный

alpha = .0125 # критический уровень статистической значимости с поправкой Бонферрони

successes = np.array([2416, 2442]) # кол-во пользователей по событию
trials = np.array([2450, 2480]) # размер групп

# пропорция успехов в первой группе:
p1 = successes[0]/trials[0]
# пропорция успехов во второй группе:
p2 = successes[1]/trials[1]
# пропорция успехов в комбинированном датасете:
p_combined = (successes[0] + successes[1]) / (trials[0] + trials[1])
# разница пропорций в датасетах
difference = p1 - p2
# считаем статистику в ст.отклонениях стандартного нормального распределения
z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1/trials[0] + 1/trials[1]))
# задаем стандартное нормальное распределение (среднее 0, ст.отклонение 1)
```

```
distr = st.norm(0, 1)
p_value = (1 - distr.cdf(abs(z_value))) * 2
print('p-значение: ', round(p_value,2))
if p_value < alpha:
    print('Отвергаем нулевую гипотезу: между долями есть значимая разница')
else:
    print('Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными')
```

```
p-значение:  0.67
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: OffersScreenAppear в группах A1/A2

```
p-значение:  0.13
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: CartScreenAppear в группах A1/A2

```
p-значение:  0.2
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: PaymentScreenSuccessful в группах A1/A2

```
p-значение:  0.1
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Выводы к расчётам:

во всех случаях p-value больше 0.05, значит нулевую гипотезу не отвергаем (т.е.нет статистически значимого различия групп 246 и 247 по кол-ву пользователей по каждому из действий)

А также можем сделать вывод, что разбиение групп 246 и 247 (A/A) работает корректно.

Аналогично поступим с группой с изменённым шрифтом:

Out[42]:

	group	event_name	number	part,%
1	248	MainScreenAppear	2498	33.6
2	248	OffersScreenAppear	1536	20.7
0	248	CartScreenAppear	1236	16.6
3	248	PaymentScreenSuccessful	1183	15.9
4	248	Tutorial	282	3.8

Проверим, есть ли стистическая разница между группами A1(246) и B(248) по каждому из событий воронки:

- H0 = нет статистически значимого различия групп 246 и 248 по кол-ву пользователей по каждому из действий
- H1 = есть статистически значимое различие групп 246 и 248 по кол-ву пользователей по каждому из действий

Out[43]:

	group	event_name	number	part,%
1	246	MainScreenAppear	2452	33.0
2	246	OffersScreenAppear	1544	20.8
0	246	CartScreenAppear	1267	17.1
3	246	PaymentScreenSuccessful	1200	16.2
4	246	Tutorial	278	3.7

Событие: MainScreenAppear в группах A1/B

```
p-значение:  0.08
```

```
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: OffersScreenAppear в группах A1/B

```
p-значение:  0.23
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: CartScreenAppear в группах A1/B

```
p-значение:  0.05
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: PaymentScreenSuccessful в группах A1/B

```
p-значение:  0.19
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Выводы к расчётам:

по результатам тестов видим, что между группами A1 и B нет статистически значимой разницы, за исключением события CartScreenAppear, таким образом делаем вывод, что по этому событию группа B значимо уступает группе A1- такой вывод сделан при уровне значимости = 0.05,

но при уровне значимости 0.0125 (поправка Бонферрони) получаем противоположный результат, который говорит о том, что при первом расчёте мы получили ошибку первого рода.

И теперь делаем окончательный вывод, что между группами A1 и B нет статистически значимой разницы

Проверим, есть ли стистическая разница между группами A2(247) и B(248) по каждому из событий воронки:

- H0 = нет статистически значимого различия групп 247 и 248 по кол-ву пользователей по каждому из действий
- H1 = есть статистически значимое различие групп 247 и 248 по кол-ву пользователей по каждому из действий

Событие: MainScreenAppear в группах A2/B

```
p-значение:  0.18
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: OffersScreenAppear в группах A2/B

```
p-значение:  0.75
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: CartScreenAppear в группах A2/B

```
p-значение:  0.49
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: PaymentScreenSuccessful в группах A2/B

```
p-значение:  0.73
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Выводы к расчётам:

по результатам тестов видим, что между группами A2 и B нет статистически значимой разницы в результатах

Проверим, есть ли стистическая разница между группами A1+ A2 и B(248) по каждому из событий воронки:

- H0 = нет статистически значимого различия групп 246+247 и 248 по кол-ву пользователей по каждому из действий
- H1 = есть статистически значимое различие групп 246+247 и 248 по кол-ву пользователей по каждому из действий

Out[52]:

	event_name	number	part,%
1	MainScreenAppear	4931	33.6
2	OffersScreenAppear	3070	20.7
0	CartScreenAppear	2506	16.6
3	PaymentScreenSuccessful	2359	15.9
4	Tutorial	563	3.8

In [53]:
```
# расчитаем сколько пользователей в обединённой группе A1+A2:
cnt_userAA = data_AA['user_id'].nunique()
cnt_userAA
```

Out[53]:
```
5002
```

Событие: MainScreenAppear в группах A1+A2/B

```
p-значение:  0.07
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: OffersScreenAppear в группах A1+A2/B

```
p-значение:  0.61
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: CartScreenAppear в группах A1+A2/B

```
p-значение:  0.13
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Событие: PaymentScreenSuccessful в группах A1+A2/B

```
p-значение:  0.58
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Выводы к расчётам:

по результатам тестов видим, что между объединенной группой A1+A2 и B нет статистически значимой разницы в результатах (*При расчётах был выбран уровень значимости alpha = .05*)

Таким образом, по результатам проверок можем сделать общий вывод к A/A/B-тесту - изменение дизайнерами шрифта не повлияет на результаты продаж в мобильном приложении.

Проблема множественной проверки:

в ходе исследования проверка статистических гипотез была проведенана 4 раза (A1 и A2, A1 и B, A2 и B, A1+A2 и B, т.е. мы имеем дело со множественной проверкой.) Её важная особенность в том, что с каждой новой проверкой гипотезы растёт вероятность ошибки первого рода ошибкой первого рода, или ложнопозитивным результатом статистического теста. Это означает, что различий между сравниваемыми группами нет, но тест показал p-value меньше уровня значимости. Получается, есть основания отвергнуть H_0.

Таким образом при выбранном критическом уровне статистической значимости alpha = .05, вероятность ошибиться хотя бы раз за k сравнений:

1-(1-0,05)^4 = 0,185 (т.е.18,5%), что первышает выбранный уровень стат. значимости.

Для корректировки уровня значимости для уменьшения вероятности ошибки первого рода (FWER -family-wise error rate), воспользуемся поправкой Бонферрони, а именно поделим уровень значимости α на число гипотез:

0,05/4 = 0,0125

И сделаем перепроверку гипотез с новым уровнем стат. значимости.

В результате сделанной перепроверки с новым уровнем значимости:

Событие: CartScreenAppear в группах A1/B, по которому нулевая гипотеза была отвергнута получаем противоположный результат, т.е. в данном случае была ошибка теста, которая с помощью поправки Бонферрони была устранена.

Ход и результаты исследования:

В ходе исследования было выполнено:

- предобработка данных для анализа :добавлен столбце с датой, проверка на пропуски и дубликаты, а также скорректирован период анализа;
- при изучении воронки событий - построен график самой воронки, посчитано кол-во событий, кол-во пользователей на долю (в процентах) пользователей, которые хоть раз совершали событие на каждом шаге. А так также выяснено, что что на шаге перехода с главной страницы на страницу выбора товаров теряется больше всего пользователей: только 61,7% от первоначального кол-ва зашедших на гл. страницу перешло к выбору товаров. И 47,7 % пользователей проходят все шаги от главного экрана до покупки.
- при анализе результатов A/A/B-теста было выяснено, что отличия в конверсии групп A1, A2 и B не являются статистически значимыми, и таким образом изменение шрифта не повлияет на конверсию в приложении.