Тема:

Проведение проверки A/B теста.

Задача:

Оценить корректность проведения теста и результаты теста, аудитория 15% новых пользователей из региона EU.

Данные

- Тест "recommender_system_test", продолжительность теста 07 декабря 2020 по 04 января 2021.
- Календарь с маркетинговыми событиями: название, регион проведения, даты проведения.
- Данные о зарегистрировавшихся пользователях: айди, дата регистрации, регион, устройство
- Данные о действиях пользователей с 07.12.20 по 04.01.2021: айди, время события, тип события, детали события.
- Данные об участниках теста: айди, название теста, тестируемая группа

Импорт библиотек:

Ввод [1]:
```
from io import BytesIO
import requests
import pandas as pd
import matplotlib.pyplot as plt
from plotly import graph_objects as go
import seaborn as sns
from scipy import stats as st
import numpy as np
import math as mth
import datetime as dt
import warnings
warnings.filterwarnings("ignore")
```

Загрузка данных:

Даные с участниками теста:

Ввод [2]:
```
spreadsheet_id = '1uop1AEvwrFZRgKPiACy8ePVsMh2okXohB0f_kVUtxJE'
file_name = 'https://docs.google.com/spreadsheets/d/{}/export?format=csv'.format(spreadsheet_id)
r = requests.get(file_name)
d_part = pd.read_csv(BytesIO(r.content))
```

Ввод [3]:
```
display(d_part.head())
d_part.info()
```

	user_id	group	ab_test
0	D1ABA3E2887B6A73	A	recommender_system_test
1	A7A3664BD6242119	A	recommender_system_test
2	DABC14FDDFADD29E	A	recommender_system_test
3	04988C5DF189632E	A	recommender_system_test
4	482F14783456D21B	B	recommender_system_test

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18268 entries, 0 to 18267
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   user_id  18268 non-null  object
 1   group    18268 non-null  object
 2   ab_test  18268 non-null  object
dtypes: object(3)
memory usage: 428.3+ KB
```

Ввод [4]:
```
print('Уникальные значения тестов:')
print(d_part['ab_test'].unique())
print('Количество пропусков в данных:')
print(d_part.isnull().sum())
print('Количество дубликатов в данных:')
print(d_part.duplicated().sum())
```

```
Уникальные значения тестов:
['recommender_system_test' 'interface_eu_test']
Количество пропусков в данных:
user_id    0
group      0
ab_test    0
dtype: int64
Количество дубликатов в данных:
0
```

Пропусков в данных в датафрейме с данными участников теста нет, дубликаты пока не обнаружены, тип данных в колонках корретный, названия колонок корректны.

Даные с действиями новых пользователей:

Периода действий пользователей с 07 декабря 2020 по 04 января 2021.

Ввод [5]:
```
spreadsheet_id = '1b9PSwONJuRjXyYwsPD_9IOy8rqmHV3PxAhR0cHTwuuQ'
file_name = 'https://docs.google.com/spreadsheets/d/{}/export?format=csv'.format(spreadsheet_id)
r = requests.get(file_name)
d_events = pd.read_csv(BytesIO(r.content))
```

Ввод [6]:
```
display(d_events.head())
d_events.info()
```

	user_id	event_dt	event_name	details
0	E1BDDCE0DAFA2679	2020-12-07 20:22:03	purchase	99.99
1	7B6452F081F49504	2020-12-07 09:22:53	purchase	9.99
2	9CD9F34546DF254C	2020-12-07 12:59:29	purchase	4.99
3	96F27A054B191457	2020-12-07 04:02:40	purchase	4.99
4	1FD7660FDF94CA1F	2020-12-07 10:15:09	purchase	4.99

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440317 entries, 0 to 440316
Data columns (total 4 columns):
 #   Column      Non-Null Count   Dtype  
---  ------      --------------   -----  
 0   user_id     440317 non-null  object 
 1   event_dt    440317 non-null  object 
 2   event_name  440317 non-null  object 
 3   details     62740 non-null   float64
dtypes: float64(1), object(3)
memory usage: 13.4+ MB
```

Ввод [7]:
```
print('Количество пропусков в данных:')
print(d_events.isnull().sum()*100/len(d_events))
print()
print('Количество дубликатов в данных:')
print(d_events.duplicated().sum())
```

```
Количество пропусков в данных:
user_id        0.000000
event_dt       0.000000
event_name     0.000000
details       85.751175
dtype: float64

Количество дубликатов в данных:
0
```

Тип данных в столбце с датой некорректный, названия столбцов имеют корректный вид, 85,8% данных в столбце с дополнительными данными о событии пропущены, будет произведено дополнительное исследодвание столбца:

Ввод [8]:
```
print('Уникальные данные столбца с данными о событии:')
display(d_events['details'].unique())
print()
print('Уникальные данные столбца с названием события:')
display(d_events['event_name'].unique())
```

```
Уникальные данные столбца с данными о событии:

array([ 99.99,   9.99,   4.99, 499.99,    nan])


Уникальные данные столбца с названием события:

array(['purchase', 'product_cart', 'product_page', 'login'], dtype=object)
```

Ввод [9]:
```
d_events_nan = d_events.groupby('event_name')['details'].unique()
display(d_events_nan)
```

```
event_name
login                                [nan]
product_cart                         [nan]
product_page                         [nan]
purchase        [99.99, 9.99, 4.99, 499.99]
Name: details, dtype: object
```

Данные в столбце с событиями: оплата, корзина с товаром, страница с товаром, вход. Пропуски в данных находятся в деталях к событиям не подтверждающихся оплатой или иными дополнительными фиксациями переход от события к событию и есть подтверждающая деталь. Пока удалению или обработке не подлежат.

Тип данных в столбце с датой будет приведён к корректному типу данных:

Ввод [10]:
```
d_events['event_dt'] = d_events['event_dt'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d %H:%M:%S'
```

Ввод [11]:
```
d_events.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440317 entries, 0 to 440316
Data columns (total 4 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   user_id     440317 non-null  object
 1   event_dt    440317 non-null  datetime64[ns]
 2   event_name  440317 non-null  object
 3   details     62740 non-null   float64
dtypes: datetime64[ns](1), float64(1), object(2)
memory usage: 13.4+ MB
```

Даные о новых пользователях:

Ввод [12]:
```
spreadsheet_id = '1gYR8fUCmRPlO9BIge8RzQ53ZFx5r23hZINIM2hmlu2Y'
file_name = 'https://docs.google.com/spreadsheets/d/{}/export?format=csv'.format(spreadsheet_id)
r = requests.get(file_name)
d_users = pd.read_csv(BytesIO(r.content))
```

Ввод [13]:
```
display(d_users.head())
d_users.info()
```

	user_id	first_date	region	device
0	D72A72121175D8BE	2020-12-07	EU	PC
1	F1C668619DFE6E65	2020-12-07	N.America	Android
2	2E1BF1D4C37EA01F	2020-12-07	EU	PC
3	50734A22C0C63768	2020-12-07	EU	iPhone
4	E1BDDCE0DAFA2679	2020-12-07	N.America	iPhone

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61733 entries, 0 to 61732
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   user_id     61733 non-null  object
 1   first_date  61733 non-null  object
 2   region      61733 non-null  object
 3   device      61733 non-null  object
dtypes: object(4)
memory usage: 1.9+ MB
```

Ввод [14]:
```
print('Количество пропусков в данных:')
print(d_users.isnull().sum()*100/len(d_events))
print()
print('Количество дубликатов в данных:')
print(d_users.duplicated().sum())
```

```
Количество пропусков в данных:
user_id       0.0
first_date    0.0
region        0.0
device        0.0
dtype: float64

Количество дубликатов в данных:
0
```

Пропусков в данных нет, дубликатов нет, столбцы имеют корректные названия. Тип данных в столбце с датой некорректный.

Тип данных в столбце с датой будет приведён к корректному типу данных:

Ввод [15]:
```
d_users['first_date'] = d_users['first_date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
```

Ввод [16]:
```
d_users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61733 entries, 0 to 61732
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   user_id     61733 non-null  object        
 1   first_date  61733 non-null  datetime64[ns]
 2   region      61733 non-null  object        
 3   device      61733 non-null  object        
dtypes: datetime64[ns](1), object(3)
memory usage: 1.9+ MB
```

Даные с маркетинговыми событиями на 2020-й год :

Ввод [17]:
```
spreadsheet_id = '1LnJh3_jQn6_yeAR7r060RhKib956LB-LP4RTsFsbQgU'
file_name = 'https://docs.google.com/spreadsheets/d/{}/export?format=csv'.format(spreadsheet_id)
r = requests.get(file_name)
d_project = pd.read_csv(BytesIO(r.content))
```

Ввод [18]:
```
display(d_project.head())
d_project.info()
```

	name	regions	start_dt	finish_dt
0	Christmas&New Year Promo	EU, N.America	2020-12-25	2021-01-03
1	St. Valentine's Day Giveaway	EU, CIS, APAC, N.America	2020-02-14	2020-02-16
2	St. Patric's Day Promo	EU, N.America	2020-03-17	2020-03-19
3	Easter Promo	EU, CIS, APAC, N.America	2020-04-12	2020-04-19
4	4th of July Promo	N.America	2020-07-04	2020-07-11

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   name       14 non-null     object
 1   regions    14 non-null     object
 2   start_dt   14 non-null     object
 3   finish_dt  14 non-null     object
dtypes: object(4)
memory usage: 576.0+ bytes
```

Ввод [19]:
```
print('Количество пропусков в данных:')
print(d_project.isnull().sum()*100/len(d_events))
print()
print('Количество дубликатов в данных:')
print(d_project.duplicated().sum())
```

```
Количество пропусков в данных:
name         0.0
regions      0.0
start_dt     0.0
finish_dt    0.0
dtype: float64

Количество дубликатов в данных:
0
```

Пропусков в данных нет, дубликатов пока нет. Будет заменён тип данных в сталбцах с датой начала и окончания маркетингового события на корректный.

Ввод [20]:
```
d_project['start_dt'] = d_project['start_dt'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
d_project['finish_dt'] = d_project['finish_dt'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
```

Ввод [21]:
```
d_project.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype         
---  ------     --------------  -----         
 0   name       14 non-null     object        
 1   regions    14 non-null     object        
 2   start_dt   14 non-null     datetime64[ns]
 3   finish_dt  14 non-null     datetime64[ns]
dtypes: datetime64[ns](2), object(2)
memory usage: 576.0+ bytes
```

Вывод по разделу :

Во всех датафреймах данные с датой были приведены к корректному типу, пропуски в данных с событиями пока замене не подлежат. Дуюликаты не выявлены.

Подготовка данных:

Создание сводного датафрейма:

Корректировка данных для объединения в датасет:

Подготовка датасета с разбивкой по группам теста:

Ввод [22]:
```
display(d_part.head())
```

	user_id	group	ab_test
0	D1ABA3E2887B6A73	A	recommender_system_test
1	A7A3664BD6242119	A	recommender_system_test
2	DABC14FDDFADD29E	A	recommender_system_test
3	04988C5DF189632E	A	recommender_system_test
4	482F14783456D21B	B	recommender_system_test

Ввод [23]:
```
print('Исследование наличия пользователей, на попадание в обе группы:')
display(d_part.groupby('user_id').agg({'group':'nunique'}).reset_index().sort_values('group', ascendi
print()
print('Исследование наличия пользователей, на попадание в обе группы для необходимого теста:')
d_part_test=d_part[d_part['ab_test']=='recommender_system_test']
display(d_part_test.groupby('user_id').agg({'group':'nunique'}).reset_index().sort_values('group', as
print()
print('Количество пользователей в группах для двух тестов:')
display(d_part.groupby('group')['user_id'].nunique())
display(d_part.groupby('group')['user_id'].nunique().sum())
print()
print('Уникальность пользователей в группах для двух тестов:')
display(d_part.groupby('user_id')['group'].unique())
print()
print('Количество уникальных пользователей в тесте recommender_system_test в группах:')
display(d_part_test.groupby('group')['user_id'].nunique())
```

Исследование наличия пользователей, на попадание в обе группы:

	user_id	group
7402	72742C5F312A1FEC	2
11050	A929DCBED46E31D1	2
5068	4E971B47912497CE	2
7780	786EB537736D5055	2
5882	5B4EFE916AD19741	2

Исследование наличия пользователей, на попадание в обе группы для необходимого теста:

	user_id	group
0	000ABE35EE11412F	1
4502	AB3E2847DF7EC531	1
4474	AA703FB9F9004F86	1
4473	AA703605654827AE	1
4472	AA5A1803D3FA76B4	1

Количество пользователей в группах для двух тестов:

```
group
A    9173
B    8269
Name: user_id, dtype: int64
```

```
17442
```

Уникальность пользователей в группах для двух тестов:

```
user_id
0002CE61FF2C4011       [A]
000ABE35EE11412F       [A]
001064FEAAB631A1       [B]
0010A1C096941592       [A]
001C05E87D336C59       [A]
                      ...
FFE858A7845F005E       [A]
FFED90241D04503F       [B]
FFEFC0E55C1CCD4F       [B]
FFF28D02B1EACBE1    [B, A]
FFF58BC33966EB51       [B]
Name: group, Length: 16666, dtype: object
```

Количество уникальных пользователей в тесте recommender_system_test в группах:

```
group
A    3824
B    2877
Name: user_id, dtype: int64
```

В датасете с изучаемым тестом нет уникальных пользователей попавших в обе группы теста, но количество участников группы B на 1000 пользователей меньше.Возможно данные выправятся после выборки пользователей из региона EU.

Объединённый датасет с информацией о клиентах и их регистрации, и информацией о тесте и тестируемой группе.

Ввод [24]:
```
data = d_users.merge(d_part, on='user_id', how='left')
display(data.head())
data.info()
```

	user_id	first_date	region	device	group	ab_test
0	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test
1	F1C668619DFE6E65	2020-12-07	N.America	Android	NaN	NaN
2	2E1BF1D4C37EA01F	2020-12-07	EU	PC	A	interface_eu_test
3	50734A22C0C63768	2020-12-07	EU	iPhone	B	interface_eu_test
4	E1BDDCE0DAFA2679	2020-12-07	N.America	iPhone	NaN	NaN

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 63335 entries, 0 to 63334
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   user_id     63335 non-null  object        
 1   first_date  63335 non-null  datetime64[ns]
 2   region      63335 non-null  object        
 3   device      63335 non-null  object        
 4   group       18268 non-null  object        
 5   ab_test     18268 non-null  object        
dtypes: datetime64[ns](1), object(5)
memory usage: 3.4+ MB
```

В датасете много пропущенных данных о виде теста и нахождения пользователя в группе, не все пользователи были разделены или данные пользователи были зарегистрированы ранее.

Объединённый датасет с информацией о событии совершенном пользователем.

Ввод [25]:
```
data = data.merge(d_events, on='user_id', how='left')
```

Ввод [26]:
```
display(data.tail(10))
data.info()
```

	user_id	first_date	region	device	group	ab_test	event_dt	event_name	details
449946	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-23 15:47:23	product_page	NaN
449947	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-25 12:31:05	product_page	NaN
449948	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-27 03:51:36	product_page	NaN
449949	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-20 14:14:28	login	NaN
449950	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-21 04:31:15	login	NaN
449951	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-23 15:47:23	login	NaN
449952	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-25 12:31:04	login	NaN
449953	1C7D23927835213F	2020-12-20	EU	iPhone	B	interface_eu_test	2020-12-27 03:51:35	login	NaN
449954	8F04273BB2860229	2020-12-20	EU	Android	NaN	NaN	2020-12-20 03:17:17	product_cart	NaN
449955	8F04273BB2860229	2020-12-20	EU	Android	NaN	NaN	2020-12-20 03:17:17	login	NaN

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 449956 entries, 0 to 449955
Data columns (total 9 columns):
 #   Column      Non-Null Count   Dtype         
---  ------      --------------   -----         
 0   user_id     449956 non-null  object        
 1   first_date  449956 non-null  datetime64[ns]
 2   region      449956 non-null  object        
 3   device      449956 non-null  object        
 4   group       110368 non-null  object        
 5   ab_test     110368 non-null  object        
 6   event_dt    446211 non-null  datetime64[ns]
 7   event_name  446211 non-null  object        
 8   details     63588 non-null   float64       
dtypes: datetime64[ns](2), float64(1), object(6)
memory usage: 34.3+ MB
```

Пользователь под индексом 449954 зарегистрирован в нужное время, находится в регионе EU, но не отнесён к тесту и к группе теста,но совершал действия.

Добавление данных из таблицы с маркетинговыми событиями:

Ввод [27]:
```
data['first_date_month']=data['first_date'].astype('datetime64[M]')
d_project['start_dt_month']=d_project['start_dt'].astype('datetime64[M]')
data = data.merge(d_project, left_on='first_date_month', right_on='start_dt_month')
```

Ввод [28]:
```
display(data.head())
```

	user_id	first_date	region	device	group	ab_test	event_dt	event_name	details	first_
0	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:10	product_page	NaN	
1	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:10	product_page	NaN	
2	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:07	login	NaN	
3	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:07	login	NaN	
4	F1C668619DFE6E65	2020-12-07	N.America	Android	NaN	NaN	2020-12-07 16:38:09	product_page	NaN	

Доля пользователей попавших в группы A и B:

Ввод [29]:
```
# группировка для пользователей попавших в обе группы
d_part_group= d_part.groupby(['user_id']).agg({'group': 'nunique'}).reset_index().sort_values('group'

# общее количество пользователей попавших в тесты:
print('Общее количество пользователей попавших в тесты')
d_part_users = d_part_group['user_id'].count()
print(d_part_users)
print()

# количество пользователей попавших в две группы:
d_part_a_b = d_part_group[d_part_group['group'] > 1]['user_id'].count()
print('Общее количество пользователей попавших в обе группы')
print(d_part_a_b)
print()

# доля пользователей попавших в обе группы
print('Доля пользователей попавших в обе группы')
d_part_ratio = d_part_a_b/d_part_users*100
print(d_part_ratio)
print()
```

```
Общее количество пользователей попавших в тесты
16666

Общее количество пользователей попавших в обе группы
776

Доля пользователей попавших в обе группы
4.656186247449898
```

Вывод по общему датафрейму:

- Со временем проведения теста совпадают две маркетинговые кампании проводящиеся на разные регионы, на СНГ и Европу и Северную Америку. Камапания направленная на EU началась 25 декабря и закончилась 03 января. Конечный период совпадает с окончанием теста, но не совпадает с началом. Данная акция даже по названию, скорее направлена на поддержание спроса в период с рождества в Европе с 25 декабря до нового года, из-за иссякающего у европейцев спроса после рождественских покупок.
- Доля пользователей попавших в обе группы относительно относительно всех пользователей крайне мала и так как группы при этом находятся в разных тестах, данное разделение не отразится на результатах теста.

Выборка из общего датафрема по региону:

Согласно заданию будет произведена выборка по региону нахождения пользователя и по соответствующему региону маркетинговой кампании.

Исследование количества пользователей попавших из региона в тест:

Ввод [30]:
```
# количество пользователей из EU
users_eu = data[data['region']=='EU']['user_id'].nunique()
print('Количество пользователей из EU')
print(users_eu)
print()

# количество пользователей из EU попавших в тест
users_eu_test = data[(data['region']=='EU')&(data['ab_test']=='recommender_system_test')]['user_id'].
print('Количество пользователей из EU попавших в тест')
print(users_eu_test)
print()

# доля пользователей попавших в тест
print('Доля пользователей EU попавших в тест')
users_ratio = users_eu_test/users_eu*100
print(users_ratio)
print()

# доля пользователей попавших в тест
print('Количество пользователей EU ожидаемых в тесте:')
users_eu_wait = users_eu*0.15
print(users_eu_wait)
print()
```

```
Количество пользователей из EU
46270

Количество пользователей из EU попавших в тест
6351

Доля пользователей EU попавших в тест
13.725956343202938

Количество пользователей EU ожидаемых в тесте:
6940.5
```

Доля попавших в тест из необходимого региона составила меньше ожидаемых 15%. Для проверки насколько эти расхождения повлияют на прохождение тетс абудет проведёно z-тестирование долей о присутствии статистической разницы в долях:

Проверка гипотезы о статистической значимости в долях:

Но: "Нет статистической значимости в долях"

Н1: "Есть статистическая значимость в долях"

Ввод [31]:
```
purchases = np.array([users_eu_wait,users_eu_test])
leads = np.array([users_eu,users_eu])

alpha = 0.05
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/leads[0] + 1/leads[1]))
distr = st.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: в конверсиях есть значимая разница')
else:
    print('Не получилось отвергнуть нулевую гипотезу, нет оснований считать конверсии разными')
```

```
p-значение:  3.286807381819301e-08
Отвергаем нулевую гипотезу: в конверсиях есть значимая разница
```

Проверка обнаружиа статистическую значимость, аудитория теста не была сформировна правильно, что может сказаться на результатах теста.

Рассчёт доли пользователей EU и других регионов из общего числа пользователей.

Ввод [32]:
```
# количество пользователей из иных регионов
users_other_regions = data[data['region'] != 'EU']['user_id'].nunique()

# количество пользоватлей всего
users_general = data['user_id'].nunique()

# Доля пользователей других регионов
other_regions_users_ratio = (users_other_regions / users_general)*100
print('Доля пользователей других регионов из общего числа пользователей')
print(other_regions_users_ratio)
print()

# Доля пользователей EU из общего числа пользователей
eu_regions_users_ratio = users_eu/users_general*100
print('Доля пользователей EU из общего числа пользователей')
print(eu_regions_users_ratio)
print()
```

```
Доля пользователей других регионов из общего числа пользователей
25.048191404921194

Доля пользователей EU из общего числа пользователей
74.9518085950788
```

Доля пользователей из иных регионов составила 25%, ситема выделения сработала с ошибками.

Выделение датафрейма соответственно региону и проведения рекламной акции:

Ввод [33]:
```
data = data[(data['region']=='EU')&(data['regions']=='EU, N.America')]
display(data.head(3))
data.info()
```

	user_id	first_date	region	device	group	ab_test	event_dt	event_name	details	first_dat
0	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:10	product_page	NaN	2(
2	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test	2020-12-07 21:52:07	login	NaN	2(
16	2E1BF1D4C37EA01F	2020-12-07	EU	PC	A	interface_eu_test	2020-12-07 09:05:47	product_cart	NaN	2(

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 336578 entries, 0 to 899910
Data columns (total 15 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   user_id           336578 non-null  object
 1   first_date        336578 non-null  datetime64[ns]
 2   region            336578 non-null  object
 3   device            336578 non-null  object
 4   group             108934 non-null  object
 5   ab_test           108934 non-null  object
 6   event_dt          332989 non-null  datetime64[ns]
 7   event_name        332989 non-null  object
 8   details           49400 non-null   float64
 9   first_date_month  336578 non-null  datetime64[ns]
 10  name              336578 non-null  object
 11  regions           336578 non-null  object
 12  start_dt          336578 non-null  datetime64[ns]
 13  finish_dt         336578 non-null  datetime64[ns]
 14  start_dt_month    336578 non-null  datetime64[ns]
dtypes: datetime64[ns](6), float64(1), object(8)
memory usage: 41.1+ MB
```

Тест сильно пересекается с новогодней промоакцией в EU. Это не слишком хорошо - необходимо избегать таких ситуаций.

Выборка данных с заданным тестом:

Согласно заданию будет произведена выборка по необходимому тесту.

Ввод [34]:
```
data = data[data['ab_test']=='recommender_system_test']
display(data.tail(5))
data.info()
```

	user_id	first_date	region	device	group	ab_test	event_dt	event_name	details	firs
899626	0416B34D35C8C8B8	2020-12-20	EU	Android	A	recommender_system_test	2020-12-24 09:12:51	product_page	NaN	
899628	0416B34D35C8C8B8	2020-12-20	EU	Android	A	recommender_system_test	2020-12-20 20:58:25	login	NaN	
899630	0416B34D35C8C8B8	2020-12-20	EU	Android	A	recommender_system_test	2020-12-21 22:28:29	login	NaN	
899632	0416B34D35C8C8B8	2020-12-20	EU	Android	A	recommender_system_test	2020-12-24 09:12:49	login	NaN	
899662	89CB0BFBC3F35126	2020-12-20	EU	PC	B	recommender_system_test	NaT	NaN	NaN	

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 26290 entries, 0 to 899662
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   user_id           26290 non-null  object
 1   first_date        26290 non-null  datetime64[ns]
 2   region            26290 non-null  object
 3   device            26290 non-null  object
 4   group             26290 non-null  object
 5   ab_test           26290 non-null  object
 6   event_dt          23420 non-null  datetime64[ns]
 7   event_name        23420 non-null  object
 8   details           3196 non-null   float64
 9   first_date_month  26290 non-null  datetime64[ns]
 10  name              26290 non-null  object
 11  regions           26290 non-null  object
 12  start_dt          26290 non-null  datetime64[ns]
 13  finish_dt         26290 non-null  datetime64[ns]
 14  start_dt_month    26290 non-null  datetime64[ns]
dtypes: datetime64[ns](6), float64(1), object(8)
memory usage: 3.2+ MB
```

Данные о количестве пользователей в группах:

Ввод [35]:
```
display(data.groupby('user_id').agg({'group':'nunique'}).reset_index().sort_values('group', ascending
print('Количество пользователей в группах:')
display(data.groupby('group')['user_id'].nunique())
print('Уникальность пользователей в группах:')
display(data.groupby('user_id')['group'].unique())
```

	user_id	group
0	000ABE35EE11412F	1
4242	AA815357216525A9	1
4240	AA703FB9F9004F86	1
4239	AA703605654827AE	1
4238	AA5A1803D3FA76B4	1

```
Количество пользователей в группах:

group
A    3634
B    2717
Name: user_id, dtype: int64

Уникальность пользователей в группах:

user_id
000ABE35EE11412F    [A]
001064FEAAB631A1    [B]
0010A1C096941592    [A]
001C05E87D336C59    [A]
00341D8401F0F665    [A]
                   ...
FFC2C5F898D1245B    [B]
FFC53FD45DDA5EE8    [B]
FFE858A7845F005E    [A]
FFED90241D04503F    [B]
FFF28D02B1EACBE1    [B]
Name: group, Length: 6351, dtype: object
```

Каждый пользователь из проверяемого теста находится только в одной группе.

Ввод [36]:
```
print('Количество дубликатов в данных:')
print(data.duplicated().sum())
print('Количество дубликатов в процентах')
print(data.duplicated().sum()*100/len(data))
```

```
Количество дубликатов в данных:
0
Количество дубликатов в процентах
0.0
```

В сводной таблице 26290 событий, дубликатов в данных нет будет произведено изучение и проверка данных, но количесвто участников в группах имеет большое различие, необходимо изменить в будущем механику выделения групп.

Изучение и проверка данных:

Изучение событий в логе:

Сводная таблица с данными по событиям:

Ввод [37]:
```
print('Сводная таблица с количеством событий в логе:')
event_count = data.groupby('event_name').agg({'user_id':'count'}).sort_values(by='user_id', ascending
event_count.columns=['event_name', 'event_counts']
display(event_count)
```

```
Сводная таблица с количеством событий в логе:
```

	event_name	event_counts
0	login	10595
1	product_page	6554
2	purchase	3196
3	product_cart	3075

Ввод [38]:
```
plt.figure(figsize=(15,5))
sns.barplot(data=event_count, x="event_counts", y="event_name")
plt.title('График распределения событий по количеству:', size=14)
plt.xlabel('Количество событий', size=12)
plt.ylabel('Наименование событий',size=12)
plt.grid(axis='x')
plt.show()
```

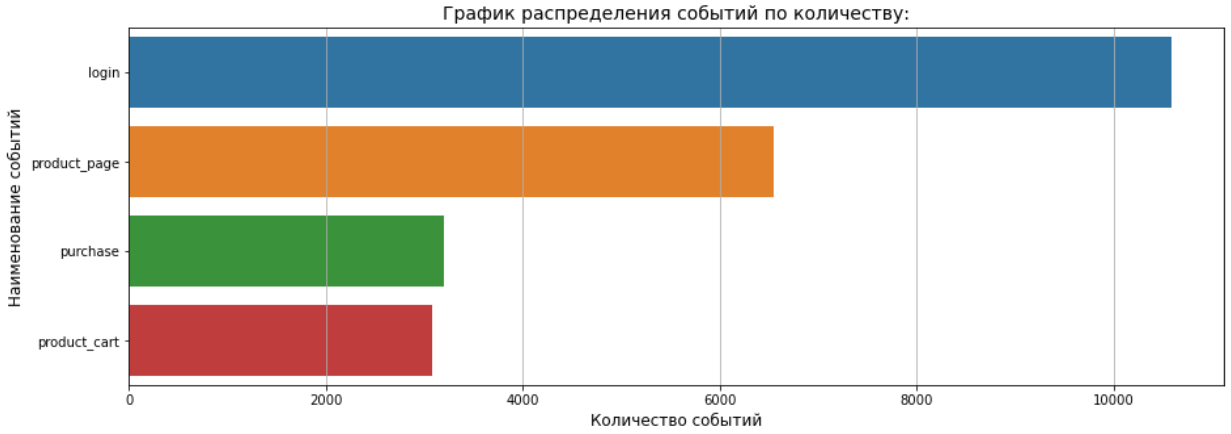


Таблица событий выглядела бы как воронка событий, если бы событие - "оплата" не превышало событие "товарная корзина", возможно это произошло из-за предложения оплаты в один клик, минуя корзину с товарами. Большие потери порядка 40% происходят на этапе перехода от входа на страницу с товарами.

Определение общего числа событий в логе:

Ввод [39]:
```
print('Общее количество событий в логе:')
events_total = event_count['event_counts'].sum()
print(events_total)
```

```
Общее количество событий в логе:
23420
```

Количество пользователей в логе:

Ввод [40]:
```
user_total = data['user_id'].nunique()
print('Количество уникальных пользователей в логе:')
print(user_total)
```

```
Количество уникальных пользователей в логе:
6351
```

В выборку попало более 6000 ожидаемых пользователей

Среднее количество событий на пользователя:

Ввод [41]:
```
# общие расчёты
print('Среднее количество событий приходящихся на одного пользователя в логе:')
event_mean = events_total/user_total
print(event_mean.round())
print()

# рассчёты для группы A
print('Среднее количество событий приходящихся на одного пользователя в группе A:')
group_a=data[data['group']=='A']
user_a_total = group_a['user_id'].nunique()
events_a_total = group_a['event_name'].count()
event_mean_a = events_a_total/user_a_total
print(event_mean_a.round())
print()

# рассчёты для группы B
print('Среднее количество событий приходящихся на одного пользователя в группе B:')
group_b=data[data['group']=='B']
user_b_total = group_b['user_id'].nunique()
events_b_total = group_b['event_name'].count()
event_mean_b = events_b_total/user_b_total
print(event_mean_b.round())
```

```
Среднее количество событий приходящихся на одного пользователя в логе:
4.0

Среднее количество событий приходящихся на одного пользователя в группе A:
5.0

Среднее количество событий приходящихся на одного пользователя в группе B:
2.0
```

Среднее количество покупок на пользователя:

Ввод [42]:
```
# средене количесвто покупок на пользователя в логе
purchase_mean = event_count.loc[2,'event_counts']/user_total
print('Среднее количество покупок приходящихся на одного пользователя в логе:')
print(purchase_mean)
print()

# среднее количесвто покупок на пользователя в группе A
purchase_mean_a = (group_a[group_a['event_name']=='purchase']['user_id'].count())/user_a_total
print('Среднее количество покупок приходящихся на одного пользователя в группе A:')
print(purchase_mean_a)
print()

# среднее количесвто покупок на пользователя в группе B
purchase_mean_b = (group_b[group_b['event_name']=='purchase']['user_id'].count())/user_b_total
print('Среднее количество покупок приходящихся на одного пользователя в группе B:')
print(purchase_mean_b)
print()
```

```
Среднее количество покупок приходящихся на одного пользователя в логе:
0.5032278381357267

Среднее количество покупок приходящихся на одного пользователя в группе A:
0.6978536048431481

Среднее количество покупок приходящихся на одного пользователя в группе B:
0.242914979757085
```

Вывод по разделу:

Среднее количество покупок у группы B заметно ниже. Похоже новая рекомендательная система делает только хуже.

Исследование временного периода:

Начальная и конечная дата периода:

Ввод [43]:
```
min_date=data['event_dt'].min()
print('Начальная дата событий')
print(min_date)
print()

max_date=data['event_dt'].max()
print('Конечная дата событий')
print(max_date)
print()
min_date_reg=data['first_date'].min()
print('Начальная дата периода регистрации')
print(min_date_reg)
print()
max_date_reg=data['first_date'].max()
print('Конечная дата периода регистрации')
print(max_date_reg)
print()
```

```
Начальная дата событий
2020-12-07 00:05:57

Конечная дата событий
2020-12-30 12:42:57

Начальная дата периода регистрации
2020-12-07 00:00:00

Конечная дата периода регистрации
2020-12-21 00:00:00
```

Гистограмма распеределения по дате и времени события:

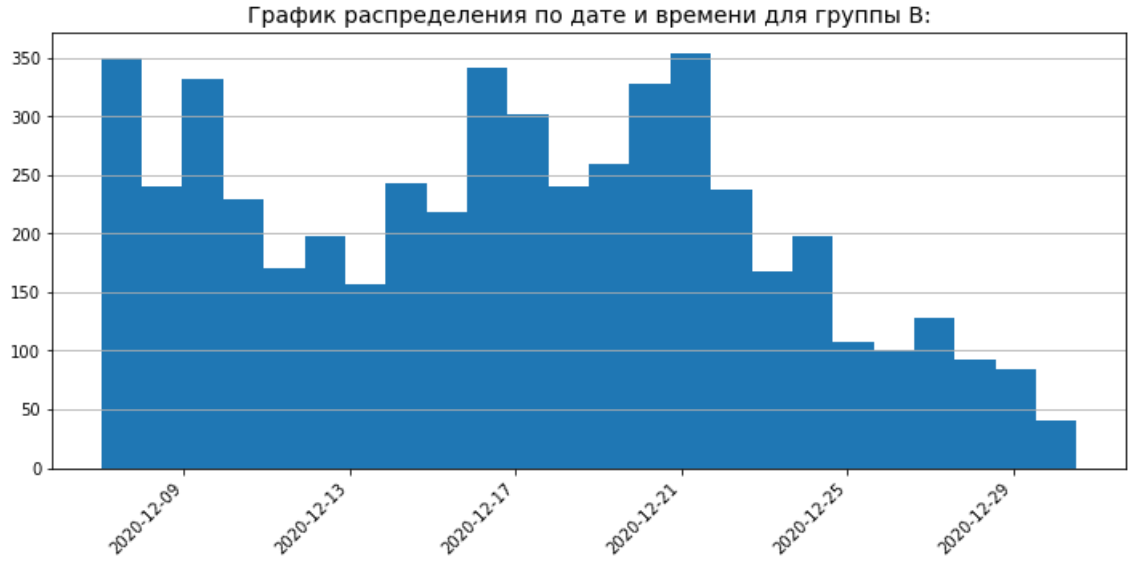
Ввод [44]:
```
plt.figure(figsize=(12,5))
data['event_dt'].hist(bins=24)
plt.xticks(rotation=45, ha='right')
plt.title('График распределения по дате и времени:', size=14)
plt.grid(axis='x')
plt.show()
```

На гистограмме количество событий с момента регистрации пользователей увеличивается до 17-го декабря, происхоит резкий спад событий после рождества, что закономерно, но в это же время начинается рекланая камапния с новогодними предложениями. Фиксация событий заканчивается 30 декабря. В период с 31.12 по 04.01 ничего не происходит. В качестве периода для изучения событий будет выбран имеющийся период, с момента начала регистрации до первого действия пользователя, предположительно поступила вся информация.

Ввод [45]:
```
# распределение для группы A
plt.figure(figsize=(12,5))
group_a['event_dt'].hist(bins=24)
plt.xticks(rotation=45, ha='right')
plt.title('График распределения по дате и времени для группы A:', size=14)
plt.grid(axis='x')
plt.show()
```

На гистограмме гистограмме для группы A количество событий распределяется похожим образом, что и для группы B. Также количество событий увеличиваетя с начала регистрации, имеет падение 25 декабря и всё заканчивается 30 декабря.

Ввод [46]:
```
# распределение для группы B
plt.figure(figsize=(12,5))
group_b['event_dt'].hist(bins=24)
plt.xticks(rotation=45, ha='right')
plt.title('График распределения по дате и времени для группы B:', size=14)
plt.grid(axis='x')
plt.show()
```



В группе B количество событий на порядок меньше, чем в группе A пиковое значение в группе B в пять раз меньше чем в группе A, скорее всего, что-то не так с нововведениями.

Получение воронки событий:

Вычисление совершённых пользователями событий. Сортировка сводной таблицы по совершенным событиям.

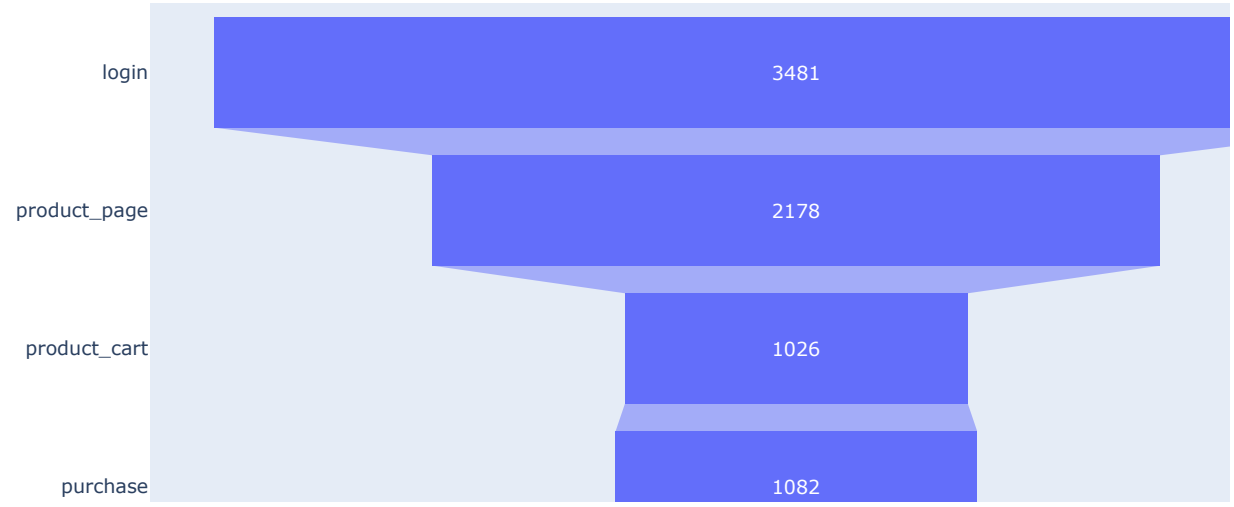
Ввод [47]:
```
event_funnel= data.groupby('event_name').agg({'user_id': 'nunique'}).reindex(
    ['login', 'product_page', 'product_cart', 'purchase']).reset_index()
event_funnel.columns=['event_name', 'users_counts']

display(event_funnel)
```

	event_name	users_counts
0	login	3481
1	product_page	2178
2	product_cart	1026
3	purchase	1082

В распределении количества событий на каждого пользователя событие оплаты совершили больше пользователей, чем событие "товарная корзина".

Ввод [48]:
```
print('Распределение событий на уникального пользователя:')
fig = go.Figure(
    go.Funnel(
        y=event_funnel['event_name'],
        x=event_funnel['users_counts']
        ))
fig.show()
```

```
Распределение событий на уникального пользователя:
```



Потери в пользователях между входом и посещением страницы с товарами составляет 42%, 31% из вошедших оплачивают и 29,5% попадают в корзину.

Определение количества пользователей в экспериментальных группах:

Ввод [49]:
```
users_funnel_max = event_funnel['users_counts'].max()
ratio_users_event = users_funnel_max / user_total
print('Доля пользователей, совершивших событие хотя бы раз:')
print('{:.0%}'.format(ratio_users_event))
print()
print('Количество посетителей главной страницы:')
print(users_funnel_max)
```

```
Доля пользователей, совершивших событие хотя бы раз:
55%

Количество посетителей главной страницы:
3481
```

Из всех зарегистрированных пользователей только 55% процентов переходит на ресурс. Так как не 100% переходят на главную страницу рассчитывать конверсии переходов необходимо к общему количеству зарегистрированных пользователей в группах.

Исследование воронки событий:

Ввод [50]:
```
display(event_funnel)
```

	event_name	users_counts
0	login	3481
1	product_page	2178
2	product_cart	1026
3	purchase	1082

Доля пользователей перешедших с главной страницы на страницу с товарами:

Ввод [51]:
```
users_funnel_offers = event_funnel.loc[1,'users_counts']/event_funnel.loc[0,'users_counts']
print('Доля пользователей перешедших с главной страницы на страницу с товарами:')
print('{:.0%}'.format(users_funnel_offers))
```

```
Доля пользователей перешедших с главной страницы на страницу с товарами:
63%
```

Доля пользователей перешедших в корзину со страницы с товарами:

Ввод [52]:
```
users_funnel_carts = event_funnel.loc[2,'users_counts']/event_funnel.loc[1,'users_counts']
print('Доля пользователей перешедших в корзину со страницы с товарами:')
print('{:.0%}'.format(users_funnel_carts))
```

```
Доля пользователей перешедших в корзину со страницы с товарами:
47%
```

Доля пользователей оплативших заказ из перешедших со страницы с товарами:

Ввод [53]:
```
users_funnel_pay = event_funnel.loc[3,'users_counts']/event_funnel.loc[1,'users_counts']
print('Доля пользователей оплативших заказ из перешедших со страницы с товарами:')
print('{:.0%}'.format(users_funnel_pay))
```

```
Доля пользователей оплативших заказ из перешедших со страницы с товарами:
50%
```

Главными остаются потери при переходена главную страницу, 40% пользователей даже не знакомятся с продуктом.

Проверка A/B теста:

Выборка для группы A:

Ввод [54]:
```
group_a_funnel = group_a.groupby('event_name').agg({'user_id':'nunique'}).sort_values(
    by='user_id', ascending=False).reindex(
    ['login', 'product_page', 'product_cart', 'purchase']).reset_index()
group_a_funnel.columns=['event_name', 'user_A']

print('Сводная таблица с данными для группы A:')
display(group_a_funnel)

print('Общее количество пользователей в группе A:')
group_a_funnel_sum =group_a['user_id'].nunique()
print(group_a_funnel_sum)
```

```
Сводная таблица с данными для группы A:
```

	event_name	user_A
0	login	2604
1	product_page	1685
2	product_cart	782
3	purchase	833

```
Общее количество пользователей в группе A:
3634
```

Выборка для группы B:

Ввод [55]:
```
group_b_funnel = group_b.groupby('event_name').agg({'user_id':'nunique'}).sort_values(
    by='user_id', ascending=False).reindex(
    ['login', 'product_page', 'product_cart', 'purchase']).reset_index()
group_b_funnel.columns=['event_name', 'user_B']

print('Сводная таблица с данными для группы B:')
display(group_b_funnel)

print('Общее количество пользователей в группе B:')
group_b_funnel_sum =group_b['user_id'].nunique()
print(group_b_funnel_sum)
```

```
Сводная таблица с данными для группы B:
```

	event_name	user_B
0	login	877
1	product_page	493
2	product_cart	244
3	purchase	249

```
Общее количество пользователей в группе B:
2717
```

Распределение пользователей по группам произведено некорректно, пользователей в группе B меньше примерно на 1000 человек. Будет произведён рассчёт конверсий для каждой группы теста:

Ввод [56]:
```
print('ЗНАЧЕНИЕ КОНВЕРСИЙ ДЛЯ ГРУППЫ A:')
print()
print()
# доля доля вошедших пользоватлей
ratio_a_login = group_a_funnel.loc[0,'user_A'] / group_a_funnel_sum
print('Доля вошедших пользователей контрольной группы A:')
print('{:.0%}'.format(ratio_a_login))
print()

# доля посетивших страницу с товаром
ratio_a_offers = group_a_funnel.loc[1,'user_A'] / group_a_funnel_sum
print('Доля пользователей контрольной группы перешедших с главной страницы на страницу с товарами:')
print('{:.0%}'.format(ratio_a_offers))
print()

# доля перешедших в корзину
ratio_a_carts = group_a_funnel.loc[2,'user_A'] / group_a_funnel_sum
print('Доля пользователей контрольной группы перешедших в корзину:')
print('{:.0%}'.format(ratio_a_carts))
print()

# доля перешедших оплативших
ratio_a_pay = group_a_funnel.loc[3,'user_A'] / group_a_funnel_sum
print('Доля оплативших пользователей контрольной группы:')
print('{:.0%}'.format(ratio_a_pay))
```

```
ЗНАЧЕНИЕ КОНВЕРСИЙ ДЛЯ ГРУППЫ A:


Доля вошедших пользователей контрольной группы A:
72%

Доля пользователей контрольной группы перешедших с главной страницы на страницу с товарами:
46%

Доля пользователей контрольной группы перешедших в корзину:
22%

Доля оплативших пользователей контрольной группы:
23%
```

Ввод [57]:
```
print('ЗНАЧЕНИЕ КОНВЕРСИЙ ДЛЯ ГРУППЫ B:')
print()
print()
# доля доля вошедших пользоватлей
ratio_b_login = group_b_funnel.loc[0,'user_B'] / group_a_funnel_sum
print('Доля вошедших пользователей контрольной группы B:')
print('{:.0%}'.format(ratio_b_login))
print()

# доля посетивших страницу с товаром
ratio_b_offers = group_b_funnel.loc[1,'user_B'] / group_b_funnel_sum
print('Доля пользователей экспериментальной группы перешедших с главной страницы на страницу с товара
print('{:.0%}'.format(ratio_b_offers))
print()

# доля перешедших в корзину
ratio_b_carts = group_b_funnel.loc[2,'user_B'] / group_b_funnel_sum
print('Доля пользователей экспериментальной группы перешедших в корзину:')
print('{:.0%}'.format(ratio_b_carts))
print()

# доля перешедших оплативших
ratio_b_pay = group_b_funnel.loc[3,'user_B'] / group_b_funnel_sum
print('Доля оплативших пользователей экспериментальной группы:')
print('{:.0%}'.format(ratio_b_pay))
print()
```

```
ЗНАЧЕНИЕ КОНВЕРСИЙ ДЛЯ ГРУППЫ B:


Доля вошедших пользователей контрольной группы B:
24%

Доля пользователей экспериментальной группы перешедших с главной страницы на страницу с товарами:
18%

Доля пользователей экспериментальной группы перешедших в корзину:
9%

Доля оплативших пользователей экспериментальной группы:
9%
```

У группы B в сравнению с группой A:

- Доля вошедших пользователей из общего числа пользователей в группе на страницу товара снизилась с 72% до 24%
- Доля пользователей перешедших из общего числа пользователей в группе на страницу товара снизилась с 46% до 18%
- Доля пользователей перешедших из общего числа пользователей в корзину в группе снизилась с 22% до 9%
- Доля пользователей проведших оплату из общего числа пользователей снизилась с 23% до 9%

Новая рекомендательная система ухудшает показатели

Функция для проверки гипотезы о разнице в конверсиях A/B:

Ввод [58]:
```
def test_convers(purchases, leads):
    alpha = 0.05
    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/leads[0] + 1/leads[1]))
    distr = st.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: в конверсиях есть значимая разница')
    else:
        print(
        'Не получилось отвергнуть нулевую гипотезу, нет оснований считать конверсии разными')
```

Проверка гипотезы о разнице в конверсиях A/B:

Но: Нет значимой разницы в конверсиях между группами.

Н1: Есть значимая разница в конверсиях между группами.

Проверка гипотезы разницы в конверсиях между группами для события:

"Свершили вход товарами из общего числа пользователей."

Ввод [59]:
```
purchases = np.array([group_a_funnel.loc[0, 'user_A'], group_b_funnel.loc[0, 'user_B']])
leads = np.array([group_a_funnel_sum, group_b_funnel_sum])
test_convers(purchases, leads)
```

```
p-значение:  0.0
Отвергаем нулевую гипотезу: в конверсиях есть значимая разница
```

Проверка гипотезы разницы в конверсиях между группами для события:

"Перешли на страницу с товарами из общего числа пользователей."

Ввод [60]:
```
purchases = np.array([group_a_funnel.loc[1, 'user_A'], group_b_funnel.loc[1, 'user_B']])
leads = np.array([group_a_funnel_sum, group_b_funnel_sum])
test_convers(purchases, leads)
```

```
p-значение:  0.0
Отвергаем нулевую гипотезу: в конверсиях есть значимая разница
```

Проверка гипотезы разницы в конверсиях между группами для события:

"Перешли в корзину относительно из общего числа пользователей."

Ввод [61]:
```
purchases = np.array([group_a_funnel.loc[2, 'user_A'], group_b_funnel.loc[2, 'user_B']])
leads = np.array([group_a_funnel_sum, group_b_funnel_sum])
test_convers(purchases, leads)
```

```
p-значение:  0.0
Отвергаем нулевую гипотезу: в конверсиях есть значимая разница
```

Проверка гипотезы разницы в конверсиях между группами для события:

"Провели оплату из общего числа пользователей:"

Ввод [62]:
```
purchases = np.array([group_a_funnel.loc[3, 'user_A'], group_b_funnel.loc[3, 'user_B']])
leads = np.array([group_a_funnel_sum, group_b_funnel_sum])
test_convers(purchases, leads)
```

```
p-значение:  0.0
Отвергаем нулевую гипотезу: в конверсиях есть значимая разница
```

Вывод:

По результатам проверки теста, выявлено следующее:

- поступление информации закончилось ранее 04 января,
- на этапе выделения пользователей 25% оказались из других регионов,
- количество пользователей зарегистрировавшихся превысило ожидаемые 6000 человек, но статистически важная доля в 15% это 6940 пользователей, разница почти в 1000 человек.
- пользователи были распределены на группы неравномерно с перевесом в 1000 человек в группе А
- распределение событий в группе В показывает их меньшее количество и плотность, на пользователей группы В приходится меньше покупок, меньше событий

Рекомендация:

Тест проведён некорректно, результатам данного теста доверять нельзя, несмотря на ухудшиевся конверсии в группе В в воронке событий. Проведение теста повторно с учётом ошибок допустимо.

Ввод [ ]:
```
```