Проект: вариант 1

Представьте, что вы работаете в компании, которая разрабатывает мобильные игры. К вам пришел менеджер с рядом задач по исследованию нескольких аспектов мобильного приложения:

```
В первую очередь, его интересует показатель retention. Напишите функцию для его подсчета.
Помимо этого, в компании провели A/B тестирование наборов акционных предложений. На основе имеющихся данных опре
делите,     какой набор можно считать лучшим и на основе каких метрик стоит принять правильное решение.
Предложите метрики для оценки результатов последнего прошедшего тематического события в игре.
```

Project: option 1

Imagine that you work for a company that develops mobile games. A manager came to you with a number of tasks to study several aspects of a mobile application:

```
First of all, he is interested in the retention indicator. Write a function to calculate it.
In addition, the company conducted A / B testing of sets of promotional offers. Based on the available data, det
ermine which     set can be considered the best and on the basis of which metrics it is worth making the right d
ecision.
Offer metrics to evaluate the results of the last past thematic event in the game.
```

In [55]:
```
import pandas as pd
import numpy as np

from scipy import stats
import matplotlib.pyplot as plt
```

Задание 1

Retention – один из самых важных показателей в компании. Ваша задача – написать функцию, которая будет считать retention игроков (по дням от даты регистрации игрока). Данные лежат в папке shared и имеют следующую структуру:

`shared/problem1-reg_data.csv – данные о времени регистрации`

`shared/problem1-auth_data.csv – данные о времени захода пользователей в игру`

Функция должна быть написана на python. В ходе решения можно тестировать работу функции как на полном датасете, так и на части (сэмпле) данных.

Exercise 1

Retention – is one of the most important indicators in the company. Your task is to write a function that will count the retention of players (by days from the date of registration of the player). The data is in the shared folder and has the following structure:

`shared/problem1-reg_data.csv – registration time data`

`shared/problem1-auth_data.csv – data on the time users entered the game`

The function must be written in python. During the solution, you can test the operation of the function both on the full dataset and on a part (sample) of data.

In [2]:
```
# считаем данные о времени регистрации
# counting data on registration time

reg_data = pd.read_csv('/home/jupyter-i-sharapova-17/shared/problem1-reg_data.csv', sep=';')
reg_data.head()
```

Out[2]:

| | reg_ts | uid |
|---|---|---|
| **0** | 911382223 | 1 |
| **1** | 932683089 | 2 |
| **2** | 947802447 | 3 |
| **3** | 959523541 | 4 |
| **4** | 969103313 | 5 |

In [3]:
```
# считаем данные о времени захода пользователей в игру
# counting data on the time users entered the game

auth_data = pd.read_csv('/home/jupyter-i-sharapova-17/shared/problem1-auth_data.csv', sep=';')
auth_data.head()
```

Out[3]:

| | auth_ts | uid |
|---|---|---|
| **0** | 911382223 | 1 |
| **1** | 932683089 | 2 |
| **2** | 932921206 | 2 |
| **3** | 933393015 | 2 |
| **4** | 933875379 | 2 |

In [ ]:

In [4]:
```
# посмотрим на типы данных
# look at data types

reg_data.dtypes
```

Out[4]:
```
reg_ts    int64
uid       int64
dtype: object
```

In [5]:
```
auth_data.dtypes
```

Out[5]:
```
auth_ts    int64
uid        int64
dtype: object
```

In [ ]:
```

```

In [6]:
```
# время в формате UNIX time — число секунд, прошедших с 1970 года
# приведем к удобному формату колонки со временем

# time in UNIX time format - the number of seconds since 1970
# bring to a convenient column format over time


reg_data['reg_ts'] = pd.to_datetime(reg_data.reg_ts, unit='s')
reg_data.head()
```

Out[6]:

| | reg_ts | uid |
|---|---|---|
| **0** | 1998-11-18 09:43:43 | 1 |
| **1** | 1999-07-22 22:38:09 | 2 |
| **2** | 2000-01-13 22:27:27 | 3 |
| **3** | 2000-05-28 14:19:01 | 4 |
| **4** | 2000-09-16 11:21:53 | 5 |

In [7]:
```
auth_data['auth_ts'] = pd.to_datetime(auth_data.auth_ts, unit='s')
auth_data.head()
```

Out[7]:

| | auth_ts | uid |
|---|---|---|
| **0** | 1998-11-18 09:43:43 | 1 |
| **1** | 1999-07-22 22:38:09 | 2 |
| **2** | 1999-07-25 16:46:46 | 2 |
| **3** | 1999-07-31 03:50:15 | 2 |
| **4** | 1999-08-05 17:49:39 | 2 |

In [ ]:

In [ ]:

In [8]:
```
# проверим тип данных
# check data type

reg_data.dtypes
```

Out[8]:
```
reg_ts    datetime64[ns]
uid                int64
dtype: object
```

In [9]:
```
auth_data.dtypes
```

Out[9]:
```
auth_ts    datetime64[ns]
uid                 int64
dtype: object
```

In [ ]:

In [10]:
```
# проверим размерность датасетов
# check the dimensions of the datasets

reg_data.shape
```

Out[10]: (1000000, 2)

In [11]:
```
auth_data.shape
```

Out[11]: (9601013, 2)

In [ ]:

In [ ]:

In [12]:
```
# посчитаем число уникальных id
# count the number of unique id

reg_data.uid.nunique()
```

Out[12]: 1000000

In [13]:
```
auth_data.uid.nunique()
```

Out[13]: 1000000

In [ ]:

In [ ]:

In [14]:
```
# соединяем два датасета
# теперь мы можем посчитать, спустя какое время после совершения первого события совершен заход пользовател
я в игру
# connect two datasets
# now we can calculate how long after the first event the user entered the game


df = auth_data.merge(reg_data, how = 'left', on = 'uid')
df
```

Out[14]:

| | auth_ts | uid | reg_ts |
|---|---|---|---|
| **0** | 1998-11-18 09:43:43 | 1 | 1998-11-18 09:43:43 |
| **1** | 1999-07-22 22:38:09 | 2 | 1999-07-22 22:38:09 |
| **2** | 1999-07-25 16:46:46 | 2 | 1999-07-22 22:38:09 |
| **3** | 1999-07-31 03:50:15 | 2 | 1999-07-22 22:38:09 |
| **4** | 1999-08-05 17:49:39 | 2 | 1999-07-22 22:38:09 |
| **...** | ... | ... | ... |
| **9601008** | 2020-09-23 15:13:54 | 1110618 | 2020-09-23 15:13:54 |
| **9601009** | 2020-09-23 15:14:46 | 1110619 | 2020-09-23 15:14:46 |
| **9601010** | 2020-09-23 15:15:39 | 1110620 | 2020-09-23 15:15:39 |
| **9601011** | 2020-09-23 15:16:31 | 1110621 | 2020-09-23 15:16:31 |
| **9601012** | 2020-09-23 15:17:24 | 1110622 | 2020-09-23 15:17:24 |

9601013 rows × 3 columns

Retention n-го дня - показывает сколько пользователей, пришедших в определенный день вернулось в продукт на n-ый день с момента своего прихода. Например: 3 сентября в продукт пришло 200 новых пользователей. 7 сентября из этих пользователей в продукт вернулось 20. То есть retention 4-го дня = 10%.

задача – написать функцию, которая будет считать retention игроков (по дням от даты регистрации игрока)

Retention of the n-th day - shows how many users who came on a certain day returned to the product on the n-th day from the date of arrival. For example: On September 3, 200 new users came to the product. September 7 of these 20 users returned to the product. That is, retention on the 4th day = 10%.

the task is to write a function that will count the retention of players (by days from the date of registration of the player)

In [15]:
```
# создадим новую колонку с разницей между моментом регистрации и заходом пользователя в игру
# возьмем только число дней
# create a new column with the difference between the moment of registration and the user entering the game
# take only the number of days

df['n_day'] = (df.auth_ts - df.reg_ts).dt.days
df
```

Out[15]:

| | auth_ts | uid | reg_ts | n_day |
|---|---|---|---|---|
| **0** | 1998-11-18 09:43:43 | 1 | 1998-11-18 09:43:43 | 0 |
| **1** | 1999-07-22 22:38:09 | 2 | 1999-07-22 22:38:09 | 0 |
| **2** | 1999-07-25 16:46:46 | 2 | 1999-07-22 22:38:09 | 2 |
| **3** | 1999-07-31 03:50:15 | 2 | 1999-07-22 22:38:09 | 8 |
| **4** | 1999-08-05 17:49:39 | 2 | 1999-07-22 22:38:09 | 13 |
| **...** | ... | ... | ... | ... |
| **9601008** | 2020-09-23 15:13:54 | 1110618 | 2020-09-23 15:13:54 | 0 |
| **9601009** | 2020-09-23 15:14:46 | 1110619 | 2020-09-23 15:14:46 | 0 |
| **9601010** | 2020-09-23 15:15:39 | 1110620 | 2020-09-23 15:15:39 | 0 |
| **9601011** | 2020-09-23 15:16:31 | 1110621 | 2020-09-23 15:16:31 | 0 |
| **9601012** | 2020-09-23 15:17:24 | 1110622 | 2020-09-23 15:17:24 | 0 |

9601013 rows × 4 columns

In [16]:
```
# найдем число пользователей, котрые зашли в приложение по дням, т.е. в 0 день после установки - зашли 100000
# на след день после установки - уже 40202 и т.д.
# find the number of users who logged into the application by day, i.e. on day 0 after installation - 100,000 logged in
# the next day after installation - already 40202, etc.

df_n_day = df.groupby('n_day') \
             .agg({'uid': 'count'})
df_n_day
```

Out[16]:

| n_day | uid |
|---|---|
| 0 | 1000000 |
| 1 | 40202 |
| 2 | 42991 |
| 3 | 49152 |
| 4 | 56070 |
| ... | ... |
| 7715 | 1 |
| 7719 | 1 |
| 7720 | 1 |
| 7726 | 1 |
| 7728 | 1 |

5916 rows × 1 columns

In [17]:
```
# сохраним в переменную число пользователей в нулевой день
# store in a variable the number of users on day zero

number_uid = df_n_day.iloc[0, 0]
number_uid
```

Out[17]: 1000000

In [18]:
```
# напишем функцию которая на вход принимает номер дня и возвращает Retention этого дня
# делим число вернувшихся пользователей в n-ый день на число пользователей нулевого дня
# умножаем на 100 - для выражения в процентах и округляем до двух знаков после запятой
# let's write a function that takes the number of the day as input and returns the Retention of this day
# divide the number of returning users on day n by the number of users on day zero
# multiply by 100 - to express as a percentage and round to two decimal places


def retention_n_day(day_number):
    retention = round(df_n_day.iloc[day_number, 0] / df_n_day.iloc[0, 0] * 100, 2)
    return print(f'Retention {day_number} дня равен {retention}%')
```

In [19]:
```
retention_n_day(10)
```

```
Retention 10 дня равен 5.11%
```

In [ ]:
```

```

Задание 2

Имеются результаты A/B теста, в котором двум группам пользователей предлагались различные наборы акционных предложений. Известно, что ARPU в тестовой группе выше на 5%, чем в контрольной. При этом в контрольной группе 1928 игроков из 202103 оказались платящими, а в тестовой – 1805 из 202667.

Какой набор предложений можно считать лучшим? Какие метрики стоит проанализировать для принятия правильного решения и как?

Task 2

There are results of an A / B test in which two groups of users were offered different sets of promotional offers. It is known that ARPU in the test group is 5% higher than in the control group. At the same time, in the control group, 1928 players out of 202103 turned out to be paying, and in the test group - 1805 out of 202667.

What set of proposals can be considered the best? What metrics should be analyzed to make the right decision and how?

In [20]:
```
# считаем данные
# counting data

df_2 = pd.read_csv('/home/jupyter-i-sharapova-17/final_project/Проект_1_Задание_2.csv', sep=';')
df_2
```

Out[20]:

| | user_id | revenue | testgroup |
|---|---|---|---|
| **0** | 1 | 0 | b |
| **1** | 2 | 0 | a |
| **2** | 3 | 0 | a |
| **3** | 4 | 0 | b |
| **4** | 5 | 0 | b |
| **...** | ... | ... | ... |
| **404765** | 404766 | 0 | a |
| **404766** | 404767 | 0 | b |
| **404767** | 404768 | 231 | a |
| **404768** | 404769 | 0 | a |
| **404769** | 404770 | 0 | b |

404770 rows × 3 columns

In [21]:
```
# посмотрим на число уникальных значений
# look at the number of unique values


df_2.nunique()
```

Out[21]:
```
user_id      404770
revenue        1477
testgroup         2
dtype: int64
```

In [22]:
```
# найдем общую сумму по колонке revenue (доход контрольной + доход тестовой групп)
# find the total amount in the revenue column (income of the control group + income of the test group)


r_sum = df_2.revenue.sum()
r_sum
```

Out[22]: 10557792

In [ ]:
```

```

In [23]:
```
# сгруппируем по тестгруппам и посчитаем доход для каждой группы в отдельности
# group by test groups and calculate the income for each group separately

r_df_2 = df_2.groupby(['testgroup'], as_index=False)\
    .agg({'revenue': 'sum'})
r_df_2
```

Out[23]:

| | testgroup | revenue |
|---|---|---|
| 0 | a | 5136189 |
| 1 | b | 5421603 |

In [24]:
```
# создадим калонку со значениями arpu, выразим в процентах
# create a column with arpu values, express as a percentage

r_df_2['arpu'] = (r_df_2.revenue / r_sum).mul(100)
r_df_2

# видим, что значения отличаются приблизительно на 2,7%
# we see that the values differ by approximately 2.7%
```

Out[24]:

| | testgroup | revenue | arpu |
|---|---|---|---|
| 0 | a | 5136189 | 48.648325 |
| 1 | b | 5421603 | 51.351675 |

In [25]:
```
# возьмемем столбцы revenue и testgroup
# take columns revenue and testgroup


df_2 = df_2[['revenue', 'testgroup']]
df_2
```

Out[25]:

| | revenue | testgroup |
|---|---|---|
| **0** | 0 | b |
| **1** | 0 | a |
| **2** | 0 | a |
| **3** | 0 | b |
| **4** | 0 | b |
| **...** | ... | ... |
| **404765** | 0 | a |
| **404766** | 0 | b |
| **404767** | 231 | a |
| **404768** | 0 | a |
| **404769** | 0 | b |

404770 rows × 2 columns

In [26]:
```
import seaborn as sns
```

In [27]:
```
# отберем только те строки, где доход больше нуля
# select only those rows where the income is greater than zero

ab_df_2 = df_2.query('revenue > 0')
ab_df_2
```

Out[27]:

| | revenue | testgroup |
|---|---|---|
| **72** | 351 | a |
| **160** | 3797 | b |
| **341** | 290 | a |
| **377** | 3768 | b |
| **385** | 250 | a |
| **...** | ... | ... |
| **404315** | 262 | a |
| **404525** | 3120 | b |
| **404543** | 369 | a |
| **404602** | 251 | a |
| **404767** | 231 | a |

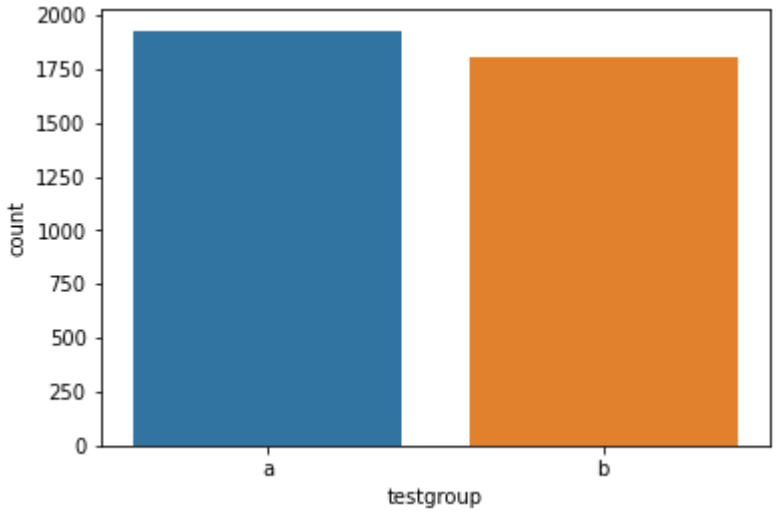
3733 rows × 2 columns

In [28]:
```
sns.countplot(ab_df_2.testgroup)

# видим, что число клиентов которые совершили покупку выше в контрольной группе
# we see that the number of customers who made a purchase is higher in the control group
```

```
/opt/tljh/user/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following va
riable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and pass
ing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f52c54327b8>
```

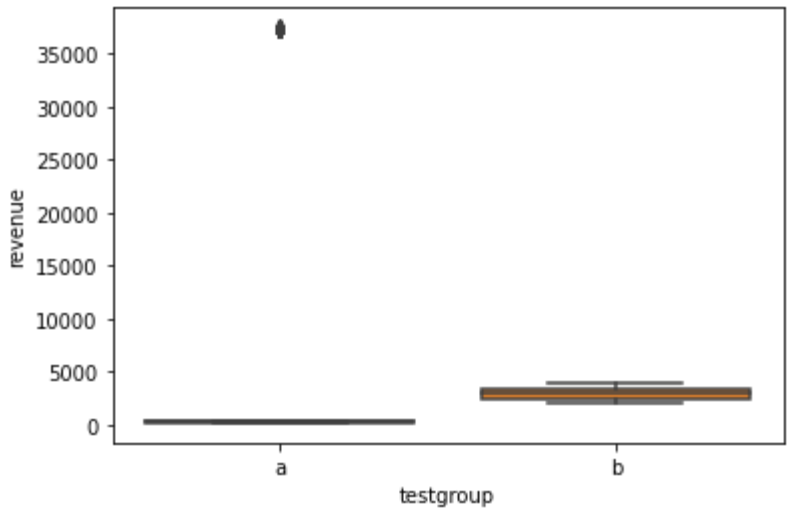


In [29]:
```
# построим барплоты для доходов тестгрупп
# build barplots for the income of test groups

sns.boxplot(x='testgroup', y='revenue', data=ab_df_2)

# непонятно, а здорово)))
# incomprehensible, but cool)))
```

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x7f52767f4d30>



In [30]:
```
# посмотрим на описательные статистики
# look at descriptive statistics


ab_df_2.describe()

# видим что максимальное значение в колонке revenue больше 37000,
# при этом 75 процентиль = 3126 (75 процентов всех покупок совершили на сумму 3126 и меньше)
# we see that the maximum value in the revenue column is more than 37000,
# while 75 percentile = 3126 (75 percent of all purchases were made in the amount of 3126 or less)
```

Out[30]:

| | revenue |
|---|---|
| **count** | 3733.000000 |
| **mean** | 2828.232521 |
| **std** | 6516.770722 |
| **min** | 200.000000 |
| **25%** | 309.000000 |
| **50%** | 2052.000000 |
| **75%** | 3126.000000 |
| **max** | 37433.000000 |

In [31]:
```
# отсортируем колонку revenue по убыванию
# sort the revenue column in descending order

ab_df_2.revenue.sort_values(ascending=False)

# неясно какое количсетво строк имеет revenue больше 3700
# it is not clear how many rows revenue has more than 3700
```

Out[31]:
```
101861    37433
196601    37407
65077     37394
21585     37385
142804    37379
          ...
246509      200
119128      200
393140      200
104975      200
230062      200
Name: revenue, Length: 3733, dtype: int64
```

In [32]:
```
# отберем строки, значение revenue которых более 35000
# select rows whose revenue value is more than 35000

ab_df_2.query('revenue > 35000')

# видим, что их 123
# ПРИМЕЧАТЕЛЬНО, что разница между платящими игроками тестгрупп тоже = 123
# (При этом в контрольной группе 1928 игроков из 202103 оказались платящими, а в тестовой – 1805 из 202667)

# можно предположить, что что-то повлияло на 123 пользователей совершить дорогостоящие покупки
# (т.н. выбросы на боксплоте) - возможно покупка годового абоненмента и т.д.
# но вопрос почему только контрольная группа? Имеющиеся данные не дают это понять

# we see that there are 123 of them
# NOTICE that the difference between the paying players of the test groups is also = 123
# (At the same time, in the control group, 1928 players out of 202103 turned out to be paying, and in the t
est group - 1805 out of 202667)

# it can be assumed that something influenced 123 users to make expensive purchases
# (so-called emissions on the boxplot) - it is possible to purchase an annual subscription, etc.
# but the question is why only the control group? The available data do not make it clear
```

Out[32]:

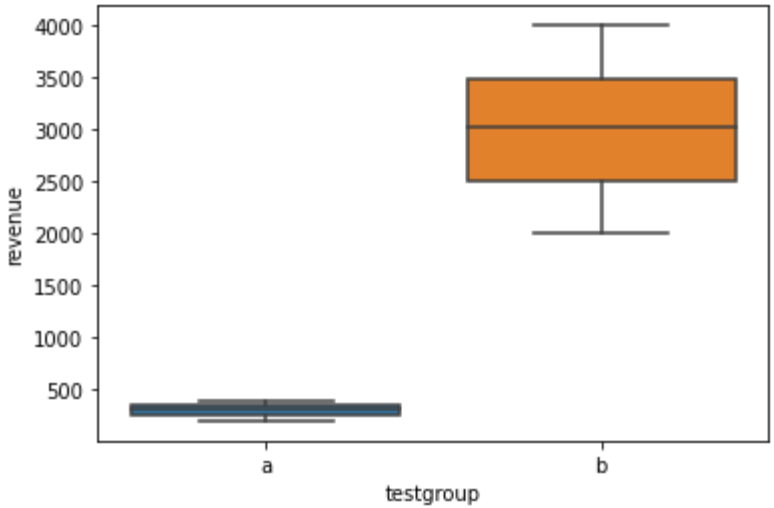
| | revenue | testgroup |
|---|---|---|
| **416** | 37324 | a |
| **4428** | 37304 | a |
| **5777** | 37315 | a |
| **8065** | 37307 | a |
| **8743** | 37334 | a |
| **...** | ... | ... |
| **360696** | 37347 | a |
| **378668** | 37297 | a |
| **389341** | 37298 | a |
| **394580** | 37302 | a |
| **399294** | 37321 | a |

123 rows × 2 columns

In [33]:
```
# попробуем построить график без этих 123 строк
# let's try to build a graph without these 123 lines

sns.boxplot(x='testgroup', y='revenue', data=ab_df_2.query('revenue < 35000'))
```

Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5276c825c0>



In [60]:
```
# посмотрим на разницу средних значений тестгрупп
# look at the difference in the average values of the test groups

data = ab_df_2.query('revenue < 35000')
data
```

Out[60]:

| | revenue | testgroup |
|---|---|---|
| **72** | 351 | a |
| **160** | 3797 | b |
| **341** | 290 | a |
| **377** | 3768 | b |
| **385** | 250 | a |
| **...** | ... | ... |
| **404315** | 262 | a |
| **404525** | 3120 | b |
| **404543** | 369 | a |
| **404602** | 251 | a |
| **404767** | 231 | a |

3610 rows × 2 columns

In [62]:
```
data.groupby(['testgroup'])\
    .agg({'revenue': 'mean'})

# видим что разница более чем на 2500
# we see that the difference is more than 2500
```

Out[62]:

| testgroup | revenue |
|---|---|
| a | 302.458172 |
| b | 3003.658172 |

In [63]:
```
# воспользуемся для сравнения средних t-тестом,
# хотя по боксплоту видно, что даже значения 25 и 75 процентилей сильно отдалены
# предположим, что нулевая гипотеза верна,

# we will use the t-test to compare the means,
# although the boxplot shows that even the values of the 25th and 75th percentiles are very far away
# suppose the null hypothesis is true,


stats.ttest_ind(data.query('testgroup == "a"').revenue,
                data.query('testgroup == "b"').revenue)

# мы применили тест Стьюдента который сранил два средних значения по нашей выборке и  pvalue=0.0
# если бы была верна нулевая гипотеза и средние не различались,
# то вероятность получить такие или более явные различия составили бы 0%
# альтернативную гипотезу, о том что средние статзначимо отличаются  принимаем
# тест показал нам, что в нашем эксперементе мы смогли получить стастически значимые различия
# и в тестовой группе набор предложений можно считать лучшим, чем в контрольной,
# если не принимать во внимание выбросы, сформированные 123 пользователями
# в контрольной группе с доходом от  них по 37 тысяч

# we applied a Student's t-test that compared the two means of our sample and pvalue=0.0
# if the null hypothesis were true and the means did not differ,
# then the probability of getting such or more pronounced differences would be 0%
# alternative hypothesis that the means are statistically significantly different is accepted
# the test showed us that in our experiment we were able to obtain statistically significant differences
# and in the test group, the set of sentences can be considered better than in the control group,
# if we do not take into account the emissions generated by 123 users
# in the control group with an income of 37 thousand
```

Out[63]:
```
Ttest_indResult(statistic=-199.39722533995328, pvalue=0.0)
```

In [ ]:

In [ ]:

Задание 3

В игре Plants & Gardens каждый месяц проводятся тематические события, ограниченные по времени. В них игроки могут получить уникальные предметы для сада и персонажей, дополнительные монеты или бонусы. Для получения награды требуется пройти ряд уровней за определенное время. С помощью каких метрик можно оценить результаты последнего прошедшего события?

Предположим, в другом событии мы усложнили механику событий так, что при каждой неудачной попытке выполнения уровня игрок будет откатываться на несколько уровней назад. Изменится ли набор метрик оценки результата? Если да, то как?

Task 3

Plants & Gardens has a limited-time themed event every month. In them, players can get unique items for the garden and characters, additional coins or bonuses. To receive a reward, you need to complete a series of levels in a certain time. What metrics can be used to evaluate the results of the last past event?

Suppose, in another event, we have complicated the mechanics of events so that for each unsuccessful attempt to complete the level, the player will roll back several levels. Will the set of performance evaluation metrics change? If so, how?

Тематические события, ограниченные по времени направлены на:

1 увеличение времени, проводимого в игре и как следствие поддержание интереса

2 продления ежемесячных платных подписок

3 дополнительных покупок, совершаемых пользователем

Пояснения по каждому пункту:

1 могу предположить, что метрика "среднее время пользователя" в игре увеличится во время тематических событий, к примеру в игре World of Warcraft такие события как ярмарка новолуния, любовная лихорадка, новогодние квесты, привлекают пользователей дополнительными квестами и помимо основных событий еще необходимо время для выполнения тематических

2 также на примере игры World of Warcraft, предположим пользователь после выхода отдона прокачал на максимум своего героя, и может не проплачивать еще месяц, а при наличии тематического события может возобновить подписку. Метрика среднее количество продленных подписок в период тематического события возможно покажет рост

3 некоторые тематические события предполагают дополнительные покупки в игре, какие-то монеты или другие ресурсы для получения уникальных предметов для персонажей. Предположу, что метрика "число дополнительных покупок" в период тематических событий увеличится

При усложненной механике событий предположу, что следует дополнительно посмотреть на отношение часла игроков продолжающих после неудачной попытки к числу игроков переставших продолжать. Т.е. посмотреть, какой процент игроков самые "упоротые/упертые")

Thematic limited-time events are aimed at:

1 increase the time spent in the game and, as a result, the maintenance of interest

2 renewals of monthly paid subscriptions

3 additional purchases made by the user

Explanations for each item:

1 I can assume that the "average user time" metric in the game will increase during thematic events, for example, in the World of Warcraft game, events such as the dark moon fair, love fever, New Year's quests attract users with additional quests and in addition to the main events, time is still needed for implementation of thematic

2 also on the example of the game World of Warcraft, suppose the user, after the release of the otdon, pumped his hero to the maximum, and may not pay for another month, and if there is a thematic event, he can renew the subscription. The metric average number of renewed subscriptions during the period of the thematic event will probably show an increase

3 some thematic events involve additional purchases in the game, some coins or other resources to get unique items for the characters. I will assume that the "number of additional purchases" metric will increase during the period of thematic events

With the complicated mechanics of events, I suppose that we should additionally look at the ratio of the hour of players who continue after an unsuccessful attempt to the number of players who stopped continuing. Those, see what percentage of players are the most "stubborn / stubborn")

In [ ]: