МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБО6РОСТРОЕНИЯ»

КАФЕДРА № 43

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

Старший преподаватель		С.А. Рогачев
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

Знакомство с Jupyter Notebook

по курсу: Основы машинного обучения

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4134к		Д. В. Самарин
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

Цель работы:

Знакомство со средами Jupyter Notebook и Google Colaboratory, а также библиотеками Pandas и matplotlib

Постановка задачи

1. Выполнить задания, приведенные ниже, и сдать их на проверку.
2. Подготовить отчет и загрузить его в репозиторий.
3. Убедиться в успешном прохождении тестов в репозитории.
4. Защитить работу.
5. Загрузить отчет в личный кабинет.

Ход работы:

Задание 1:

Часть 1. GitHub и ноутбуки Jupyter, Google Colaboratory

Начать необходимо с просмотра этого ноутбука, чтобы ознакомиться с ноутбуками Jupyter и средой Google Colaboratory (сокращенно - Colab). Открыть ознакомительный ноутбук в Colab можно перейдя по ссылке `https://colab.research.google.com/github/<ORGANIZATION>/<REPOSITORY>/blob/main/introduction_and_overview.ipynb`, где `<ORGANIZATION>` необходимо заменить на название организации на GitHub, используемой в этом курсе (это упомянутое выше "сокращенное название курса"), а `<REPOSITORY>` - заменить на название личного репозитория студента (`<assignment_name>-<username>` выше). Следуйте инструкциям в ознакомительном ноутбуке, чтобы настроить свою учетную запись Google Colaboratory.

Измените код в следующей ячейке, чтобы указать, что вы изучили ознакомительный ноутбук, прочитали о работе ноутбуков Jupyter/Colaboratory, настроили свою учетную запись Google Colaboratory и настроили свою учетную запись на GitHub. Вам нужно изменить каждую переменную с `False` на `True`, чтобы показать, какие задачи вы выполнили (по самоотчету). Также обновите переменную `github_username`, чтобы указать свое имя пользователя на GitHub.

```
### BEGIN YOUR CODE
#I have read through the Introduction and Overview notebook
READ_INTRODUCTION = True

#I understand (at a high level) what Jupyter notebooks are and how to read and
#interact with them (or I have been in touch with the course instructor to ask for help)
LEARNED_ABOUT_JUPYTER = True

#I've created (or already have) a Google account and can access Google
#Colaboratory under my own account
ACCESS_COLABORATORY = True

#I've created a GitHub account
CREATED_GITHUB_ACCOUNT = True
github_username = 'dYGamma'

#My info
my_name = 'Dmitry'
### END YOUR CODE
```

Часть 2. Базовый вывод информации

Функция `print` может использоваться как для отображения результатов вычислений, так и для отладки кода. Один из базовых подходов к отладке кода в Jupyter-ноутбуке — периодически выводить диагностические сообщения с помощью `print`, чтобы понять, что происходит в конкретном месте кода.

Самый простой способ использования функции `print` — вызвать `print(...)`, заменив `...` на текст, который необходимо вывести. Текст должен быть заключен в одинарные (') или двойные (") кавычки, вот так:

```
print('Hello, world!')
```

Измените код в следующей ячейке, чтобы вывести приветствие себе (например, `"Hello, Noname!"` или что-то в этом роде, заменив `"Noname"` на ваше собственное имя, заданное в предыдущей ячейке). Проверьте результат, нажав shift + enter, чтобы выполнить код в ячейке. Приветствие появится под ячейкой с кодом.

```
### BEGIN YOUR CODE
print('Hello, Dmitry')
### END YOUR CODE
```

```
Hello, Dmitry
```

Объявление функций

Внесите небольшое изменение в программу "Привет, мир!" выше. Вместо того чтобы выводить приветствие, напишите функцию, которая принимает ваше имя `name` в качестве входного параметра и возвращает строку `'Hello, <name>!'`, заменяя `<name>` на то, что указал вызвавший функцию пользователь.

```
def greet(name):
  ### BEGIN YOUR CODE
  return f'Hello, {name}!'
  ### END YOUR CODE
```

Задание 2:

Part 1. A quest to find your task

Let's start by importing `numpy` library. We will need it later on to do some maths. We also need `matplotlib.pyplot` to visualize the results of our calculations.

```
import numpy as np
from matplotlib import pyplot as plt
```

Follow the link to a Google Sheet with a list of students. Locate your name on the list and take note of the corresponding `Student ID` in the first column. Fill it in the cell below and run the cell. If you can't find yourself on the list, consult your course instructor.

```
### BEGIN YOUR CODE

Student_ID = 14

### END YOUR CODE
```

Now run the next cell. It will print a function number for you.

```
task_id = None if Student_ID is None else Student_ID % 25 if Student_ID % 25 > 0 else 25
print(f"Please, choose a mathematical function No {task_id} below")
```

```
Please, choose a mathematical function No 14 below
```

In the list of mathematical functions presented above y, or more correctly speaking $y(x)$, is a dependent variable produced by calculating a mathematical function. a, b, c, d are scalar function parameters and x is an independent variable.

Now that you have selected a function, write it down in a cell below using LaTeX and run the cell to render it

$$\begin{cases} y = \sqrt{1 - (|x| - 1)^2} \\ y = \arccos(1 - |x|) - \pi \end{cases}$$

Part 2. Make python do the maths

Write a python function that calculates the mathematical function $y(x)$ given scalar parameters a, b, c, d (if aplicable) and a list of values of independent variable x. You can find mathematical functions available in the `numpy` library here.

An example for function $y(x) = a \sin^2 x + b \log_c x$ might look like this:

```
def my_function(x,a,b,c,d):
    return a * np.sin(x) ** 2 + b * np.log(x) / np.log(c)
```

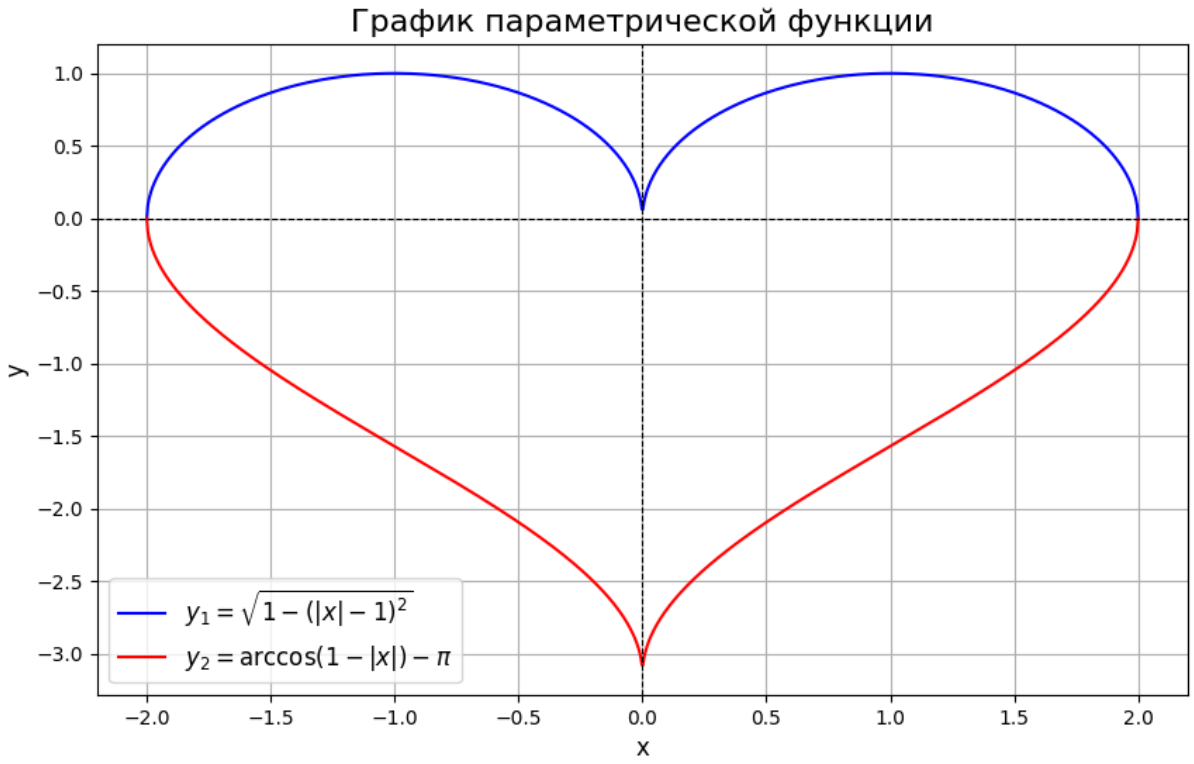
```
import numpy as np
from matplotlib import pyplot as plt
def my_function(x):
  ### BEGIN YOUR CODE
  y1 = np.sqrt(1 - (np.abs(x) - 1) ** 2)
  y2 = np.arccos(1 - np.abs(x)) - np.pi
  return y1, y2
  ### END YOUR CODE
```

Set some values for paramters a, b, c, d, and define a range for x variable:

```
### BEGIN YOUR CODE


x = np.linspace(-2, 2, 1000)

### END YOUR CODE
```

With all prerequesities ready we calculate values for the function $y(x)$. Now we can finally create a plot of our function. Note that you will most likely have to change values for a, b, c, d and x in the cell above in order to produce a nice looking plot below.

```
y1, y2 = my_function(x)

### BEGIN YOUR CODE

y1 = np.where((1 - (np.abs(x) - 1) ** 2) >= 0, y1, np.nan)

# Построение графиков
plt.figure(figsize=(10, 6))

# График первой части функции
plt.plot(x, y1, label=r'$y_1 = \sqrt{1 - (|x| - 1)^2}$', color='blue')

# График второй части функции
plt.plot(x, y2, label=r'$y_2 = \arccos(1 - |x|) - \pi$', color='red')

# Оформление графика
plt.axhline(0, color='black', linewidth=0.8, linestyle='--')  # Линия y = 0
plt.axvline(0, color='black', linewidth=0.8, linestyle='--')  # Линия x = 0
plt.legend(fontsize=12)
plt.title("График параметрической функции", fontsize=16)
plt.xlabel("x", fontsize=12)
plt.ylabel("y", fontsize=12)
plt.grid(True)

# Показ графика
plt.show()

### END YOUR CODE
```



Задание 3:

Задачи

1. Определить номер варианта

Перейдите по ссылке из личного кабинета на Google Таблицу со списком студентов. Найдите свое ФИО в списке и запомните соответствующий порядковый номер (поле № п/п) в первом столбце. Заполните его в ячейке ниже и выполните ячейку. Если вы не можете найти себя в списке, обратитесь к своему преподавателю.

```
### BEGIN YOUR CODE

Student_ID = 14

### END YOUR CODE
```

Теперь выполните следующую ячейку. Она вычислит номер задания и выведет его.

```
datasets = [('Chipotle','https://raw.githubusercontent.com/justmarkham/DAT8/master/data/chipotle.tsv'), ('US Air Carrier ma

dataset_id = None if Student_ID is None else Student_ID % 3
if dataset_id is None:
    print("ОШИБКА! Не указан порядковый номер студента в списке группы.")
else:
    print(f"Датасет '{datasets[dataset_id][0]}' доступен по следующей ссылке: {datasets[dataset_id][1]}")
    print(f"В заданиях ниже, где нужно выбрать вопрос, всегда выбирайте вопрос № {dataset_id+1}")
```

```
Датасет 'Open Food Facts' доступен по следующей ссылке: https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/mai
n/data/en.openfoodfacts.org.products.tsv.tar.bz2
В заданиях ниже, где нужно выбрать вопрос, всегда выбирайте вопрос № 3
```

Скачайте датасет с помощью команды `!wget <dataset_url>`, где `<dataset_url>` необходимо заменить на ссылку на датасет, появившуюся после выполнения предыдущей ячейки. При необходимости разархивируйте датасет, используя команды `!unzip`, `!tar` и др.

Примечание: в Jupyter-ноутбуке можно использовать любые команды командного интерпретатора bash. Для этого необходимо поставить в ячейке с кодом восклицательный знак `!`, после которого записать команду bash со всеми необходимыми аргументами. Результат выполнения этой команды bash будет возвращен в Jupyter и его можно использовать в коде на Python.

In [3]:

```
### BEGIN YOUR CODE

!wget https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/en.openfoodfacts.org.products.tsv.tar.bz2

# !unzip ...
!tar -xvjf dataset.tar.bz2
# !gunzip ...

### END YOUR CODE
```

```
--2024-12-25 11:40:15--  https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/en.openfoodfacts.org.prod
ucts.tsv.tar.bz2
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 75977297 (72M) [application/octet-stream]
Saving to: ‘dataset.tar.bz2’

dataset.tar.bz2     100%[===================>]  72.46M   232MB/s    in 0.3s

2024-12-25 11:40:16 (232 MB/s) - ‘dataset.tar.bz2’ saved [75977297/75977297]

._en.openfoodfacts.org.products.tsv
en.openfoodfacts.org.products.tsv
```

2. Загрузите датасет в pandas.DataFrame, сохраните его в переменной `df`. Сконвертируйте названия столбцов в нижний регистр

In [4]:

```
import pandas as pd

df = pd.read_csv("en.openfoodfacts.org.products.tsv", sep='\t', low_memory=False)
# Преобразование названий столбцов в нижний регистр
df.columns = df.columns.str.lower()
# Вывод первых строк для проверки
print(df.head())
```

```
          code                                               url  \
```

```
0  0000000003087  http://world-en.openfoodfacts.org/product/0000...
1  0000000004530  http://world-en.openfoodfacts.org/product/0000...
2  0000000004559  http://world-en.openfoodfacts.org/product/0000...
3  0000000016087  http://world-en.openfoodfacts.org/product/0000...
4  0000000016094  http://world-en.openfoodfacts.org/product/0000...

                     creator   created_t      created_datetime  \
0  openfoodfacts-contributors  1474103866  2016-09-17T09:17:46Z
1             usda-ndb-import  1489069957  2017-03-09T14:32:37Z
2             usda-ndb-import  1489069957  2017-03-09T14:32:37Z
3             usda-ndb-import  1489055731  2017-03-09T10:35:31Z
4             usda-ndb-import  1489055653  2017-03-09T10:34:13Z

  last_modified_t last_modified_datetime                  product_name  \
0      1474103893   2016-09-17T09:18:13Z             Farine de blé noir
1      1489069957   2017-03-09T14:32:37Z  Banana Chips Sweetened (Whole)
2      1489069957   2017-03-09T14:32:37Z                        Peanuts
3      1489055731   2017-03-09T10:35:31Z         Organic Salted Nut Mix
4      1489055653   2017-03-09T10:34:13Z                Organic Polenta

  generic_name quantity  ... fruits-vegetables-nuts_100g  \
0          NaN      1kg  ...                         NaN
1          NaN      NaN  ...                         NaN
2          NaN      NaN  ...                         NaN
3          NaN      NaN  ...                         NaN
4          NaN      NaN  ...                         NaN

  fruits-vegetables-nuts-estimate_100g collagen-meat-protein-ratio_100g  \
0                                  NaN                               NaN
1                                  NaN                               NaN
2                                  NaN                               NaN
3                                  NaN                               NaN
4                                  NaN                               NaN

  cocoa_100g chlorophyl_100g carbon-footprint_100g nutrition-score-fr_100g  \
0        NaN             NaN                   NaN                     NaN
1        NaN             NaN                   NaN                    14.0
2        NaN             NaN                   NaN                     0.0
3        NaN             NaN                   NaN                    12.0
4        NaN             NaN                   NaN                     NaN

  nutrition-score-uk_100g glycemic-index_100g water-hardness_100g
0                     NaN                NaN                 NaN
1                    14.0                NaN                 NaN
2                     0.0                NaN                 NaN
3                    12.0                NaN                 NaN
4                     NaN                NaN                 NaN

[5 rows x 163 columns]
```

3. Какие столбцы присутствуют в наборе данных? (0.25 балла)

[5]:
```
columns = df.columns.tolist()

print(columns)
```

```
['code', 'url', 'creator', 'created_t', 'created_datetime', 'last_modified_t', 'last_modified_datetime', 'product_name', 'gen
eric_name', 'quantity', 'packaging', 'packaging_tags', 'brands', 'brands_tags', 'categories', 'categories_tags', 'categories_
en', 'origins', 'origins_tags', 'manufacturing_places', 'manufacturing_places_tags', 'labels', 'labels_tags', 'labels_en', 'e
mb_codes', 'emb_codes_tags', 'first_packaging_code_geo', 'cities', 'cities_tags', 'purchase_places', 'stores', 'countries',
'countries_tags', 'countries_en', 'ingredients_text', 'allergens', 'allergens_en', 'traces', 'traces_tags', 'traces_en', 'ser
ving_size', 'no_nutriments', 'additives_n', 'additives', 'additives_tags', 'additives_en', 'ingredients_from_palm_oil_n', 'in
gredients_from_palm_oil', 'ingredients_from_palm_oil_tags', 'ingredients_that_may_be_from_palm_oil_n', 'ingredients_that_may_
be_from_palm_oil', 'ingredients_that_may_be_from_palm_oil_tags', 'nutrition_grade_uk', 'nutrition_grade_fr', 'pnns_groups_1',
'pnns_groups_2', 'states', 'states_tags', 'states_en', 'main_category', 'main_category_en', 'image_url', 'image_small_url',
'energy_100g', 'energy-from-fat_100g', 'fat_100g', 'saturated-fat_100g', '-butyric-acid_100g', '-caproic-acid_100g', '-capryl
ic-acid_100g', '-capric-acid_100g', '-lauric-acid_100g', '-myristic-acid_100g', '-palmitic-acid_100g', '-stearic-acid_100g',
'-arachidic-acid_100g', '-behenic-acid_100g', '-lignoceric-acid_100g', '-cerotic-acid_100g', '-montanic-acid_100g', '-melissi
c-acid_100g', 'monounsaturated-fat_100g', 'polyunsaturated-fat_100g', 'omega-3-fat_100g', '-alpha-linolenic-acid_100g', '-eic
osapentaenoic-acid_100g', '-docosahexaenoic-acid_100g', 'omega-6-fat_100g', '-linoleic-acid_100g', '-arachidonic-acid_100g',
'-gamma-linolenic-acid_100g', '-dihomo-gamma-linolenic-acid_100g', 'omega-9-fat_100g', '-oleic-acid_100g', '-elaidic-acid_100
g', '-gondoic-acid_100g', '-mead-acid_100g', '-erucic-acid_100g', '-nervonic-acid_100g', 'trans-fat_100g', 'cholesterol_100
g', 'carbohydrates_100g', 'sugars_100g', '-sucrose_100g', '-glucose_100g', '-fructose_100g', '-lactose_100g', '-maltose_100
g', '-maltodextrins_100g', 'starch_100g', 'polyols_100g', 'fiber_100g', 'proteins_100g', 'casein_100g', 'serum-proteins_100
g', 'nucleotides_100g', 'salt_100g', 'sodium_100g', 'alcohol_100g', 'vitamin-a_100g', 'beta-carotene_100g', 'vitamin-d_100g',
'vitamin-e_100g', 'vitamin-k_100g', 'vitamin-c_100g', 'vitamin-b1_100g', 'vitamin-b2_100g', 'vitamin-pp_100g', 'vitamin-b6_10
0g', 'vitamin-b9_100g', 'folates_100g', 'vitamin-b12_100g', 'biotin_100g', 'pantothenic-acid_100g', 'silica_100g', 'bicarbona
te_100g', 'potassium_100g', 'chloride_100g', 'calcium_100g', 'phosphorus_100g', 'iron_100g', 'magnesium_100g', 'zinc_100g',
'copper_100g', 'manganese_100g', 'fluoride_100g', 'selenium_100g', 'chromium_100g', 'molybdenum_100g', 'iodine_100g', 'caffei
ne_100g', 'taurine_100g', 'ph_100g', 'fruits-vegetables-nuts_100g', 'fruits-vegetables-nuts-estimate_100g', 'collagen-meat-pr
otein-ratio_100g', 'cocoa_100g', 'chlorophyl_100g', 'carbon-footprint_100g', 'nutrition-score-fr_100g', 'nutrition-score-uk_1
00g', 'glycemic-index_100g', 'water-hardness_100g']
```

4. Ответьте на вопрос и сохраните ответ в переменной `answer1` (0.25 балла)

Вопросы:

1. Какое блюдо (`item_name`) заказывали чаще всего?
2. Сколько авиаперевозчиков (`carrier`) представлены в датасете?
3. По скольки продуктам в датасете имеется информация о содержании аллергенов (`allergens`)?

[9]:
```
# 3. По скольки продуктам в датасете имеется информация о содержании аллергенов (allergens)?
products_with_allergens = df['allergens'].notnull().sum()

answer1 = {
    "products_with_allergens": products_with_allergens
}

print(answer1)
```

```
{'products_with_allergens': 37176}
```

5. Ответьте на вопрос и сохраните ответ в переменной `answer2` (0.5 балла)

Вопросы:

1. Сколько всего было заказов блюда, название которого сохранено в `answer1` ?
2. Посчитайте общие суммарные количества перевезенных пассажиров (`passangers`), фунтов груза (`freight`) и почты (`mail`) на маршруте из Великобритании (UK) в США (US). В `answer2` запишите максимальное из трех получившихся чисел.
3. Сколько всего продуктов, относящихся к категории "молочные" (`Dairies,Milks`), с заполненным названием?

[10]:
```
# Отфильтровываем продукты категории "молочные" и проверяем, есть ли у них название
dairy_products = df[df['categories'].str.contains('Dairies|Milks', case=False, na=False)]
dairy_with_name = dairy_products[dairy_products['product_name'].notna()]

answer2 = dairy_with_name.shape[0]

print(answer2)
```

```
2105
```

6. Ответьте на вопрос и сохраните ответ в переменной `answer3` (0.5 балла)

Вопросы:

1. Какой доход получила сеть Chipotle Mexican Grill на заказах, попавших в датасет?
2. Какой авиаперевозчик (`unique_carrier_name`) перевез больше всего груза (`mail` + `freight`)?
3. Как называется продукт категории `Fats` с максимальной жирностью, не превышающей 30 г на 100 г продукта?

[11]:
```
# Отфильтровываем продукты категории "Fats" и жирность не более 30 г
fats_products = df[df['categories'].str.contains('Fats', case=False, na=False)]
fats_below_30 = fats_products[fats_products['fat_100g'] <= 30]

# Находим продукт с максимальной жирностью
max_fat_product = fats_below_30.loc[fats_below_30['fat_100g'].idxmax()]

answer3 = max_fat_product['product_name']

print(answer3)
```

```
Margarine a tartiner light a l'huile de tournesol
```

7. Ответьте на вопрос и сохраните ответ в переменной `answer4` (0.5 балла)

Вопросы:

1. Каков средний доход с одного заказа?
2. Какое максимальное количество пассажиров одна авиакомпания смогла перевезти из США в другие страны за все время?
3. Какова энергетическая ценность в кДж продукта из России (`category_en`) имеющего максимальное содержание холистерина?

[n [16]:
```
# Отфильтровываем продукты, произведенные в России
russian_products = df[df['countries_en'].str.contains('Russia', case=False, na=False)]

# Находим продукт с максимальным содержанием холестерина
max_cholesterol_product = russian_products.loc[russian_products['cholesterol_100g'].idxmax()]

answer4 = max_cholesterol_product['energy_100g']

print(answer4)
```

```
2319.0
```

8. Ответьте на вопрос и сохраните ответ в переменной `answer5` (1 балл)

Вопросы:

1. Сколько раз был заказан самый популярный напиток (Coke, Sprite, Mountain Dew и т.п.)?
2. Между какими двумя городами было перевезено наибольшее количество пассажиров? Учтите оба направления. Ответ запишите в виде списка из двух строк.
3. Привести названия всех аллергенов к нижнему регистру. Какой аллерген встречается в продуктах чаще всего?

[18]:
```
# Приводим все аллергены в столбце allergens к нижнему регистру
df['allergens'] = df['allergens'].str.lower()

# Разделяем аллергены по запятой, если они встречаются в виде списка
allergens_list = df['allergens'].dropna().str.split(',')

# Создаем один общий список всех аллергенов
all_allergens = [allergen.strip() for sublist in allergens_list for allergen in sublist]

# Находим наиболее частый аллерген
most_common_allergen = pd.Series(all_allergens).mode()[0]

answer5 = most_common_allergen

print(answer5)
```

```
lait
```

9. Ответьте на вопрос и сохраните ответ в переменной `answer6` (1 балл)

Вопросы:

1. Какой суммарный доход принесли напитки в заказах вегетарианцев?
2. Для пары городов из предыдущего вопроса найдите 3 авиакомпании, которые перевезли больше всего пассажиров. Посчитайте, какой процент от общего пассажиропотока между этими городами перевезла каждая из трех авиакомпаний. В `answer6` запишите найденные проценты в виде списка из трех чисел, округлив их до двух знаков после запятой.
3. Найти самый опасный продукт, содержащий наибольшее количество аллергенов.

19]:
```
# Разделяем аллергены по запятой и подсчитываем их количество
df['num_allergens'] = df['allergens'].dropna().str.split(',').apply(len)

# Находим продукт с наибольшим количеством аллергенов
most_dangerous_product = df.loc[df['num_allergens'].idxmax(), 'product_name']

answer6 = most_dangerous_product

print(answer6)
```

```
Nos toasts chauds
```

10. Ответьте на вопрос и сохраните ответ в переменной `answer7` (1 балл)

Вопросы:

1. Сколько было сделано вегетарианских заказов? Заказ не считается вегетарианским, если в нем были не вегетарианские блюда.
2. Для каждой страны найдите процент международного пассажиропотока (относительно США), используя общее количество пассажиров на рейсах класса F. В `answer7` запишите название страны с третьим по величине пассажиропотоком в/из США.
3. Переведите названия групп продуктов (`pnns_groups_1` , `pnns_groups_2`) в нижний регистр. В переменную `answer7` запишите список, содержащий три элемента: название группы продуктов 1, название группы продуктов 2 и среднее количество пищевых волокон (`fiber`) для седьмой по насыщенности пищевыми волокнами группы продуктов.

[35]:
```
# Переводим названия групп продуктов в нижний регистр с использованием .loc
df.loc[:, 'pnns_groups_1'] = df['pnns_groups_1'].str.lower()
df.loc[:, 'pnns_groups_2'] = df['pnns_groups_2'].str.lower()

# Считаем среднее количество пищевых волокон для каждой группы
fiber_means = df.groupby('pnns_groups_1')['fiber_100g'].mean()

# Находим седьмую по насыщенности пищевыми волокнами группу
fiber_means_sorted = fiber_means.sort_values(ascending=False)
seventh_group = fiber_means_sorted.index[6]
seventh_group_mean_fiber = fiber_means_sorted.iloc[6]

answer7 = [df['pnns_groups_1'].iloc[0], df['pnns_groups_2'].iloc[0], seventh_group_mean_fiber]

print(answer7)
```

```
['unknown', 'unknown', 3.450413952342162]
```

11. Ответьте на вопрос и сохраните ответ в переменной `answer8` (1 балл)

Вопросы:

1. Какой соус или дополнительный ингредиент по выбору (`choice_description`) чаще всего берут вместе с бурито с курицей (Chicken Burrito)?
2. В каком месяце пассажиропоток между городами, записанными в переменную `answer5` , был максимальным?
3. Какое название у группы продуктов `pnns_groups_2` , явояющейся наиболее сбалансированной с точки зрения среднего содержания калорий, жиров и углеводов? Под "сбалансированной" понимать близость БЖУ к пропорции 1:1:4.

[37]:
```
# Вычисляем отклонения от пропорции 1:1:4
def calculate_balance(row):
    # Проверяем, чтобы значения калорий не были нулевыми
    if row['energy_100g'] == 0:
        return np.nan  # Возвращаем NaN, если калории равны нулю
    # Нормируем макроэлементы на 100 г продукта
    fats_ratio = row['fat_100g'] / row['energy_100g']
    carbs_ratio = row['carbohydrates_100g'] / row['energy_100g']
    # Идеальная пропорция (1:1:4) для жиров, углеводов и калорий
    ideal_fats_ratio = 1 / 6  # 1/6 от калорий
    ideal_carbs_ratio = 4 / 6  # 4/6 от калорий
    # Считаем абсолютные отклонения от идеальной пропорции
    fat_deviation = abs(fats_ratio - ideal_fats_ratio)
    carbs_deviation = abs(carbs_ratio - ideal_carbs_ratio)
    return fat_deviation + carbs_deviation

# Применяем функцию для каждой строки датафрейма с использованием .loc
df.loc[:, 'balance_deviation'] = df.apply(calculate_balance, axis=1)

# Убираем строки с NaN значениями в отклонениях
df_cleaned = df.dropna(subset=['balance_deviation'])

# Считаем среднее отклонение для каждой группы продуктов
grouped_balance = df_cleaned.groupby('pnns_groups_2')['balance_deviation'].mean()

# Находим группу с минимальным отклонением
best_group = grouped_balance.idxmin()


answer8 = best_group

print(answer8)
```

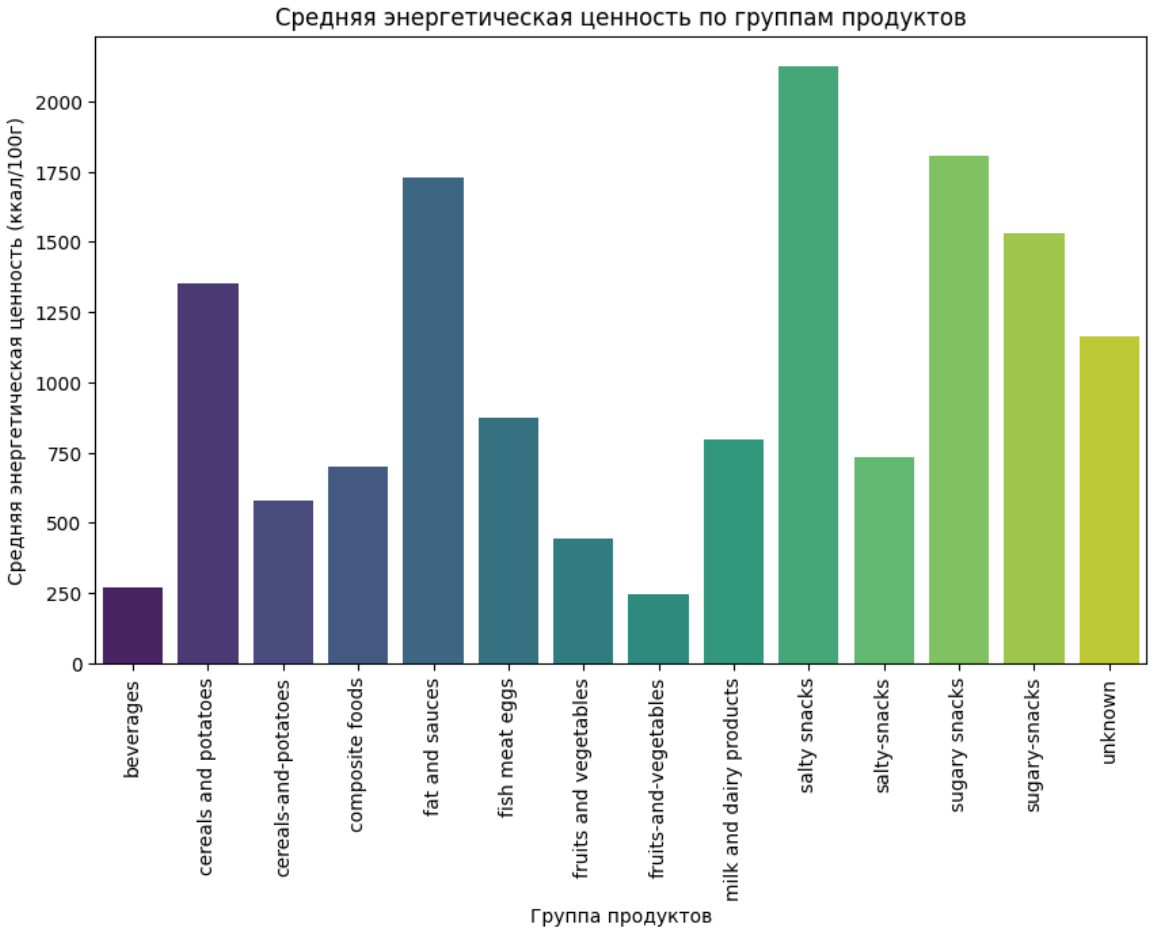
```
fruit nectars
```

12. Визуализируйте данные в соответствии с заданием (1 балл)

1. Построить гистограмму распределения общей стоимости заказов. Найти и отметить на графике средний чек и медианную стоимость заказа.
2. Постройте стековую столбчатую гистограмму пассажиропотока с разбивкой по городам (отдельные столбцы) и авиакомпаниям (разбивка внутри столбца).
3. Построить столбчатую гистограмму усредненной по группам продуктов энергетической ценности, с группировкой по `pnns_groups_1`.

In [39]:

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Пример удаления строк с пропущенными значениями в нужных столбцах
df = df.dropna(subset=['pnns_groups_1', 'energy_100g'])

# Приводим названия групп продуктов к нижнему регистру с использованием .loc
df.loc[:, 'pnns_groups_1'] = df['pnns_groups_1'].str.lower()

# Группируем данные по pnns_groups_1 и вычисляем среднее значение энергии
grouped_energy = df.groupby('pnns_groups_1')['energy_100g'].mean().reset_index()

# Строим столбчатую гистограмму с добавлением hue
plt.figure(figsize=(10, 6))
sns.barplot(x='pnns_groups_1', y='energy_100g', data=grouped_energy, palette='viridis', hue='pnns_groups_1')

# Настройка визуализации
plt.title('Средняя энергетическая ценность по группам продуктов')
plt.xlabel('Группа продуктов')
plt.ylabel('Средняя энергетическая ценность (ккал/100г)')
plt.xticks(rotation=90)  # Поворот подписей для читаемости

plt.show()
```



Вывод:

Я познакомился со средами Jupyter Notebook и Google Colaboratory, а также

библиотеками Pandas и matplotlib