

ГУАП

КАФЕДРА № 43

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель

должность, уч. степень, звание

подпись, дата

М. Д. Поляк

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

Знакомство с Jupyter Notebook

по курсу: Основы машинного обучения

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №

4233К

20.04.2025

подпись, дата

С. В. Голанова

инициалы, фамилия

Санкт-Петербург 2025

Задание 1

Цель

1. Научиться использовать ноутбуки в среде Jupyter (Colaboratory)
2. Настроить аккаунт для работы в Google Colaboratory

Вариант

Вариант 4.

Задание 1

Откройте Jupyter-ноутбук [jupyter_assignment.ipynb](#) в этом репозитории. Скопируйте путь в адресной строке браузера. Перейдите в [Google Colab](#), в меню выберите "Файл" -> "Открыть ноутбук", в открывшемся окне слева выбрать "GitHub", затем:

- вставить в поле для поиска скопированный URL;
- поставить галочку "Показывать личные хранилища" ("Include private repos");
- и нажать на иконку с лупой. При необходимости разрешить Colab доступ к аккаунту GitHub, если откроется новое окно с таким приглашением. Среди результатов поиска выбрать [jupyter_assignment.ipynb](#) и приступить к выполнению задания.

Задание 2

Откройте в Google Colab Jupyter-ноутбук [matplotlib_assignment.ipynb](#), ознакомьтесь с его содержимым и выполните задание.

Задание 3

Откройте в Google Colab Jupyter-ноутбук [pandas_assignment.ipynb](#), ознакомьтесь с его содержимым и выполните задание.

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.

```
In [ ]:

### 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 = 'linesofia14'


#My info
```

```
my_name = 'Sofia'
### 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.

```
In [ ]:
### BEGIN YOUR CODE
print('Hello, Sofia!')
### END YOUR CODE
```

Hello, Sofia!

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.

```
In [ ]:
def greet(name):
    ### BEGIN YOUR CODE
    if isinstance(name, list):
        names = [str(item) for item in name]
        if not names:
            return "Hello!"
        elif len(names) == 1:
            return f"Hello, {names[0]}!"
        elif len(names) == 2:
```

```

    return f"Hello, {names[0]} and {names[1]}!"

else:

    return f"Hello, {' '.join(names[:-1])}, and {names[-1]}!"

else:

    return f"Hello, {str(name)}!"

### END YOUR CODE

```

Повышение отказоустойчивости кода

Что произойдет, если на вход функции поступит список людей? Можно ли сделать так, чтобы код приветствовал нескольких людей, если на вход функции поступит список имен (например, 'Hello, Alex and Bertha!' или 'Hello, Arvin, Beth, and Calvin!')? А что, если кто-то передаст другой тип данных вместо str ('Hello, 3.14!')? Может ли ваш код корректно обработать эти ситуации?

Чтобы понять, какие ситуации будут проверяться, ознакомьтесь с таблицей ниже:

function	args	solution	points
identity	READ_INTRODUCTION	1	0.05
identity	LEARNED_ABOUT_JUPYTER	1	0.05
identity	ACCESS_COLABORATORY	1	0.05
identity	CREATED_GITHUB_ACCOUNT	1	0.05
greet	["Test User"]	"Hello, Test User!"	0.2

- Столбец **function** указывает, какая функция из этого ноутбука будет выполнена. (Примечание: функция **identity** всегда возвращает значение своего входного аргумента.)
- Столбец **args** указывает на один или более (безымянных) аргументов, которые будут переданы указанной в столбце **function** функции.
- Столбец **solution** указывает, какой ответ ожидается в качестве правильного.
- Столбец **points** указывает, сколько баллов будет начислено при успешном прохождении соответствующего теста. В дополнение к перечисленным в таблице публично видимым тестам, код также будет проверяться с помощью скрытых тестов. Итоговая оценка за задание состоит из суммы баллов за публичные и скрытые тесты.

Задание 2.

Задание 2. Знакомство с Matplotlib и LaTeX

В этом задании по номеру варианта задана параметризованная математическая функция. Необходимо её оценить и визуализировать на графике с помощью Python.

Цель

Научиться выполнять базовые математические операции с помощью numpy и визуализировать результаты с помощью matplotlib

Оценивание и баллы

За это задание в общей сложности можно получить до 2 баллов. Задание частично проверяется автоматически, а частично будет оцениваться вручную. Чтобы получить максимальный балл, необходимо успешно выполнить приведенные ниже задачи.

Часть 1. Определить номер варианта

Начнем с импорта библиотеки numpy. Она понадобится позже для выполнения некоторых математических операций. Также потребуется библиотека matplotlib.pyplot, чтобы визуализировать результаты вычислений.

In [2]:

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

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 преподавателю, принимающему лабораторные работы.

In [3]:

```
### BEGIN YOUR CODE
```

```
Student_ID = 4
```

```
### END YOUR CODE
```

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

In [4]:

```
task_id = None if Student_ID is None else Student_ID % 25 if Student_ID % 25 > 0 else
```

25

```
print(f"Пожалуйста, используйте математическую функцию No {task_id} ниже.")
```

Пожалуйста, используйте математическую функцию No 4 ниже.

$$1. y = ax^5 + bx^2 + cx + d$$

$$2. y = a \sin(bx + c)$$

$$3. y = \operatorname{tg}(ax^2 + bx + c)$$

$$4. y = a \ln(b + cx)$$

$$5. y = a \ln \frac{x}{b+cx}$$

$$6. y = \log_d(ax^2 + bx + c)$$

$$7. y = d^{ax^2+bx+c}$$

$$8. y = \log_a \log_b(cx + d)$$

$$9. y = \frac{1}{a \ln|bx + c|}$$

$$10. y = a \sin bx + c \sin dx$$

$$11. y = ax^2 \sin bx$$

$$12. y = a \frac{\sin(bx)}{x} + c$$

$$13. ax^2 + by^2 = c^2$$

$$14. \begin{cases} y = \sqrt{1 - (|x| - 1)^2} \\ y = \arccos(1 - |x|) - \pi \end{cases}$$

$$15. y = ax^3 + bx^2 + cx + d$$

$$16. y = a(\sin bx + \operatorname{tg} cx)$$

$$17. y = ax^2 \cos bx$$

$$18. y = a \cos(bx + c)$$

$$19. y = a \sin bx + c \cos dx$$

$$20. y = \frac{a \cos bx}{x} + c$$

$$21. y = ax^4 + bx^3 + cx + d$$

$$22. y = ax^{-(bx^2+c)}$$

$$23. y = \log_a(b \sin cx)$$

$$24. y = a \ln \frac{bx^2}{cx+d}$$

$$25. y = \operatorname{ctg}(ax^2 + bx + c)$$

В списке математических функций, представленных выше, у, или, более корректно, $y(x)$, является зависимой переменной, полученной в результате вычисления математической функции. a, b, c, d — это скалярные параметры функции, а x — независимая переменная.

Теперь, когда функция выбрана, запишите её в ячейке ниже, используя LaTeX, и выполните ячейку, чтобы отобразить её.

$$y = a \ln(b + cx)$$

Вид формулы на LaTeX представлен ниже.

$$y = a \ln(b + cx)$$

Часть 2. Вычисления в Python

Напишите функцию на Python, которая вычисляет математическую функцию $y(x)$ с заданными скалярными параметрами a, b, c, d (если применимо) и списком значений независимой переменной x . Найти математические функции, доступные в библиотеке numpy, можно [здесь](#).

Пример для функции $y(x) = a \sin^2 x + b \log_c x$ может выглядеть так:

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

In [5]:

```
def my_function(x,a,b,c,d):
```

```
### BEGIN YOUR CODE

return a * np.log(b + c * x)

### END YOUR CODE
```

Задайте значения для параметров a, b, c, d и укажите диапазон для переменной x:

In [6]:

```
### BEGIN YOUR CODE
```

```
a = 2
b = 5
c = 1
d = 2
x = np.linspace(0, 20, 500)
```

```
### END YOUR CODE
```

Вычислите значения функции $y(x)$ на интервале x . Необходимо сделать это векторно и без использования циклов. Постройте график функции. Измените значения параметров a, b, c, d и x в ячейке выше, чтобы получить аккуратный, понятный и красивый график.

In [34]:

```
### BEGIN YOUR CODE
```

```
# y = my_function(...)
```

```
# plt.plot(...)
```

```
'''
```

```
y = my_function(x, a, b, c, d)
```

```
##plt.plot(x, y)
```

```
plt.plot(x, y, color='red', linestyle='--', linewidth=2, label=r'$y = a \cdot \ln(b + c x)$')
```

```
plt.title('График функции $y = a \cdot \ln(b + c x)$')
```

```
plt.xlabel('x')
```

```
plt.ylabel('y')
```

```
plt.grid(True)
```

```
plt.legend()
```

```
plt.show()
```

```

#### END YOUR CODE""

# Создаём вектор x, учитывая, что  $b + c \cdot x > 0$ 
#x = np.linspace(0, 20, 500)

# Наборы параметров (a, b, c)
params = [
    (1, 5, 1),
    (2, 3, 0.5)
]

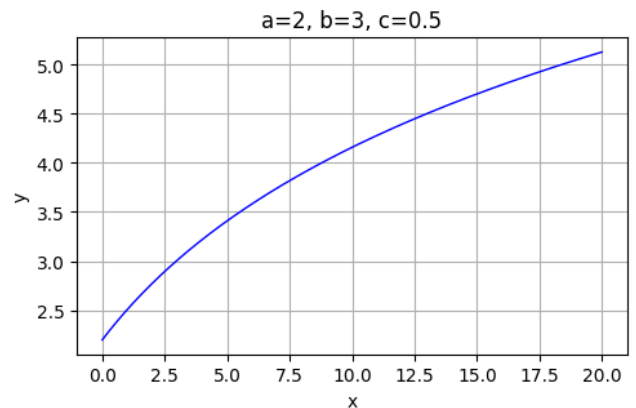
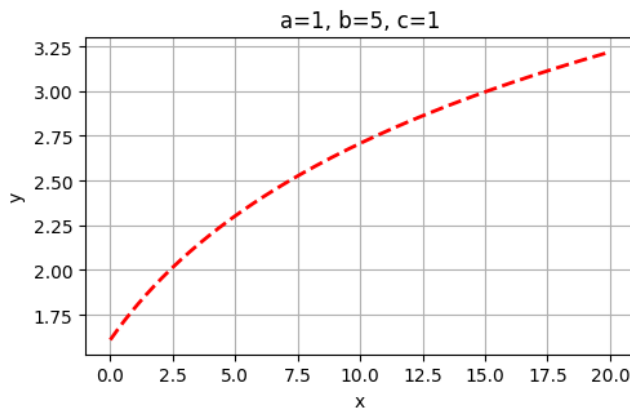
# Создаём фигуру с сеткой 1x2 подграфиков
fig, axs = plt.subplots(1, 2, figsize=(10, 4))
fig.suptitle('Графики функции  $y = a \cdot \ln(b + c \cdot x)$  с разными параметрами',
fontsize=16)

IsFirst = True;
for ax, (a, b, c) in zip(axs.flatten(), params):
    # Для каждого набора параметров корректируем x, чтобы аргумент логарифма
    # был положительным
    x_start = max(0, -b / c + 1e-5)
    x_vals = np.linspace(x_start, 20, 500)
    y_vals = my_function(x_vals, a, b, c, 0)
    if IsFirst:
        ax.plot(x_vals, y_vals, color='red', linestyle='--', linewidth=2)
        IsFirst = False # переключаем флаг
    else:
        ax.plot(x_vals, y_vals, color='blue', linestyle='-', linewidth=1)
        IsFirst = True # переключаем флаг обратно
    ax.set_title(f'a={a}, b={b}, c={c}')
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.grid(True)

plt.tight_layout(rect=[0, 0, 1, 0.95]) # Оставляем место для общего заголовка
plt.show()

```

Графики функции $y = a \cdot \ln(b + cx)$ с разными параметрами



Кастомизируйте свой график

Настройте внешний вид рисунка в ячейке выше. Измените цвет графика функции (например, задайте его красным), обновите стиль линии (например, нарисуйте пунктирную линию или штрих-пунктир), добавьте подписи, метки и так далее. Ознакомьтесь с [документацией](#) для получения дополнительных сведений. Попробуйте разместить несколько графиков с разными наборами значений параметров a , b , c , d на одном рисунке, используя `subplot`. Посмотрите примеры из официального [руководства](#) для вдохновения.

Часть 3. Линейная алгебра в Python

Вычислите скалярное произведение двух векторов `vector_a` и `vector_b`, заданных в ячейке с кодом ниже. Результат поместите в переменную `dot_product`.

In [35]:

```
rng = np.random.RandomState(Student_ID)
```

```
vector_a = rng.choice(np.arange(100, dtype=np.int32), size=(1,5), replace=False)
```

```
vector_b = rng.choice(np.arange(100, dtype=np.int32), size=(5,1), replace=False)
```

```
### BEGIN YOUR CODE
```

```
dot_product = np.dot(vector_a, vector_b).item()
```

```
#dot_product = (vector_a @ vector_b).item() - альтернативная запись
```

```
##dot_product = None
```

```
### END YOUR CODE
```

```
print(f'Вектор A: {vector_a}\nВектор B: {vector_b}\nСкалярное произведение <A, B>={dot_product}')
```

Вектор A: [[20 10 96 16 63]]

Вектор B: [[33]

[88]

[96]

[21]

[56]]

Скалярное произведение $\langle A, B \rangle = 14620$

Задание 3. Знакомство с Pandas

В этом задании необходимо научиться базовым приемам работы с данными. Необходимо загрузить заданный по варианту датасет, провести разведочный анализ данных, сформулировать гипотезы о возможном значении всех полей в датасете, проверить эти гипотезы, и ответить на вопросы по данным.

Цель

Знакомство с разведочным анализом данных и применение Pandas для извлечения информации

Оценивание и баллы

За это задание в общей сложности можно получить до 4 баллов. Задание частично проверяется автоматически, а частично будет оцениваться вручную. Чтобы получить максимальный балл, необходимо успешно выполнить приведенные ниже задачи.

Важные замечания

1. *Откройте этот файл в своем репозитории на GitHub и скопируйте адрес из адресной строки браузера. Перейдите в [Google Colab](#), выберите Файл -> Открыть ноутбук -> GitHub, вставьте скопированный URL и нажмите кнопку поиска (кнопка с лупой справа от поля ввода строки для поиска). В Google Colab откроется копия этого ноутбука из персонального репозитория на GitHub.*
2. *Не удаляйте и не изменяйте имена переменных в ячейках с кодом ниже. Можно добавлять в каждую ячейку произвольное количество строк кода, главное - сохранить результат решения задачи в предопределенную (-ые) переменную (-ые) в соответствующей ячейке. Если этого не сделать, автоматические тесты не будут пройдены.*
3. *Чтобы сохранить работу, выберите Файл -> Сохранить копию на GitHub и **вручную выберите правильный репозиторий из раскрывающегося списка.***
4. *Если в процессе выполнения задания этот файл окажется испорчен, ознакомьтесь с инструкцией в ноутбуке [jupyter assignment](#), раздел "Повторная сдача".*

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.

```
[1]
0 сек.
### BEGIN YOUR CODE

Student_ID = 4

### END YOUR CODE
```

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

```
[2]
0 сек.

datasets = [('Chipotle','https://raw.githubusercontent.com/justmarkham/DAT8/master/data/chipotle.tsv'), ('US Air Carrier market in 2019','https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/aircarrier_market_us_2019.zip'), ('Open Food Facts', 'https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/en.openfoodfacts.org.products.tsv.tar.bz2')]

dataset_id = None if Student_ID is None else Student_ID % len(datasets)
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 'US Air Carrier market in 2019' доступен по следующей ссылке:
https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/aircarrier\_market\_us\_2019.zip
```

В заданиях ниже, где нужно выбрать вопрос, всегда выбирайте вопрос № 2

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.

```
[3]
2 сек.
#### BEGIN YOUR CODE

# !wget PLACE_DATASET_URL_HERE

# !unzip ...
# !tar ...
# !gunzip ...

# Скачиваем архив с датасетом
!wget https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/aircarrier_market_us_2019.zip

# Разархивируем датасет в текущую папку
!unzip -o aircarrier_market_us_2019.zip

#### END YOUR CODE
```

--2025-04-20 17:00:18--
https://raw.githubusercontent.com/markpolyak/datasets/refs/heads/main/data/aircarrier_market_us_2019.zip
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: 10159563 (9.7M) [application/zip]
Saving to: 'aircarrier_market_us_2019.zip'

```
aircarrier_market_u 100%[=====>] 9.69M 64.2MB/s in 0.2s
```

```
2025-04-20 17:00:19 (64.2 MB/s) - 'aircarrier_market_us_2019.zip' saved
[10159563/10159563]
```

Archive: aircarrier_market_us_2019.zip

inflating: aircarrier_market_us_2019.csv

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

```
[20]
2 сек.
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
# Загрузка датасета из CSV файла
df = pd.read_csv('aircarrier_market_us_2019.csv')

# Конвертация названий столбцов в нижний регистр
df.columns = df.columns.str.lower()

# Проверим результат
print(df.head())

# Place your code to read the dataset here. Don't forget to import all the libraries you need!
```

```
passengers  freight  mail  distance  unique_carrier  airline_id \
0      0.0  53185.0  0.0   8165.0          EK      20392
1      0.0   9002.0  0.0   6849.0          EK      20392
2      0.0 2220750.0  0.0   7247.0          EK      20392
3      0.0 1201490.0  0.0   8165.0          EK      20392
4      0.0  248642.0  0.0   6849.0          EK      20392

unique_carrier_name  unique_carrier_entity  region  carrier  ...  dest_state_nm \
0      Emirates          9678A  I  EK  ...      Texas
```

1	Emirates	9678A	I	EK ...	New York
2	Emirates	9678A	I	EK ...	Illinois
3	Emirates	9678A	I	EK ...	NaN
4	Emirates	9678A	I	EK ...	NaN

	dest_country	dest_country_name	dest_wac	year	quarter	month \
0	US	United States	74	2019	1	3
1	US	United States	22	2019	1	3
2	US	United States	41	2019	1	3
3	AE	United Arab Emirates	678	2019	1	3
4	AE	United Arab Emirates	678	2019	1	3

	distance_group	class	data_source
0	17	G	IF
1	14	G	IF
2	15	G	IF
3	17	G	IF
4	14	G	IF

[5 rows x 41 columns]

3. Какие столбцы присутствуют в наборе данных? (0.1 балла)

[5]

0 сек.

#columns = ... # Place your code here instead of '...'

columns = df.columns.tolist()

print(columns) #Это было изначально

```
['passengers', 'freight', 'mail', 'distance', 'unique_carrier', 'airline_id', 'unique_carrier_name',
'unique_carrier_entity', 'region', 'carrier', 'carrier_name', 'carrier_group', 'carrier_group_new',
'origin_airport_id', 'origin_airport_seq_id', 'origin_city_market_id', 'origin', 'origin_city_name',
'origin_state_abr', 'origin_state_fips', 'origin_state_nm', 'origin_country', 'origin_country_name',
'origin_wac', 'dest_airport_id', 'dest_airport_seq_id', 'dest_city_market_id', 'dest', 'dest_city_name',
'dest_state_abr', 'dest_state_fips', 'dest_state_nm', 'dest_country', 'dest_country_name', 'dest_wac',
'year', 'quarter', 'month', 'distance_group', 'class', 'data_source']
```

4. Ответьте на вопрос и сохраните ответ в переменной answer1 (0.1 балла)

Вопросы:

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

[6]

0 сек.

#answer1 = ... # Place your code here instead of '...'

```
answer1 = df['carrier'].nunique()
print(answer1)
```

319

5. Ответьте на вопрос и сохраните ответ в переменной answer2 (0.1 балла)

Вопросы:

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

[7]

0 сек.

#answer2 = ... # Place your code here instead of '...'

```
# Фильтруем данные по маршруту из Великобритании (GB) в США (US)
# Фильтруем по странам в столбцах origin_country и dest_country
filtered = df[(df['origin_country'] == 'GB') & (df['dest_country'] == 'US')]

# Суммируем пассажиров, груз и почту
total_passengers = filtered['passengers'].sum()
total_freight = filtered['freight'].sum()
total_mail = filtered['mail'].sum()
```

```
# Записываем максимальное из трех значений в answer2
```

```
answer2 = max(total_passengers, total_freight, total_mail)
```

```
print(answer2)
```

903296879.0

6. Ответьте на вопрос и сохраните ответ в переменной answer3 (0.2 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?

[8]

0 сек.

```
#answer3 = ... # Place your code here instead of '...'
```

```
# Считаем суммарный груз (mail + freight) для каждого unique_carrier_name
```

```
df['total_cargo'] = df['mail'] + df['freight']
```

```
# Группируем по unique_carrier_name, суммируем груз и находим перевозчика с максимальным грузом
```

```
answer3 = df.groupby('unique_carrier_name')['total_cargo'].sum().idxmax()
```

```
print(answer3)
```

Federal Express Corporation

7. Ответьте на вопрос и сохраните ответ в переменной answer4 (0.25 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 (countries_en) имеющего максимальное содержание холестерина?

[9]

0 сек.

```
#answer4 = ... # Place your code here instead of '...'

# Фильтруем данные по рейсам из США (origin_country == 'US') в другие страны (dest_country != 'US')
filtered = df[(df['origin_country'] == 'US') & (df['dest_country'] != 'US')]

# Сгруппируем по unique_carrier_name, суммируем пассажиров
passengers_by_carrier = filtered.groupby('unique_carrier_name')['passengers'].sum()

# Находим максимальное количество пассажиров, перевезённых одной авиакомпанией
answer4 = passengers_by_carrier.max()

print(answer4)
```

14867653.0

8. Ответьте на вопрос и сохраните ответ в переменной answer5 (0.25 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. Приведите названия всех аллергенов к нижнему регистру. Какой аллерген встречается в продуктах чаще всего?

[10]

2 сек.

```
#answer5 = ... # Place your code here instead of '...'

# Создадим столбец с упорядоченной парой городов (чтобы учесть оба направления)
df['city_pair'] = df.apply(lambda row: tuple(sorted([row['origin_city_name'], row['dest_city_name']])), axis=1)

# Группируем по паре городов и суммируем пассажиров
passengers_by_city_pair = df.groupby('city_pair')['passengers'].sum()

# Находим пару городов с максимальным количеством пассажиров
max_pair = passengers_by_city_pair.idxmax()
```

```
# Записываем ответ в виде списка из двух строк
```

```
answer5 = [max_pair[0], max_pair[1]]
```

```
print(answer5)
```

```
['Chicago, IL', 'New York, NY']
```

9. Ответьте на вопрос и сохраните ответ в переменной answer6 (0.5 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. Найдите самый опасный продукт, содержащий наибольшее количество аллергенов.

```
[14]
```

```
2 сек.
```

```
#answer6 = ... # Place your code here instead of '...'
```

```
# Используем пару городов из предыдущего ответа answer5
```

```
city1, city2 = answer5
```

```
# Создаем столбец с упорядоченной парой городов, чтобы учесть оба направления
```

```
df['city_pair'] = df.apply(lambda row: tuple(sorted([row['origin_city_name'], row['dest_city_name']])), axis=1)
```

```
# Фильтруем данные по выбранной паре городов
```

```
filtered = df[df['city_pair'] == (city1, city2)]
```

```
# Считаем суммарное количество пассажиров для каждой авиакомпании на этом маршруте
```

```
passengers_by_carrier = filtered.groupby('unique_carrier_name')['passengers'].sum()
```

```
# Берем топ-3 авиакомпании по пассажиропотоку
```

```
top3 = passengers_by_carrier.sort_values(ascending=False).head(3)
```

```
# Общий пассажиропоток между этими городами (по условию задачи)
total_passengers = passengers_by_carrier.sum()
```

```
# Вычисляем процент пассажиров, перевезенных каждой из топ-
3 авиакомпаний, округляем до 2 знаков
answer6 = [round((count / total_passengers) * 100, 2) for count in top3]
```

```
print(answer6)
```

```
[np.float64(31.31), np.float64(23.23), np.float64(13.33)]
```

10. Ответьте на вопрос и сохраните ответ в переменной answer7 (0.5 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.

```
[17]
```

```
0 сек.
```

```
#answer7 = ... # Place your code here instead of '...'
```

```
# Фильтруем данные по рейсам класса F (премиум-класс)
```

```
df_f = df[df['class'] == 'F']
```

```
# Фильтруем международные рейсы с участием США и создаём копию, чтобы не менять исх
одный df
```

```
df_f_intl = df_f[((df_f['origin_country'] == 'US') & (df_f['dest_country'] != 'US')) |
                 ((df_f['dest_country'] == 'US') & (df_f['origin_country'] != 'US'))].copy()
```

```

# Создаём новый столбец foreign_country в копии, не меняя оригинал
foreign_countries = df_f_intl.apply(
    lambda row: row['dest_country'] if row['origin_country'] == 'US' else row['origin_country'], axis
    =1)

# Группируем по иностранной стране и считаем суммарное количество пассажиров
passengers_by_country = df_f_intl.groupby(foreign_countries)['passengers'].sum()

# Считаем общий пассажиропоток (международный премиум-
класс) для нормировки процентов
total_passengers = passengers_by_country.sum()

# Считаем процент пассажиропотока каждой страны относительно общего
percent_by_country = (passengers_by_country / total_passengers) * 100

# Сортируем по убыванию и выбираем страну с третьим по величине пассажиропотоком
answer7 = percent_by_country.sort_values(ascending=False).index[2]
print(answer7)

```

GB

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 буррито 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, являющейся наиболее сбалансированной с точки зрения среднего содержания белков, жиров и углеводов? Под "сбалансированной" понимать близость БЖУ к пропорции 1:1:4.

[19]

2 сек.

#answer8 = ... # Place your code here instead of '...'

import calendar

```

# Используем пару городов из предыдущего ответа answer5
city1, city2 = answer5

# Создаем столбец с упорядоченной парой городов, чтобы учесть оба направления
df['city_pair'] = df.apply(lambda row: tuple(sorted([row['origin_city_name'], row['dest_city_name']])), axis=1)

# Фильтруем данные по выбранной паре городов
filtered = df[df['city_pair'] == (city1, city2)]

# Группируем по месяцам и суммируем пассажиров
passengers_by_month = filtered.groupby('month')['passengers'].sum()

# Находим месяц с максимальным пассажиропотоком
answer8 = passengers_by_month.idxmax()
print(answer8)
month_name = calendar.month_name[answer8]
print(month_name)

```

5

May

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

1. Постройте 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. Постройте столбчатую гистограмму усредненной по группам продуктов энергетической ценности, с группировкой по pnnns_groups_1.

[22]

Place your code here

```

# Группируем по городам отправления и авиакомпаниям, суммируем пассажиров
grouped = df.groupby(['origin_city_name', 'unique_carrier_name'])['passengers'].sum().unstack(fill_value=0)

# Построение графика
ax = grouped.plot(kind='bar', stacked=True, figsize=(14,8))

```

```
# Настройка графика
```

```
plt.title('Пассажиропоток по городам и авиакомпаниям')
```

```
plt.xlabel('Город')
```

```
plt.ylabel('Количество пассажиров')
```

```
# Легенда с фиксированным расположением и выносом за пределы графика
```

```
plt.legend(title='Авиакомпания', loc='center left', bbox_to_anchor=(1.0, 0.5))
```

```
# Поворот подписей по оси X
```

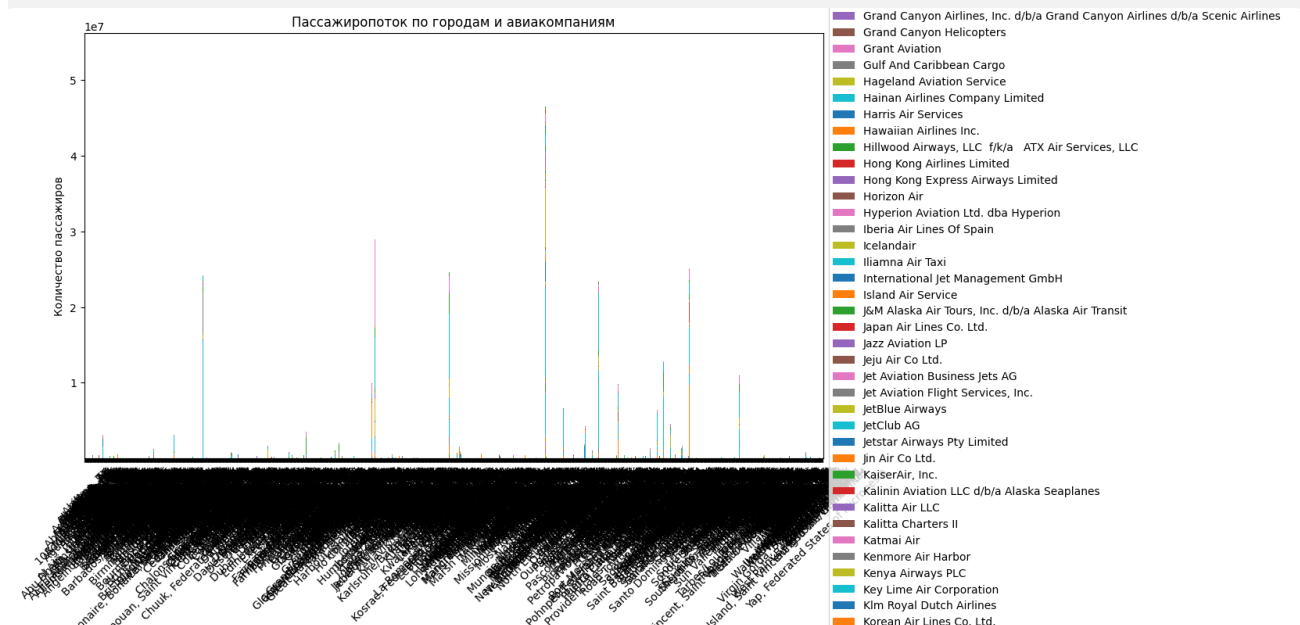
```
plt.xticks(rotation=45)
```

```
# Подгонка отступов
```

```
plt.tight_layout()
```

```
plt.subplots_adjust(right=0.8, bottom=0.2)
```

```
plt.show()
```



Полный вид графика приведен ниже.

12. Визуализируйте данные в соответствии с заданием (1 балл)
2. Постройте стековую столбчатую гистограмму пассажиропотока с разбивкой по городам (отдельные столбцы) и авиакомпаниям (разбивка внутри столбца).
Гистограмма, где показано, только 10 городов с самым большим пассажирооборотом

```
[25]
8 сек.

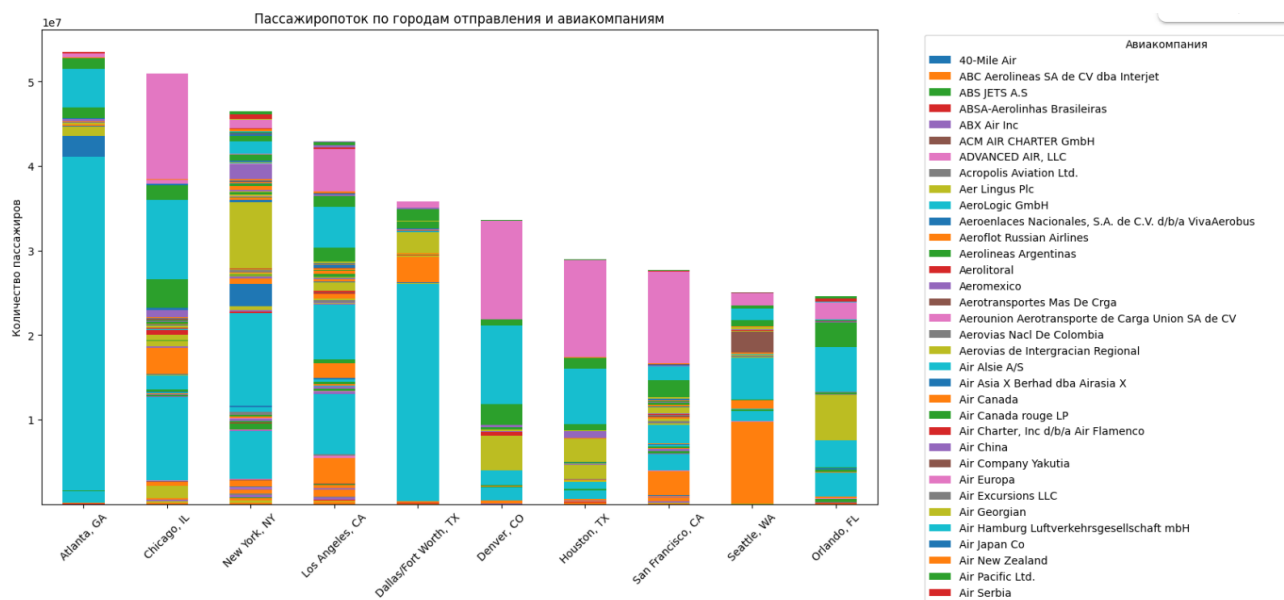
# Группируем по городам отправления и авиакомпаниям, суммируем пассажиров
grouped2 = df.groupby(['origin_city_name', 'unique_carrier_name'])['passengers'].sum().unstack(fill_value=0)

# Выбираем топ-10 городов по общему пассажиропотоку
top_cities = grouped2.sum(axis=1).sort_values(ascending=False).head(10).index

# Берем данные только по топ-10 городам
grouped_top = grouped2.loc[top_cities]

# Строим стековую столбчатую гистограмму
ax = grouped_top.plot(kind='bar', stacked=True, figsize=(14, 8))

ax.set_title('Пассажиропоток по городам отправления и авиакомпаниям')
ax.set_xlabel('Город отправления')
ax.set_ylabel('Количество пассажиров')
ax.legend(title='Авиакомпания', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Ниже представлен полный вид гистограммы.

Вывод

В ходе выполнения лабораторной работы были получены практические навыки работы с ноутбуками в среде Jupyter (Colaboratory). Также была настроена учётная запись для работы в Google Colaboratory. Кроме того, приобретены умения выполнять базовые математические операции с использованием библиотеки `numpy` и визуализировать результаты с помощью библиотеки `matplotlib`.

В ходе лабораторной работы также было уделено внимание разведочному анализу данных и применению библиотеки `Pandas` для извлечения информации.