Мы сохранили страницу с википедии про языки программирования и сохранили по адресу <https://stepik.org/media/attachments/lesson/209717/1.html>

Скачайте её с помощью скрипта на Питоне и посчитайте, какой язык упоминается чаще Python или C++ (ответ должен быть одной из этих двух строк).

Решение: файл **ex00.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Файл <https://stepik.org/media/attachments/lesson/209719/2.html> содержит статью с Википедии про язык Python. В этой статье есть теги code, которыми выделяются конструкции на языке Python. Вам нужно найти все строки, содержащиеся между тегами <code> и </code> и найти те строки, которые встречаются чаще всего и вывести их в алфавитном порядке, разделяя пробелами.

Например, если исходный текст страницы выглядел бы так:

<code>a</code>

<a>bracadabr</a>

<code>c</code>

<code>b</code>

<code>b</code>

<code>c</code>

то в ответ надо было бы ввести строку "b c".

Решение: файл **ex01.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Сейчас вам нужно установить библиотеку BeautifulSoup. Чтобы проверить, что всё установилось хорошо, необходимо запусить код, представленный ниже и сдать в качестве ответа то, что он выводит.

Решение: файл **ex02.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В файле <https://stepik.org/media/attachments/lesson/209723/3.html> находится одна таблица. Просуммируйте все числа в ней и введите в качестве ответа одно число - эту сумму. Для доступа к ячейкам используйте возможности BeautifulSoup.

Решение: файл **ex03.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В файле <https://stepik.org/media/attachments/lesson/209723/4.html> находится одна таблица. Просуммируйте все числа в ней. Теперь мы добавили разных тегов для изменения стиля отображения. Для доступа к ячейкам используйте возможности BeautifulSoup.

Решение: файл **ex04.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В файле <https://stepik.org/media/attachments/lesson/209723/5.html> находится одна таблица. Просуммируйте все числа в ней. Теперь мы не только добавили разных тегов для изменения стиля отображения, но и сделали невалидный HTML-код (правда, браузеры его отображают, а вот с BeautifulSoup могут быть проблемы). Невалидный HTML-код - не редкость в интернете, надо учиться работать и с этим.

Вы можете исправить html-код или попробовать использовать нестандартный парсер html, такой как html5lib.

Решение: файл **ex05.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Для решения этой задачи необходимо установить библиотеку xlrd, скачать файл <https://stepik.org/media/attachments/lesson/245266/tab.xlsx> и создать в папке с этим файлом скрипт со следующем содержанием:

import xlrd

wb = xlrd.open\_workbook('tab.xlsx')

sheet\_names = wb.sheet\_names()

sh = wb.sheet\_by\_name(sheet\_names[0])

nmin = sh.row\_values(6)[2]

for rownum in range(7, 27):

temp = sh.row\_values(rownum)

nmin = min(nmin, temp[2])

print(nmin)

Запустите скрипт и в качестве ответа введите то, что он выведет.

Решение: файл **ex06.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Вася планирует карьеру и переезд. Для это составил таблицу, в которой для каждого региона записал зарплаты для разных интересные ему профессий. Таблица доступна по ссылке <https://stepik.org/media/attachments/lesson/245267/salaries.xlsx>. Выведите название региона с самой высокой медианной зарплатой (медианой называется элемент, стоящий в середине массива после его упорядочивания) и, через пробел, название профессии с самой высокой средней зарплатой по всем регионам.

Решение: файл **ex07.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Васю назначили завхозом в туристической группе и он подошёл к подготовке ответственно, составив справочник продуктов с указанием калорийности на 100 грамм, а также содержание белков, жиров и углеводов на 100 грамм продукта. Ему не удалось найти всю информацию, поэтому некоторые ячейки остались незаполненными (можно считать их значение равным нулю). Также он использовал какой-то странный офисный пакет и разделял целую и дробную часть чисел запятой. Таблица доступна по ссылке<https://stepik.org/media/attachments/lesson/245290/trekking1.xlsx>

Вася хочет минимизировать вес продуктов и для этого брать самые калорийные продукты. Помогите ему и упорядочите продукты по убыванию калорийности. В случае, если продукты имеют одинаковую калорийность - упорядочите их по названию. В качестве ответа необходимо сдать названия продуктов, по одному в строке.

Решение: файл **ex08.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Васю назначили завхозом в туристической группе и он подошёл к подготовке ответственно, составив справочник продуктов с указанием калорийности на 100 грамм, а также содержание белков, жиров и углеводов на 100 грамм продукта. Ему не удалось найти всю информацию, поэтому некоторые ячейки остались незаполненными (можно считать их значение равным нулю). Также он использовал какой-то странный офисный пакет и разделял целую и дробную часть чисел запятой. Таблица доступна по ссылке <https://stepik.org/media/attachments/lesson/245290/trekking2.xlsx>

Вася составил раскладку по продуктам на один день (она на листе "Раскладка") с указанием названия продукта и его количества в граммах. Посчитайте 4 числа: суммарную калорийность и граммы белков, жиров и углеводов. Числа округлите до целых вниз и введите через пробел.

Решение: файл **ex09.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Васю назначили завхозом в туристической группе и он подошёл к подготовке ответственно, составив справочник продуктов с указанием калорийности на 100 грамм, а также содержание белков, жиров и углеводов на 100 грамм продукта. Ему не удалось найти всю информацию, поэтому некоторые ячейки остались незаполненными (можно считать их значение равным нулю). Также он использовал какой-то странный офисный пакет и разделял целую и дробную часть чисел запятой. Таблица доступна по ссылке <https://stepik.org/media/attachments/lesson/245290/trekking3.xlsx>

Вася составил раскладку по продуктам на весь поход (она на листе "Раскладка") с указанием номера дня, названия продукта и его количества в граммах. Для каждого дня посчитайте 4 числа: суммарную калорийность и граммы белков, жиров и углеводов. Числа округлите до целых вниз и введите через пробел. Информация о каждом дне должна выводиться в отдельной строке.

Решение: файл **ex10.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Главный бухгалтер компании "Рога и копыта" случайно удалил ведомость с начисленной зарплатой. К счастью, у него сохранились расчётные листки всех сотрудников. Помогите по этим расчётным листкам восстановить зарплатную ведомость. Архив с расчётными листками доступен по ссылке <https://stepik.org/media/attachments/lesson/245299/rogaikopyta.zip>

Ведомость должна содержать 1000 строк, в каждой строке должно быть указано ФИО сотрудника и, через пробел, его зарплата. Сотрудники должны быть упорядочены по алфавиту.

Решение: файл **ex11.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В этой задаче нужно просто установить библиотеку xmltodict, скачать файл <https://stepik.org/media/attachments/lesson/245571/map1.osm>, создать в директории с файлом скрипт со следующим содержанием:

import xmltodict

fin = open('map1.osm', 'r', encoding='utf8')

xml = fin.read()

fin.close()

parsedxml = xmltodict.parse(xml)

print(parsedxml['osm']['node'][100]['@id'])

и ввести в качестве ответа вывод этого скрипта.

Решение: файл **ex12.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В OpenStreetMap XML встречаются теги node, которые соответствуют некоторым точкам на карте. Ноды могут не только обозначать какой-то точечный объект, но и входить в состав way (некоторой линии, возможно замкнутой) и не иметь собственных тегов. Для доступного по ссылке <https://stepik.org/media/attachments/lesson/245678/map1.osm> фрагмента карты посчитайте, сколько node имеет хотя бы один вложенный тэг tag, а сколько - не имеют. В качестве ответа введите два числа, разделённых пробелом.

Решение: файл **ex13.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Вася решил открыть АЗС (заправку). Чтобы оценить уровень конкуренции он хочет изучить количество заправок в интересующем его районе. Вася скачал интересующий его кусок карты OSM <https://stepik.org/media/attachments/lesson/245681/map2.osm> и хочет посчитать, сколько на нём отмечено точечных объектов (node), являющихся заправкой. В качестве ответа вам необходимо вывести одно число - количество АЗС.

"Как обозначается заправка в OpenStreetMap" - пример хорошего запроса чтобы узнать, как обозначается заправка в OpenStreetMap.

Решение: файл **ex14.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Вася, открывший заправку в прошлом уроке, разорился. Конкуренция оказалась слишком большой. Вася предполагает, что это произошло от того, что теги заправки могут быть не только на точке, но и на каком-то контуре. Определите, сколько заправок на самом деле (не только обозначенных точкой) есть на фрагменте карты <https://stepik.org/media/attachments/lesson/245681/map2.osm>

Решение: файл **ex15.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Это задание необходимо для того, чтобы проверить, что у вас все хорошо с библиотекой matplotlib.

Создайте и запустите следующий файл:

#!/usr/bin/env python3

import matplotlib as mpl

import matplotlib.pyplot as plt

import math

dpi = 80

fig = plt.figure(dpi = dpi, figsize = (512 / dpi, 384 / dpi) )

mpl.rcParams.update({'font.size': 10})

plt.axis([0, 10, -1.5, 1.5])

plt.title('Sine & Cosine')

plt.xlabel('x')

plt.ylabel('F(x)')

xs = []

sin\_vals = []

cos\_vals = []

x = 0.0

while x < 10.0:

sin\_vals += [ math.sin(3 \* x) ]

cos\_vals += [ math.cos(x) ]

xs += [x]

x += 0.1

plt.plot(xs, sin\_vals, color = 'blue', linestyle = 'solid',

label = 'sin(3x)')

plt.plot(xs, cos\_vals, color = 'red', linestyle = 'dashed',

label = 'cos(x)')

plt.legend(loc = 'upper right')

fig.savefig('trigan.png')

Откройте файл trigan.png и выберите, значение какой функции больше в точке x = 6. У вас всего одна попытка ответить на этот вопрос, будьте внимательны!

Решение: файл **ex16.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В видео мы нашли хорошую статью, которая находится по ссылке <https://eax.me/python-matplotlib/>, она может быть вам полезна.

Воспользуемся первым примером оттуда:

#!/usr/bin/env python3

# vim: set ai et ts=4 sw=4:

import matplotlib as mpl

import matplotlib.pyplot as plt

import math

dpi = 80

fig = plt.figure(dpi = dpi, figsize = (512 / dpi, 384 / dpi) )

mpl.rcParams.update({'font.size': 10})

plt.axis([0, 10, -1.5, 1.5])

plt.title('Sine & Cosine')

plt.xlabel('x')

plt.ylabel('F(x)')

xs = []

sin\_vals = []

cos\_vals = []

x = 0.0

while x < 10.0:

sin\_vals += [ math.sin(x) ]

cos\_vals += [ math.cos(x) ]

xs += [x]

x += 0.1

plt.plot(xs, sin\_vals, color = 'blue', linestyle = 'solid',

label = 'sin(x)')

plt.plot(xs, cos\_vals, color = 'red', linestyle = 'dashed',

label = 'cos(x)')

plt.legend(loc = 'upper right')

fig.savefig('trigan.png')

Вам необходимо переместить легенду (подписи к графикам) из правого верхнего в левый нижний угол. Исправьте ровно одну строку в этой программе (не меняя ничего другого) и сдайте исправленный исходный код программы. Пожалуйста, используйте одинарные кавычки и расстановку пробелов в том же стиле, в котором написан пример.

Решение: файл **ex17.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Воспользуемся еще одним примером из статьи <https://eax.me/python-matplotlib/>:

#!/usr/bin/env python3

# vim: set ai et ts=4 sw=4:

import matplotlib as mpl

import matplotlib.pyplot as plt

import matplotlib.dates as mdates

import datetime as dt

import csv

data\_names = ['cafe', 'pharmacy', 'fuel', 'bank', 'waste\_disposal',

'atm', 'bench', 'parking', 'restaurant',

'place\_of\_worship']

data\_values = [9124, 8652, 7592, 7515, 7041, 6487, 6374, 6277,

5092, 3629]

dpi = 80

fig = plt.figure(dpi = dpi, figsize = (512 / dpi, 384 / dpi) )

mpl.rcParams.update({'font.size': 10})

plt.title('OpenStreetMap Point Types')

ax = plt.axes()

ax.yaxis.grid(True, zorder = 1)

xs = range(len(data\_names))

plt.bar([x + 0.05 for x in xs], [ d \* 0.9 for d in data\_values],

width = 0.2, color = 'red', alpha = 0.7, label = '2016',

zorder = 2)

plt.bar([x + 0.3 for x in xs], data\_values,

width = 0.2, color = 'blue', alpha = 0.7, label = '2017',

zorder = 2)

plt.xticks(xs, data\_names)

fig.autofmt\_xdate(rotation = 25)

plt.legend(loc='upper right')

fig.savefig('bars.png')

Вам необходимо сделать так, чтобы размер генерируемого изображения стал 1024 на 768. Исправьте ровно одну строку в этой программе (не меняя ничего другого) и сдайте исправленный исходный код программы. Пожалуйста, используйте одинарные кавычки и расстановку пробелов в том же стиле, в котором написан пример.

Решение: файл **ex18.py**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В этой задаче вам необходимо сдать на проверку html-файл, в котором будет таблица размером 2 на 2, которая должна содержать в ячейках первой строки числа 1 и 2, а в ячейках второй строки числа 3 и 4. При открытии вашего файла в браузере это должно выглядеть примерно так:

https://ucarecdn.com/c016e25f-2981-45ad-989c-8c91f693204c/

Ваш файл должен начинаться с тегов <html> и <body> и заканчиваться </body> и </html>.

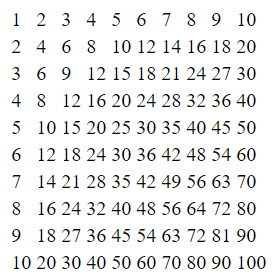
Для создания таблицы можно пользоваться тегами <table> (создание таблицы), <tr> (создание строки в таблице) и <td> (создание отдельной ячейки). Все открытые теги нужно закрыть, причем сделать это нужно в правильном порядке.

Пожалуйста, не используйте никаких украшений и других тегов - мы не сможем проверить такие решения.

Решение: файл **ex19.py**, **ex19.html**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В этой задаче вам необходимо научиться генерировать html-код на питоне и сдать на проверку html-файл, в котором будет таблица размером 10 на 10, которая должна содержать таблицу умножения для чисел от 1 до 10. При открытии вашего файла в браузере это должно выглядеть примерно так:



Ваш файл должен начинаться с тегов <html> и <body> и заканчиваться </body> и </html>.

Для создания таблицы можно пользоваться тегами <table> (создание таблицы), <tr> (создание строки в таблице) и <td> (создание отдельной ячейки). Все открытые теги нужно закрыть, причем сделать это нужно в правильном порядке.

Пожалуйста, не используйте никаких украшений и других тегов - мы не сможем проверить такие решения.

Решение: файл **ex20.py, ex20.html**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Ваш файл должен начинаться с тегов <html> и <body> и заканчиваться </body> и </html>.

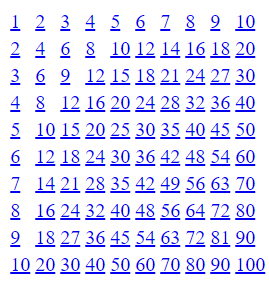
Для создания таблицы можно пользоваться тегами <table> (создание таблицы), <tr> (создание строки в таблице) и <td> (создание отдельной ячейки). Все открытые теги нужно закрыть, причем сделать это нужно в правильном порядке.

Для создания ссылки пользуйтесь тегом <a>. Например, ссылка на страницу http://hse.ru с текстом "Высшая школа экономики" должна выглядеть так: <a href=http://hse.ru>Высшая школа экономики</a>.

Пожалуйста, не используйте никаких украшений и других тегов - мы не сможем проверить такие решения.

В этой задаче вам предстоит научиться создавать ссылки. Вам нужно сгенерировать html-код на питоне и сдать на проверку html-файл, в котором будет таблица размером 10 на 10, которая должна содержать таблицу умножения для чисел от 1 до 10. Каждое число в таблице должно быть ссылкой на страницу http://<это число>.ru. Например, число 12 должно быть ссылкой на страницу http://12.ru

При открытии вашего файла в браузере это должно выглядеть примерно так:



Ваш файл должен начинаться с тегов <html> и <body> и заканчиваться </body> и </html>.

Для создания таблицы можно пользоваться тегами <table> (создание таблицы), <tr> (создание строки в таблице) и <td> (создание отдельной ячейки). Все открытые теги нужно закрыть, причем сделать это нужно в правильном порядке.

Для создания ссылки пользуйтесь тегом <a>. Например, ссылка на страницу http://hse.ru с текстом "Высшая школа экономики" должна выглядеть так: <a href=http://hse.ru>Высшая школа экономики</a>.

Пожалуйста, не используйте никаких украшений и других тегов - мы не сможем проверить такие решения.

Решение: файл **ex21.py, ex21.html**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В этой задаче вам предстоит настроить и запустить веб-сервер. Создайте файл index.html с формой, в которой есть единственное текстовое поле ввода с именем INPUT\_TEXT и кнопка отправить.

В папке cgi-bin создайте скрипт со следующим содержанием:

#!/usr/bin/env python3

import cgi

def ohash(s):

ans = 0

for c in s:

ans = ans \* 123417 + ord(c)

return ans

form = cgi.FieldStorage()

text = form.getfirst("INPUT\_TEXT", "не задано")

print("Content-type: text/html\n")

print("""<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8">

</head>

<body>""")

print("<h1>" + str(ohash(text)) + "</h1>")

print("""</body>

</html>""")

Запустите скрипт, поднимающий веб-сервер с вашей формой, откройте страницу и введите слово Python (обязательно с большой буквы). У вас должна открыться страница, содержащая единственное число, введите его в качестве ответа.

Решение: папка **ex22**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

В этой задаче вам предстоит настроить и запустить веб-сервер, а также обнаружить проблему в скрипте, обрабатывающем данные формы. Создайте файл index.html с формой, в которой есть единственное текстовое поле ввода с именем INPUT\_TEXT и кнопка отправить.

В папке cgi-bin создайте скрипт со следующим содержанием:

#!/usr/bin/env python3

import cgi

form = cgi.FieldStorage()

text = form.getfirst("INPUT\_TEXT", "не задано")

text = text[:-1]

print("Content-type: text/html\n")

print("""<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8">

</head>

<body>""")

for i in range(1, len(text) + 1, 2):

print("<h1>" + str(hash(text[i])) + "</h1>")

print("""</body>

</html>""")

Запустите скрипт, поднимающий веб-сервер с вашей формой, откройте страницу и введите слово Python (обязательно с большой буквы). У вас должна открыться страница, содержащая несколько чисел. Однако, если вы откроете ее исходный текст, то обнаружите, что она обрывается без закрытия некоторых тегов, а это значит, что что-то пошло не так.

Поймите, где искать сообщение об ошибке и в качестве ответа введите номер строки, в которой произошла ошибка.

Решение: папка **ex23**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**