Отчет по пятой практической работе

Выполнение задания 1:

Код программы:

import requests

def get\_cat\_fact():

url = 'https://catfact.ninja/fact'

response = requests.get(url)

if response.status\_code == 200:

data = response.json()

return data.get('fact')

else:

return None

def get\_custom\_cat\_fact():

endpoint = input("Введите конечную точку API (например, 'fact', 'breeds'): ")

url = f'https://catfact.ninja/{endpoint}'

response = requests.get(url)

if response.status\_code == 200:

data = response.json()

return data

else:

return None

def main():

print("Привет! Чтобы получить случайный факт о котах, нажмите 1.")

print("Для выбора конкретной конечной точки API, нажмите 2.")

choice = input("Выберите опцию (1 или 2): ")

if choice == '1':

fact = get\_cat\_fact()

if fact:

print("Случайный факт о котах:")

print(fact)

else:

print("Не удалось получить факт о котах.")

elif choice == '2':

custom\_data = get\_custom\_cat\_fact()

if custom\_data:

print("Данные из выбранной конечной точки:")

print(custom\_data)

else:

print("Не удалось получить данные из выбранной конечной точки.")

else:

print("Некорректный выбор. Пожалуйста, выберите 1 или 2.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат работы программы:

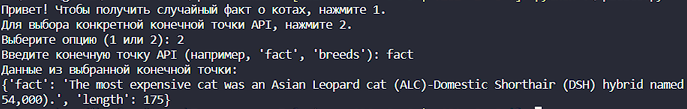


Рисунок 1 – Результат работы программы

Выполнение задания 2:

Код программы:

import requests

from PIL import Image

from io import BytesIO

def get\_cat\_images(query=None, params=None):

url = 'https://cataas.com/cat'

if query:

url += f'/{query}'

response = requests.get(url, params=params)

if response.status\_code == 200:

return response.content

else:

return None

def display\_image(image\_data):

if image\_data:

image = Image.open(BytesIO(image\_data))

image.show()

else:

print("Не удалось получить изображение.")

def main():

print("Привет! Чтобы получить фотографию кота, нажмите 1.")

print("Для получения фотографии с дополнительными параметрами, нажмите 2.")

choice = input("Выберите опцию (1 или 2): ")

if choice == '1':

image\_data = get\_cat\_images()

display\_image(image\_data)

elif choice == '2':

query = input("Введите запрос (например, 'funny', 'cute'): ")

params = {'filter': query}

image\_data = get\_cat\_images(params=params)

display\_image(image\_data)

else:

print("Некорректный выбор. Пожалуйста, выберите 1 или 2.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат работы программы:



Рисунок 2 – Результат работы программы

Выполнение задания 3:

Код программы:

import requests

def get\_holidays(country\_code):

url = f'https://date.nager.at/Api/v2/NextPublicHolidays/{country\_code}'

response = requests.get(url)

if response.status\_code == 200:

data = response.json()

return data

else:

return None

def display\_holidays(holidays):

if holidays:

print("Праздники и даты:")

for holiday in holidays:

print(f"{holiday['name']} - {holiday['date']}")

else:

print("Не удалось получить данные о праздниках.")

def main():

print("Привет! Этот скрипт позволяет получить список праздников по стране.")

country\_code = input("Введите код страны (например, 'US' для США, 'GB' для Великобритании): ")

holidays = get\_holidays(country\_code)

display\_holidays(holidays)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат работы программы:

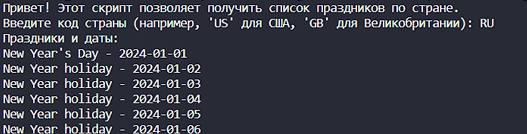


Рисунок 3 – Результат работы программы

Выполнение задания 4:

Код программы:

import requests

def get\_agents():

url = 'https://valorant-api.com/v1/agents'

response = requests.get(url)

if response.status\_code == 200:

agents\_data = response.json()

for agent in agents\_data['data']:

print(f"Имя персонажа: {agent['displayName']}")

print(f"Имя агента: {agent['developerName']}")

print(f"Описание: {agent['description']}")

print("--------------------")

else:

print("Ошибка при получении данных об агентах")

def get\_maps():

url = 'https://valorant-api.com/v1/maps'

response = requests.get(url)

if response.status\_code == 200:

maps\_data = response.json()

for map\_info in maps\_data['data']:

print(f"Имя карты: {map\_info['displayName']}")

print(f"Описание: {map\_info['narrativeDescription']}")

print("--------------------")

else:

print("Ошибка при получении данных о картах")

def get\_weapons():

url = 'https://valorant-api.com/v1/weapons'

response = requests.get(url)

if response.status\_code == 200:

weapons\_data = response.json()

for weapon in weapons\_data['data']:

print(f"Название оружия: {weapon['displayName']}")

print(f"Тип: {weapon['category']}")

print("--------------------")

else:

print("Ошибка при получении данных об оружии")

def get\_events():

url = 'https://valorant-api.com/v1/events'

response = requests.get(url)

if response.status\_code == 200:

events\_data = response.json()

for event in events\_data['data']:

print(f"Название события: {event['displayName']}")

print(f"Дата начала: {event['startTime']}")

print(f"Дата окончания: {event['endTime']}")

print("--------------------")

else:

print("Ошибка при получении данных о событиях")

get\_agents()

get\_maps()

get\_weapons()

get\_events()

Результат работы программы:

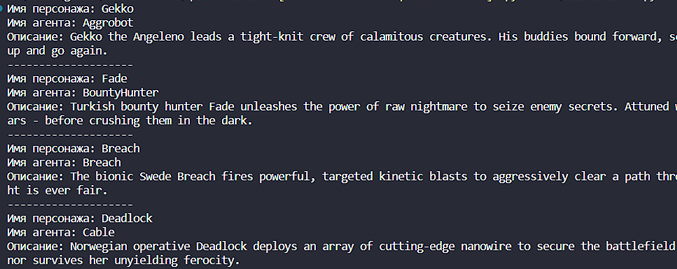


Рисунок 4 – Результат работы программы

Выполнение задания 5:

Код программы:

from flask import Flask, jsonify, request

app = Flask(\_\_name\_\_)

cars = [

{"id": 1, "brand": "Мерседес", "model": "E-Class", "year": 2020},

{"id": 2, "brand": "БМВ", "model": "5 Series", "year": 2021},

{"id": 3, "brand": "Ауди", "model": "A4", "year": 2019},

{"id": 4, "brand": "БМВ", "model": "X3", "year": 2022},

{"id": 5, "brand": "Мерседес", "model": "S-Class", "year": 2023}

]

# Получение всех машин

@app.route('/api/cars', methods=['GET'])

def get\_cars():

return jsonify({"машины": cars})

# Получение информации о конкретной машине по ID

@app.route('/api/cars/<int:car\_id>', methods=['GET'])

def get\_car(car\_id):

car = next((car for car in cars if car['id'] == car\_id), None)

if car:

return jsonify({"машина": car})

return jsonify({"сообщение": "Машина не найдена"}), 404

# Поиск машины по бренду, модели или году

@app.route('/api/search', methods=['GET'])

def search\_cars():

query = request.args.get('q')

if query:

result = [car for car in cars if query.lower() in car['brand'].lower() or query.lower() in car['model'].lower() or query.lower() in str(car['year'])]

if result:

return jsonify({"результат": result})

return jsonify({"сообщение": "Машины не найдены"}), 404

# Добавление новой машины

@app.route('/api/cars', methods=['POST'])

def add\_car():

data = request.json

if data and 'brand' in data and 'model' in data and 'year' in data:

new\_id = max(car['id'] for car in cars) + 1

new\_car = {"id": new\_id, "brand": data['brand'], "model": data['model'], "year": data['year']}

cars.append(new\_car)

return jsonify({"машина": new\_car}), 201

return jsonify({"сообщение": "Неверные данные"}), 400

# Удаление машины по ID

@app.route('/api/cars/<int:car\_id>', methods=['DELETE'])

def delete\_car(car\_id):

global cars

cars = [car for car in cars if car['id'] != car\_id]

return jsonify({"сообщение": "Машина удалена"}), 200

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

Результат работы программы:

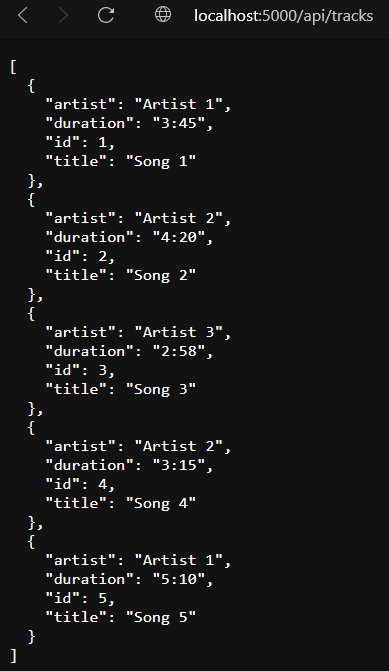


Рисунок 5 - Результат работы программы