МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторная работа №4
По дисциплине “**Современные платформы программирования**”

Выполнил:
студент группы ПО-11
Сильчук Д.А.
Проверил:
Козик И. Д.

Брест 2025

Цель: научиться работать с Github API, приобрести практические навыки написания программ для работы с REST API или GraphQL API.

Вариант 6

Общее задание: используя Github API, реализовать предложенное задание на языке Python. Выполнить визуализацию результатов, с использованием графика или отчета. Можно использовать как REST API (рекомендуется), так и GraphQL.

Задание 1. Анализ вклада разработчика в open-source на GitHub
Напишите Python-скрипт, который:
1. Запрашивает у пользователя имя пользователя GitHub.
2. Получает все публичные репозитории, в которые этот пользователь вносил вклад (не только его собственные, но и форки, pull requests).
3. Для каждого репозитория определяет:
   - Количество коммитов пользователя
   - Количество открытых pull requests
   - Количество закрытых pull requests
   - Количество issues, созданных пользователем

4. Вычисляет уровень активности пользователя, например, на основе взвешенной формулы:
Активность = (Коммиты * 1) + (Открытые PR * 2) + (Закрытые PR * 3) + (Issues * 1.5)
5. Определяет самый активный проект, в котором пользователь работал.
6. Сохраняет данные в github_contribution.json.
“Введите имя пользователя GitHub: octocat
Пользователь octocat внес вклад в 12 репозиториев.
Общее количество коммитов: 320
Открытых pull requests: 15
Закрытых pull requests: 30
Созданных issues: 25
Активность: 530 баллов
Самый активный проект: fastapi/fastapi (120 коммитов, 10 PR)
Результаты сохранены в github_contribution.json”

Код программы:

```
import urllib.request
import json
from datetime import datetime

# GitHub API configuration
GITHUB_API_URL = "https://api.github.com"
HEADERS = {
    "User-Agent": "PythonScript",
    "Accept": "application/vnd.github+json"
}

def make_request(url):
    """Make HTTP request using urllib"""
    req = urllib.request.Request(url, headers=HEADERS)
    try:
        with urllib.request.urlopen(req) as response:
            return json.loads(response.read().decode())
    except urllib.error.HTTPError as e:
        print(f"Error accessing GitHub API: {e.code} {e.reason}")
        return None
    except urllib.error.URLError as e:
```

```
        print(f"URL Error: {e.reason}")
        return None

def get_user_contributions(username):
    """Analyze GitHub user contributions"""
    print(f"\nAnalyzing contributions for GitHub user: {username}")

    repos_url = f"{GITHUB_API_URL}/users/{username}/repos?type=all"
    repos = make_request(repos_url)

    if not repos:
        print("No public repositories found for this user.")
        return None

    contribution_data = {
        "username": username,
        "analysis_date": datetime.now().isoformat(),
        "total_repositories": len(repos),
        "total_commits": 0,
        "total_open_prs": 0,
        "total_closed_prs": 0,
        "total_issues_created": 0,
        "activity_score": 0,
        "repositories": [],
        "most_active_project": None
    }

    max_activity = 0

    for repo in repos:
        repo_name = repo["full_name"]
        print(f"\nAnalyzing repository: {repo_name}")

        repo_stats = {
            "repository": repo_name,
            "commits": get_commit_count(username, repo_name),
            "open_pull_requests": get_pull_request_count(username, repo_name, "open"),
            "closed_pull_requests": get_pull_request_count(username, repo_name, "closed"),
            "issues_created": get_issues_created_count(username, repo_name),
            "activity_score": 0
        }

        repo_stats["activity_score"] = (
            repo_stats["commits"] * 1 +
            repo_stats["open_pull_requests"] * 2 +
            repo_stats["closed_pull_requests"] * 3 +
            repo_stats["issues_created"] * 1.5
        )

        contribution_data["total_commits"] += repo_stats["commits"]
        contribution_data["total_open_prs"] += repo_stats["open_pull_requests"]
        contribution_data["total_closed_prs"] += repo_stats["closed_pull_requests"]
        contribution_data["total_issues_created"] += repo_stats["issues_created"]
        contribution_data["activity_score"] += repo_stats["activity_score"]

        contribution_data["repositories"].append(repo_stats)

        if repo_stats["activity_score"] > max_activity:
            max_activity = repo_stats["activity_score"]
            contribution_data["most_active_project"] = {
                "name": repo_name,
                "commits": repo_stats["commits"],
                "pull_requests": repo_stats["open_pull_requests"] +
repo_stats["closed_pull_requests"]
            }

    return contribution_data

def get_commit_count(username, repo_name):
    """Get commit count for a repository"""
    print(f"Counting commits in {repo_name}...")
    url = f"{GITHUB_API_URL}/repos/{repo_name}/commits?author={username}"
    return len(make_request(url) or [])

def get_pull_request_count(username, repo_name, state):
    """Get PR count for a repository"""
    print(f"Counting {state} pull requests in {repo_name}...")
    url = f"{GITHUB_API_URL}/repos/{repo_name}/pulls?state={state}&creator={username}"
```

```
    return len(make_request(url) or [])

def get_issues_created_count(username, repo_name):
    """Get issues count for a repository"""
    print(f"Counting issues created in {repo_name}...")
    url = f"{GITHUB_API_URL}/repos/{repo_name}/issues?creator={username}"
    return len(make_request(url) or [])

def save_to_json(data, filename):
    """Save data to JSON file"""
    with open(filename, 'w') as f:
        json.dump(data, f, indent=2)
    print(f"\nResults saved to {filename}")

def main():
    username = input("Enter GitHub username: ").strip()
    contribution_data = get_user_contributions(username)

    if contribution_data:
        print("\n=== Contribution Summary ===")
        print(f"User {username} has contributed to {contribution_data['total_repositories']}
repositories.")
        print(f"Total commits: {contribution_data['total_commits']}")
        print(f"Open pull requests: {contribution_data['total_open_prs']}")
        print(f"Closed pull requests: {contribution_data['total_closed_prs']}")
        print(f"Issues created: {contribution_data['total_issues_created']}")
        print(f"Activity score: {contribution_data['activity_score']:.1f}")

        if contribution_data['most_active_project']:
            proj = contribution_data['most_active_project']
            print(f"\nMost active project: {proj['name']}")
            print(f" - Commits: {proj['commits']}")
            print(f" - Pull requests: {proj['pull_requests']}")

        save_to_json(contribution_data, "github_contribution.json")

if __name__ == "__main__":
    main()
```

Результат выполнения:

```
Enter GitHub username: yeazyyy

Analyzing contributions for GitHub user: yeazyyy

Analyzing repository: yeazyyy/OOTPISP-2024
Counting commits in yeazyyy/OOTPISP-2024...
Counting open pull requests in yeazyyy/OOTPISP-2024...
Counting closed pull requests in yeazyyy/OOTPISP-2024...
Counting issues created in yeazyyy/OOTPISP-2024...

=== Contribution Summary ===
User yeazyyy has contributed to 1 repositories.
Total commits: 5
Open pull requests: 0
Closed pull requests: 0
Issues created: 0
Activity score: 5.0

Most active project: yeazyyy/OOTPISP-2024
 - Commits: 5
 - Pull requests: 0

Results saved to github_contribution.json
```

Вывод: научился работать с Github API, приобрёл практические навыки написания программ для работы с REST API или GraphQL API.