Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Пермский национальный исследовательский

политехнический университет»

(ПНИПУ)

Кафедра вычислительной математики и механики

Лабораторная работа

по дисциплине: «Информатика в приложении к отрасли»

по теме: «Непрерывная интеграция, анализ качества кода»

Выполнил: Галашов Данил

Студенты группы: ЦТУ-20-3б

Пермь

2022 г.

**Порядок выполнения задания**

Выполнение работы удобно разбить на следующие этапы:

1. Создание репозитория на сервисе Github
2. Написание тестовой программы (взять из предыдущей лабы)
3. Разобраться с тем, как создавать список установленных модулей requrements.txt и устанавливать модули по списку командой

Для создания списка установленных модулей используется команда:

pip freeze > requirements.txt

Для установки модулей по списку используется команда:

pip install –r requirements.txt

1. Разобраться с тем, как автоматизировать запуск тестов и генерацию отчета по ним (в виде 1 файла html)

Для запуска тестов используется команда:

pytest

Для запуска тестов и создания отчета в виде 1 html файла используется команда:

pytest --html=tmp\report.html --self-contained-html

1. Разобраться с тем, как проводить анализ покрытия тестами кода и генерировать отчеты в виде html и xml

Для анализа покрытия кода используется команды:

coverage run -m pytest

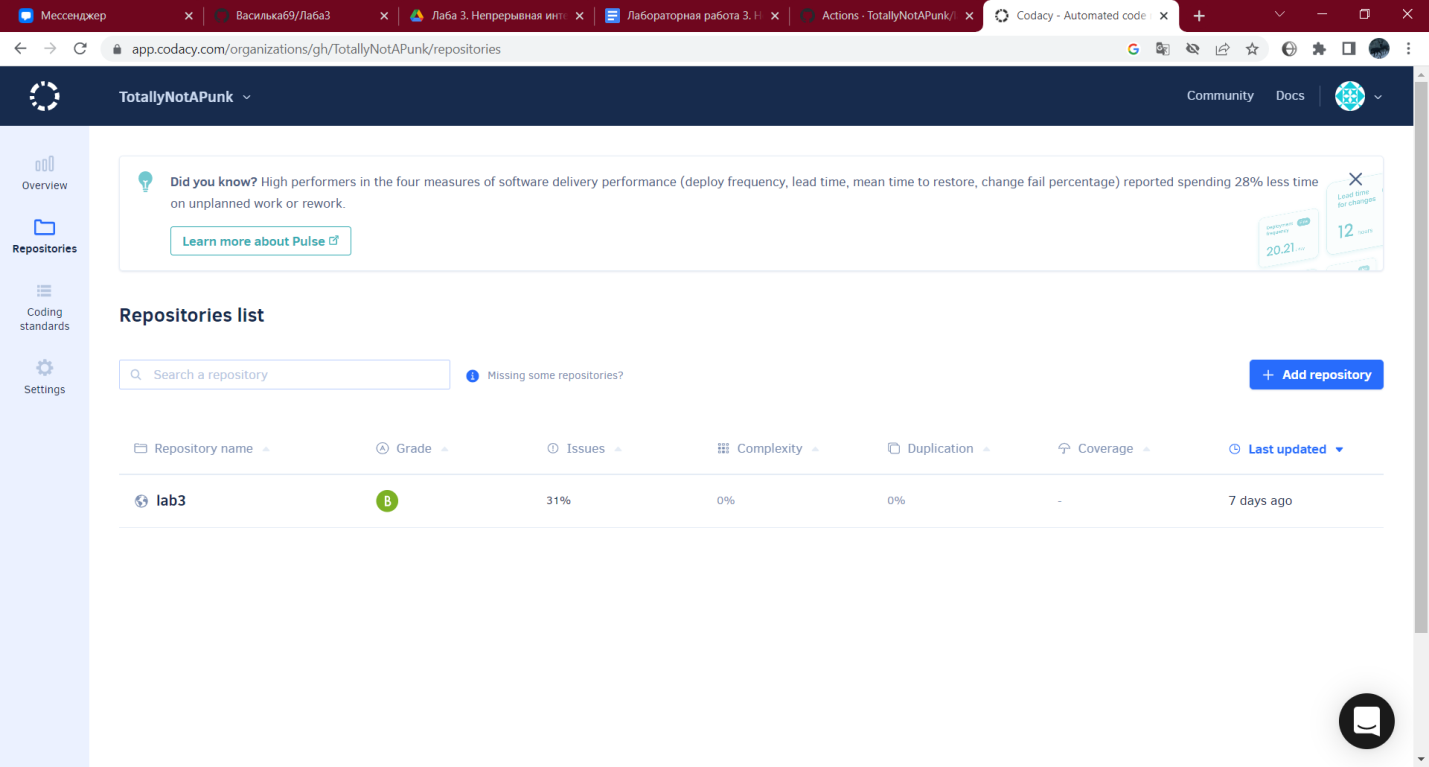
Для анализа покрытия кода и генерации отчета в форме html используется команда:

coverage html --directory tmp\coverage\_result

Для анализа покрытия кода и генерации отчета в форме xml используется команда:

coverage xml -o tmp\coverage\_result.xml

1. Интеграция анализаторов кода Codacy в репозиторий



1. Разобраться с тем, как выгружать xml отчет покрытия кода тестами на сервис Codacy

Выгрузки xml отчета на сервис Codacy необходимо указать токен API вашего проекта с помощью команды:

export CODACY\_PROJECT\_TOKEN=<your project API token>

Затем, Запустите Codacy Coverage Reporter, указав относительный путь к отчету о покрытии для загрузки командой:

bash <(curl -Ls https://coverage.codacy.com/get.sh) report -r <coverage report file name>

1. Реализовать шаги по тестированию кода в Github Actions
2. name: CICD  
   on:  
    push:  
    branches: [main]  
   jobs:  
    test-and-coverage-package:  
    runs-on: ubuntu-latest  
    env:  
    COVERAGE\_HTML\_REPORT\_PATH: /tmp/coverage\_html\_report  
    COVERAGE\_XML\_REPORT\_PATH: /tmp/coverage\_xml\_report.xml  
    PYTEST\_REPORT\_PATH: /tmp/pytest\_report.html  
    steps:  
    - name: "Checkout repository"  
    uses: actions/checkout@v3  
    - name: "Install python"  
    uses: actions/setup-python@v4  
    with:  
    python-version: '3.10'  
    architecture: 'x64'  
    - name: "Install project dependencies"  
    run: pip install -r requirements.txt  
    - name: "Run tests and code coverage analysis"  
    run: coverage run -m pytest --html=${{ env.PYTEST\_REPORT\_PATH }} --self-contained-html  
    - name: "Generate code coverage html report"  
    run: coverage html --directory ${{ env.COVERAGE\_HTML\_REPORT\_PATH }}  
    - name: "Archive pytest and coverage reports"  
    uses: actions/upload-artifact@v3  
    if: ${{!cancelled()}}  
    with:  
    name: "Test and coverage reports"  
    path: |  
    ${{ env.PYTEST\_REPORT\_PATH }}  
    ${{ env.COVERAGE\_HTML\_REPORT\_PATH }}  
    - name: "Generate code coverage xml report"  
    run: coverage xml -o ${{ env.COVERAGE\_XML\_REPORT\_PATH }}  
    - name: "Push coverage xml report to codacy"  
    uses: codacy/codacy-coverage-reporter-action@v1  
    if: ${{!cancelled()}}  
    with:  
    project-token: ${{ secrets.CODACY\_PROJECT\_API\_TOKEN }}  
    coverage-reports: ${{ env.COVERAGE\_XML\_REPORT\_PATH }}
3. Разобраться с тем, как запускать в Github Actions Job, если предыдущая Job завершилась без ошибок

publish-package-as-artifact:  
 needs: ["test-and-coverage-package"]

1. Реализовать выгрузку кода в виде артефакта при успешном прохождении тестов

publish-package-as-artifact:  
 needs: ["test-and-coverage-package"]  
 runs-on: ubuntu-latest  
 steps:  
 - name: "Checkout repository"  
 uses: actions/checkout@v3  
 - name: "Publish package as artifact"  
 uses: actions/upload-artifact@v3  
 with:  
 name: "Built package"  
 path: "src"