# Лабораторная работа №10 Испытание ПО методом регрессионного тестирования

1. Цель работы
   1. Освоить процесс применения регрессионного тестирования.
   2. Изучить работу с GitHub Actions для автоматического тестирования.
2. Литература
   1. Игнатьев, А. В. Тестирование программного обеспечения : учебное пособие для вузов / А. В. Игнатьев. — 4-е изд., стер. — Санкт-Петербург : Лань, 2025. — 56 с. — ISBN 978-5-507-50858-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/481331> — Режим доступа: для авториз. пользователей.
   2. GitHub Actions. Создание и тестирование для .NET – Текст : электронный // Документация по GitHub, 2025. – URL: https://docs.github.com/ru/actions/use-cases-and-examples/building-and-testing/building-and-testing-net
3. Подготовка к работе
   1. Повторить теоретический материал (см. п.2).
   2. Изучить описание лабораторной работы.
4. Основное оборудование
   1. Персональный компьютер.
5. Задание
   1. Создать новый проект библиотеки классов, в котором реализовать статический метод для валидации пароля в соответствии со следующими правилами: длина пароля должна составлять не менее 8 символов, и он должен содержать хотя бы одну цифру и латинский символ.
   2. Разработать модульные тесты для метода, рассмотрев как позитивные, так и негативные входные данные.
   3. Автоматическое регрессионное тестирование
      1. В каталоге решения создать папку .github, в которой создать папку workflows, с файлом autotests.yml
      2. В файле указать

name: .NET Regression Tests

on: # Запускается при push и pull request в ветку master

push:

branches: [ "master" ]

pull\_request:

branches: [ "master" ]

jobs:

build-and-test:

runs-on: ubuntu-latest # ОС виртуальной машины

steps:

- name: Checkout repository

uses: actions/checkout@v4 # Клонирование репозитория на ВМ

- name: Setup .NET

uses: actions/setup-dotnet@v4

with:

dotnet-version: '8.0.x' # Установка .NET 8 на ВМ

- name: Restore dependencies

run: dotnet restore # Восстановление зависимостей

- name: Build solution # Построение решения

run: dotnet build --no-restore --configuration Release

- name: Run tests # Запуск тестов

run: dotnet test --no-build --configuration Release --verbosity normal

* + 1. Создать репозиторий GitHub из решения.
    2. В удаленном репозитории открыть раздел Actions и открыть там последнюю операцию. Изучить содержимое страницы и вывод результатов выполнения action
  1. Отображение результатов автоматического тестирования
     1. Изменить содержимое autotests.yml

- name: Run tests and generate TRX log # Запуск тестов

run: dotnet test --no-build --configuration Release --logger "trx;LogFileName=test\_results.trx"

- name: Test Report # Сохранение отчета

uses: dorny/test-reporter@v2

if: success() || failure() # запускает этот шаг в любом случае

with:

name: Test results

path: "\*\*/\*.trx" # Путь сохранения результатов теста

reporter: dotnet-trx # Формат результата

* + 1. Сохранить изменения в репозитории, изучить вывод последней задачи в разделе Actions
  1. Добавление бэйджа
     1. Добавить файл readme.md в исключения для Github Actions в разделе on push:

paths-ignore:

- '\*\*/README.md'

* + 1. Нажмите кнопку «…» и выберите там опцию «Create status badge»
    2. Скопируйте код и вставьте его в файл README.md вашего репозитория.
  1. Измените код метода валидации пароля, добавьте условие наличия буквы в верхнем и нижнем регистре и спецсимвола. Сохраните изменения в репозитории, проверьте результаты выполнения тестов в разделе Actions.
     1. Измените тесты проекта так, чтобы они снова успешно выполнялись.

1. Порядок выполнения работы
   1. Повторить теоретический материал п. 3.1;
   2. Выполнить задания п.5.1-5.6
   3. Ответить на контрольные вопросы п.8;
   4. Заполнить отчет п. 7.
2. Содержание отчета
   1. Титульный лист;
   2. Цель работы;
   3. Ответы на контрольные вопросы п. 6.3;
   4. Вывод по проделанной работе.
3. Контрольные вопросы
   1. Что такое регрессионное тестирование?
   2. Для чего используются GitHub Actions?