МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации

**

**ОТЧЁТ**

**по лабораторной работе №1**

**«**Работа с Git и GitHub**»**

**по дисциплине: «***Программирование***»**

Выполнил:Проверил:

Студент гр. «АБс-324», «АВТФ» *Ассистент ЗИ*

*Клаус В. А. Исаев Г. А.*

«\_\_» \_\_\_\_\_\_ 2024 г.«\_\_\_» \_\_\_\_\_\_ 2024 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (подпись)

Новосибирск 2024

**Цели и задачи работы**: изучение платформы GitHub и практическое использование Git для хранения, обновления и распространения исходного кода проекта.

**Задание к работе**: Разработать алгоритм решения задачи по индивидуальному заданию:

Геометрические фигуры. Реализовать функции вычисления параметров для двух-трех следующих геометрических фигур (по одной фигуре на студента):

Прямоугольник – периметр, площадь, длина диагонали;

**Методика выполнения работы**:

1. Разработать алгоритм решения задачи по индивидуальному заданию.
2. Написать и отладить программу решения задачи.
3. Протестировать работу программы на различных исходных данных.
4. Продемонстрировать работу с Git и GitHub на примере написанной программы.
5. По запросу преподавателя быть готовым модифицировать/добавить функционал программы.
6. Ответить на теоретические вопросы к лабораторной работе на выбор преподавателя.

**Код программы с комментариями:**

#include <iostream> // подключение библиотек

#include <math.h>

using namespace std;

int main(){

setlocale(LC\_ALL, "Rus");

double a, b; // инициализация переменных

do { // цикл do-while проверки ввода переменных

cout << "Введите 2 стороны прямоугольника(стороны не должны быть равны нулю и отрицательному числу): ";

cin >> a >> b;

}

while (a<=0 || b<=0);

cout << "Периметр прямоугольника: " << 2\*a+2\*b << endl; // вывод результата

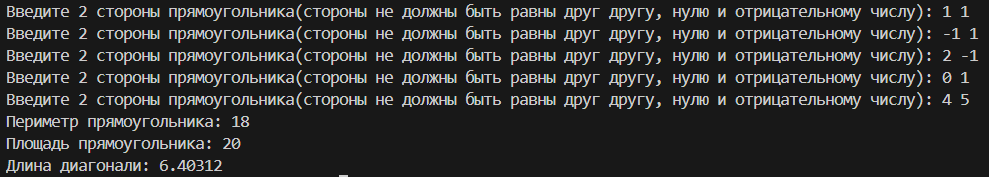
cout << "Площадь прямоугольника: " << a\*b << endl;

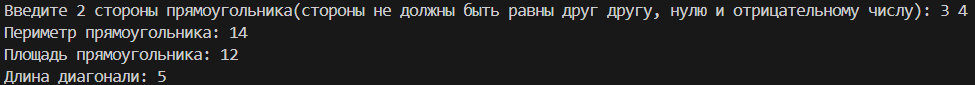
cout << "Длина диагонали: " << sqrt(a\*a+b\*b) << endl;

return 0;

}

**Результат работы программы:**





**Полное описание всех этапов процесса работы с Git и GitHub на примере написанной программы:**

1. Создание пустого репозитория в GitHub.
2. Клонирование репозитория в рабочую папку, открытие проекта в текстовом редакторе(Visual Studio Code).
3. Создание отдельной ветки “Vadim” и файла rectangle.cpp, в котором будет оформляться код. Создаём отдельную ветку, чтобы избежать помех в работе в команде.
4. Оформление кода, доработка кода, сохранение и отладка.
5. Загрузка отдельной ветки “Vadim” с файлом rectangle.cpp в репозиторий. Загрузка осуществлялась через терминал текстового редактора.
6. Создание файла README.md в основной ветке “main”. Оформление файла, сохранение и загрузка в репозиторий.
7. Выполнение слияния дополнительной ветки “Vadim” в основную “main”. Сохранение и загрузка в репозиторий.

**Функционал:**

При работе с GitHub и Git пользовались графическим интерфейсом GitHub и командами в терминале текстового редактора Visual Studio Code:

1. git init - создает новый репозиторий.

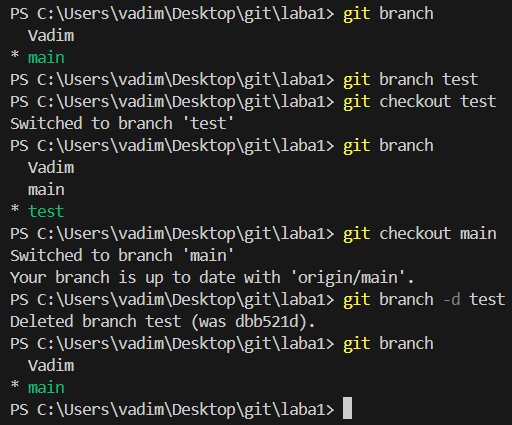


1. git branch - выводит список веток.

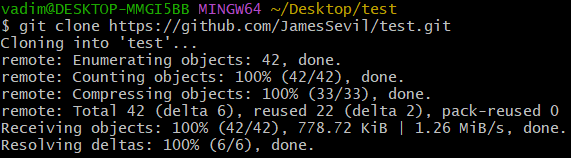
git branch название\_ветки - создает новую ветку.

git checkout название\_ветки - осуществляет переход по ветке.

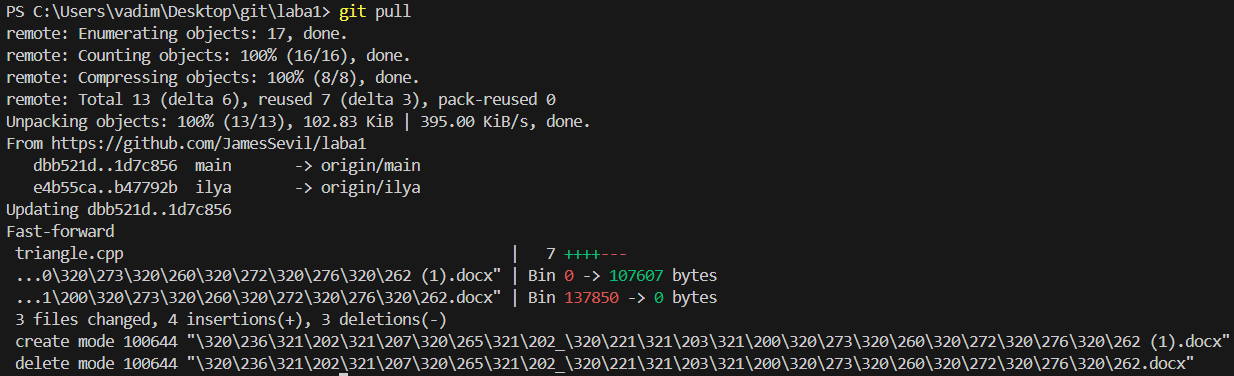
git branch -d название\_ветки - удаляет ветку.



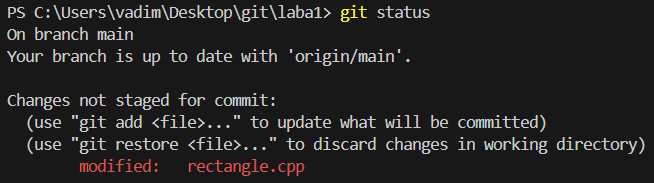
1. git clone ссылка - клонировать существующий репозиторий.



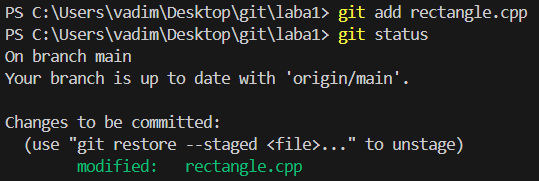
1. git pull - обновляет информацию в клонированном репозитории из облачного.



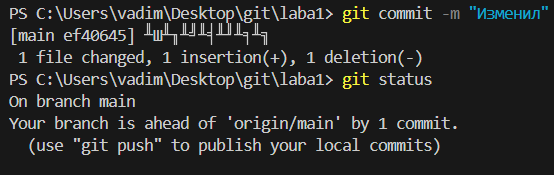
1. git status - отображает состояние рабочего каталога и раздела проиндексированных файлов.



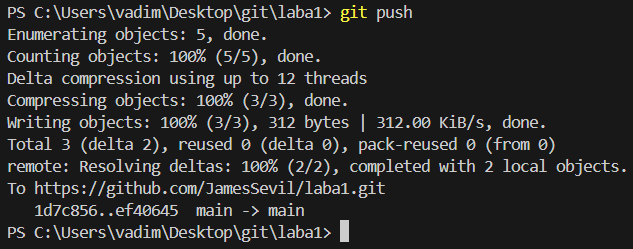
1. git add название\_файла(. - все файлы) - добавляет изменения из рабочего каталога в раздел проиндексированных файлов.



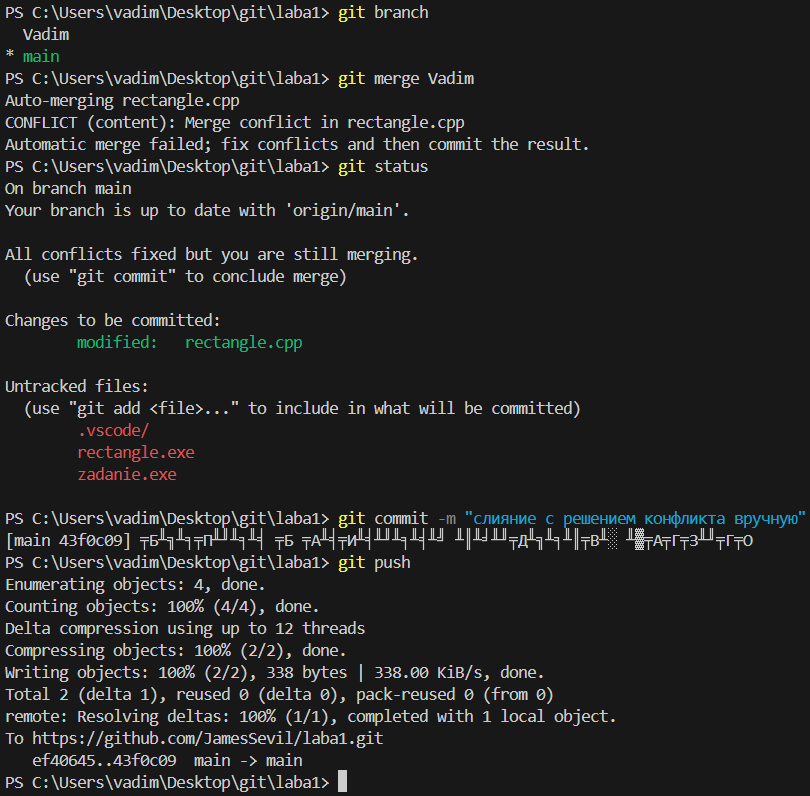
1. git commit -m “название\_файла” - фиксирует изменения с комментарием.



1. git push - загружает изменения в облачный репозиторий.



1. git merge - осуществляет слияние веток(сливаемая ветка сохраняется).



**Вывод:**

В результате работы были выполнены все поставленные цели и задачи: изучена платформа GitHub и практическое использование Git для хранения, обновления и распространения исходного кода проекта. Научились работать над проектом в команде.

**Ссылка на репозиторий GitHub:** https://github.com/JamesSevil/laba1