**Вариант 16**

1. Решить задачу, используя функцию. Составить программу для нахождения общего количества заданной буквы в трёх заданных предложениях. (Определить функцию для расчёта количества некоторой буквы в предложении.)

3. Описать структуру с именем PRODUCT, содержащую поля:

*наименование, цена, дата производства, срок годности, количество, производитель.*

Написать функции:

• создания массива продуктов не менее 12 записей (структур) сведений о продуктах (ввод с клавиатуры);

• вывести сведения с сортировкой по дате производства по возрастанию;

• вывести сведения о товарах, срок годности которых истекает через двое суток;

• вывести сведения о количестве товаров для каждого производителя;

• вывести сведения о товаре с максимальной ценой, срок годности которого не истек.

Все необходимые данные для функций должны передаваться в качестве их параметров. Использование глобальных параметров не допускается. Создать проект, который демонстрирует работу всех функций.

4. «Книга»: *название; автор (фамилия; имя); год выхода; издательство; себестоимость; цена; прибыль.*

Вывести данные про книги авторов, фамилия которых начинается с буквы “К”.

ЗАДАНИЕ 1. КОНСОЛЬНЫЕ ПРИЛОЖЕНИЯ В REPL.IT

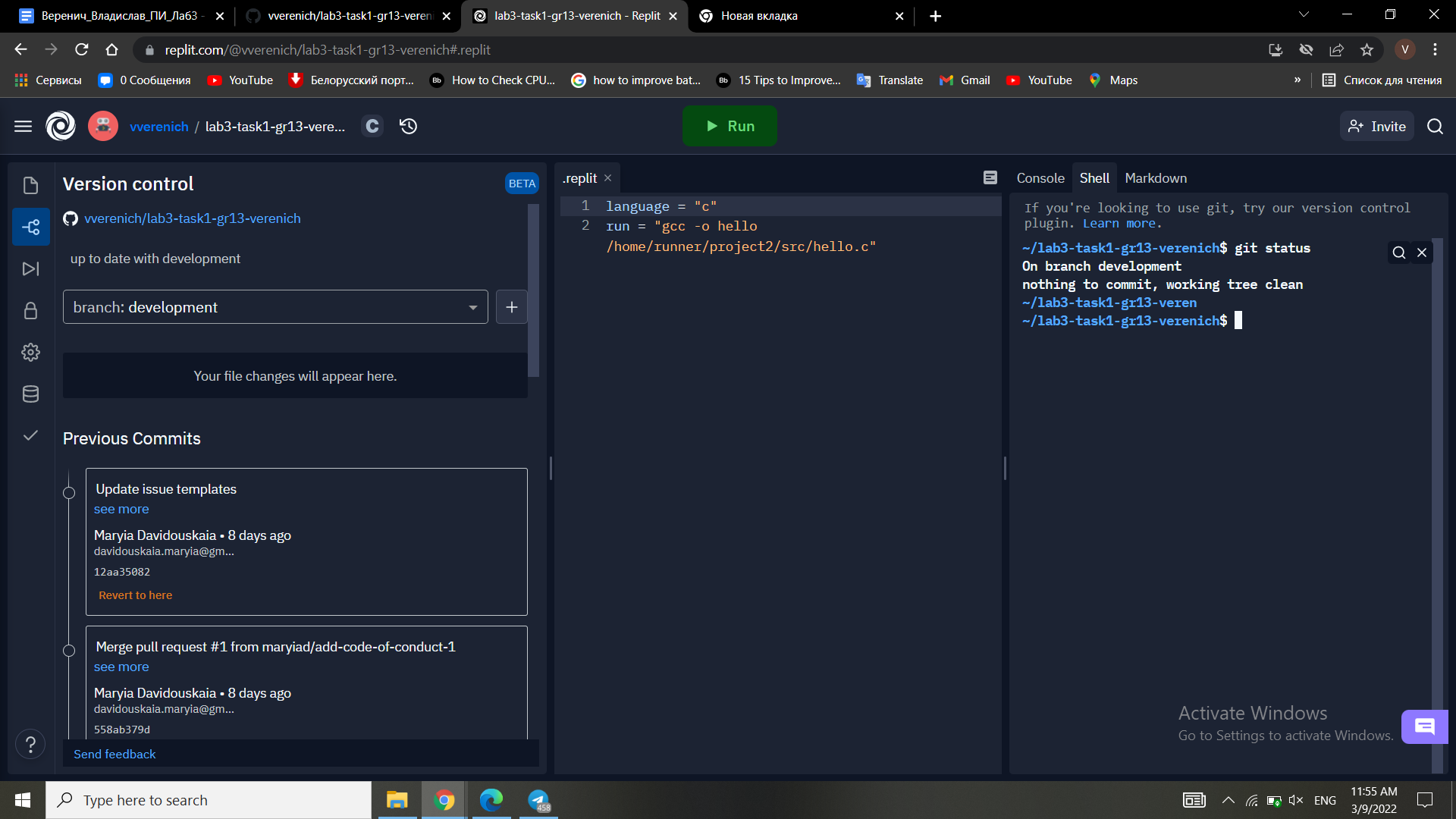
Для решения задания 1 необходимо использовать:

• шаблон репозитория проекта <https://github.com/maryiad/project-forlab3-tp>

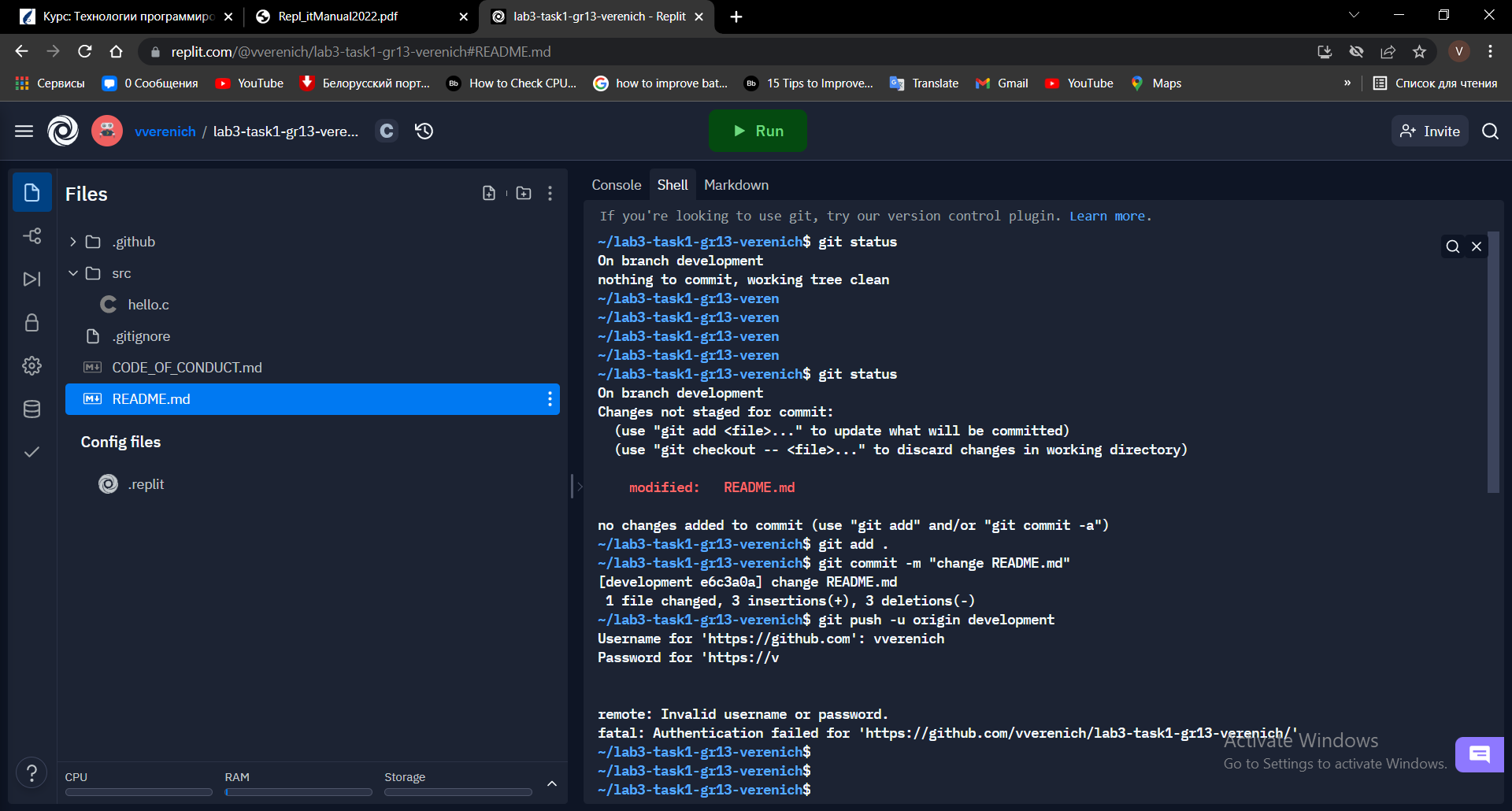
• онлайн среду разработки https://repl.it/, в которой авторизоваться с учетной записью на github.

Порядок выполнения задания:

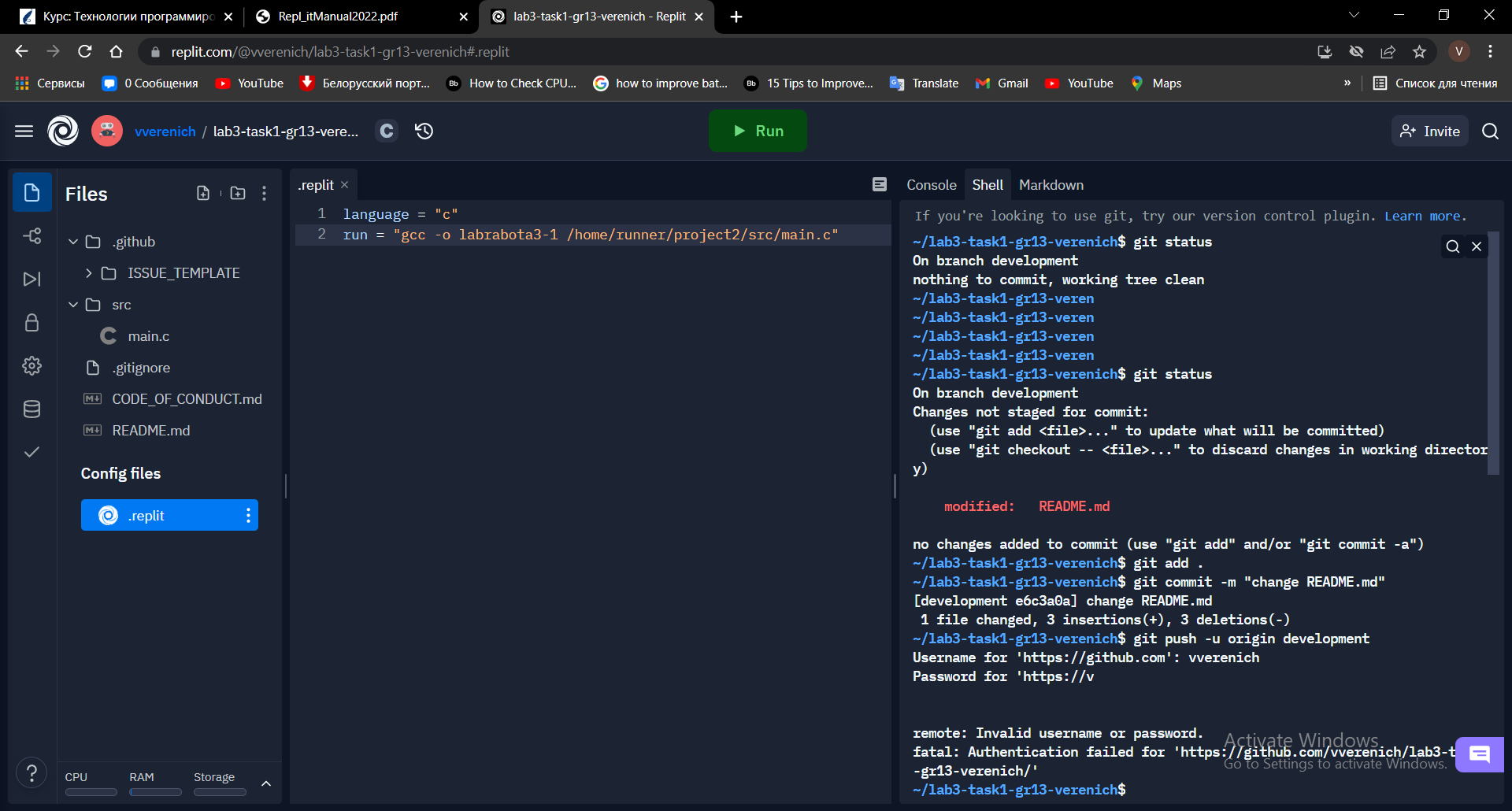
1. Познакомьтесь с руководством Руководство по созданию проекта в Github и Repl.it по созданию репозитория и активации аккаунта для сервиса repl.it.
2. Создайте репозиторий из шаблона репозитория согласно рекомендациям руководства из п. 1 текущего задания. В качестве имени репозитория введите имя согласно шаблону lab3-task1-gr13-petrov, указав корректно номер своей группы и фамилию вместо petrov.
3. Импортируйте проект в repl.it согласно рекомендациям из руководства в п. 1 текущего задания. Далее все изменения выполняются в среде repl.it и публикуются на github.
4. Создать ветку в репозитории для данной задачи согласно Требования к репозиториям, например development, и переключиться в неё. Все изменения по проекту выполнять в новой ветке!



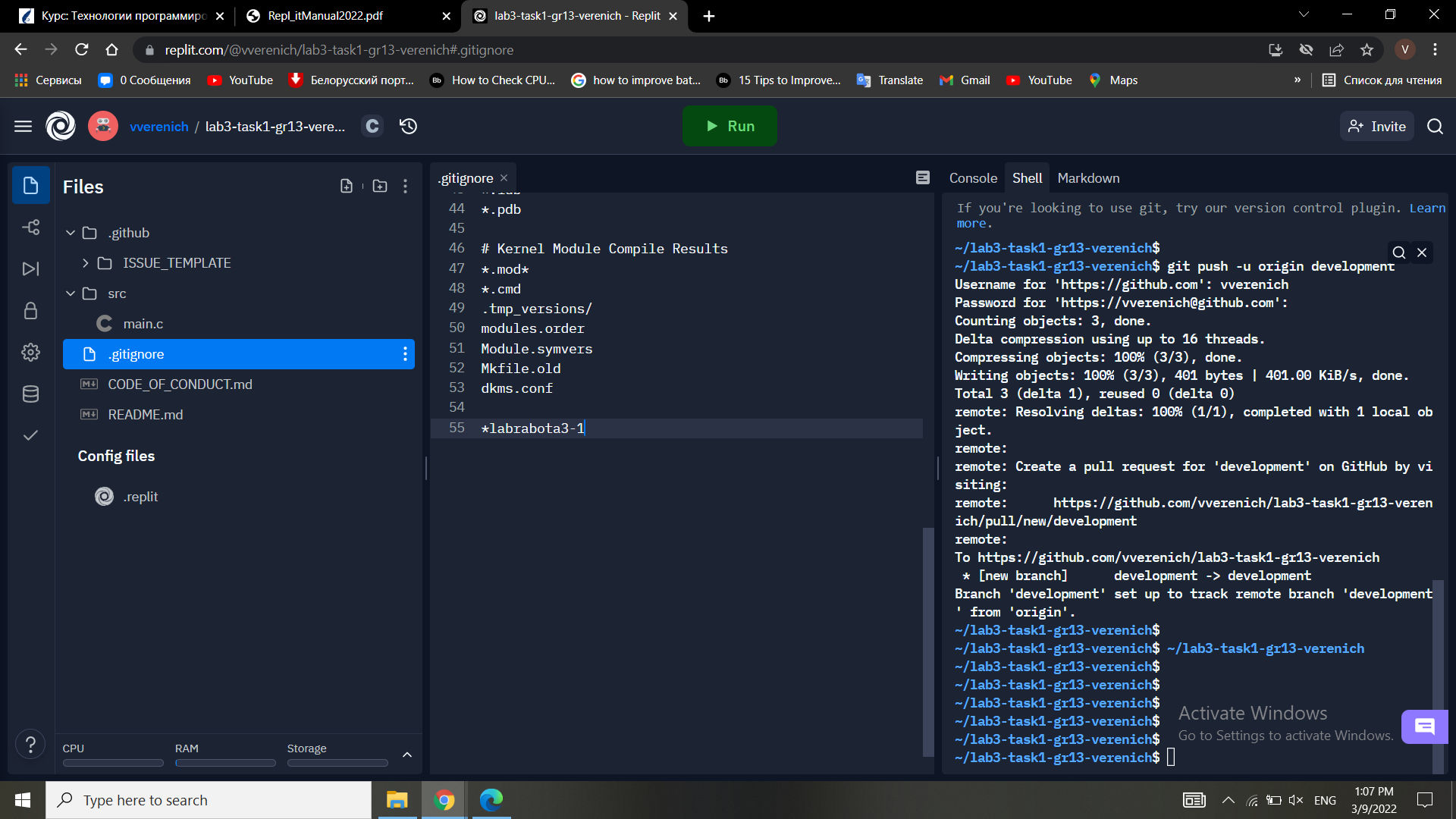
1. Отредактируйте файл Readme в созданном проекте, изменив текст в разделах Overview, Usage, Building. Добавьте в Readme свою фамилию, имя и номер подгруппы вместо текста {add your last name, first name and group number}. Закоммитить изменения.



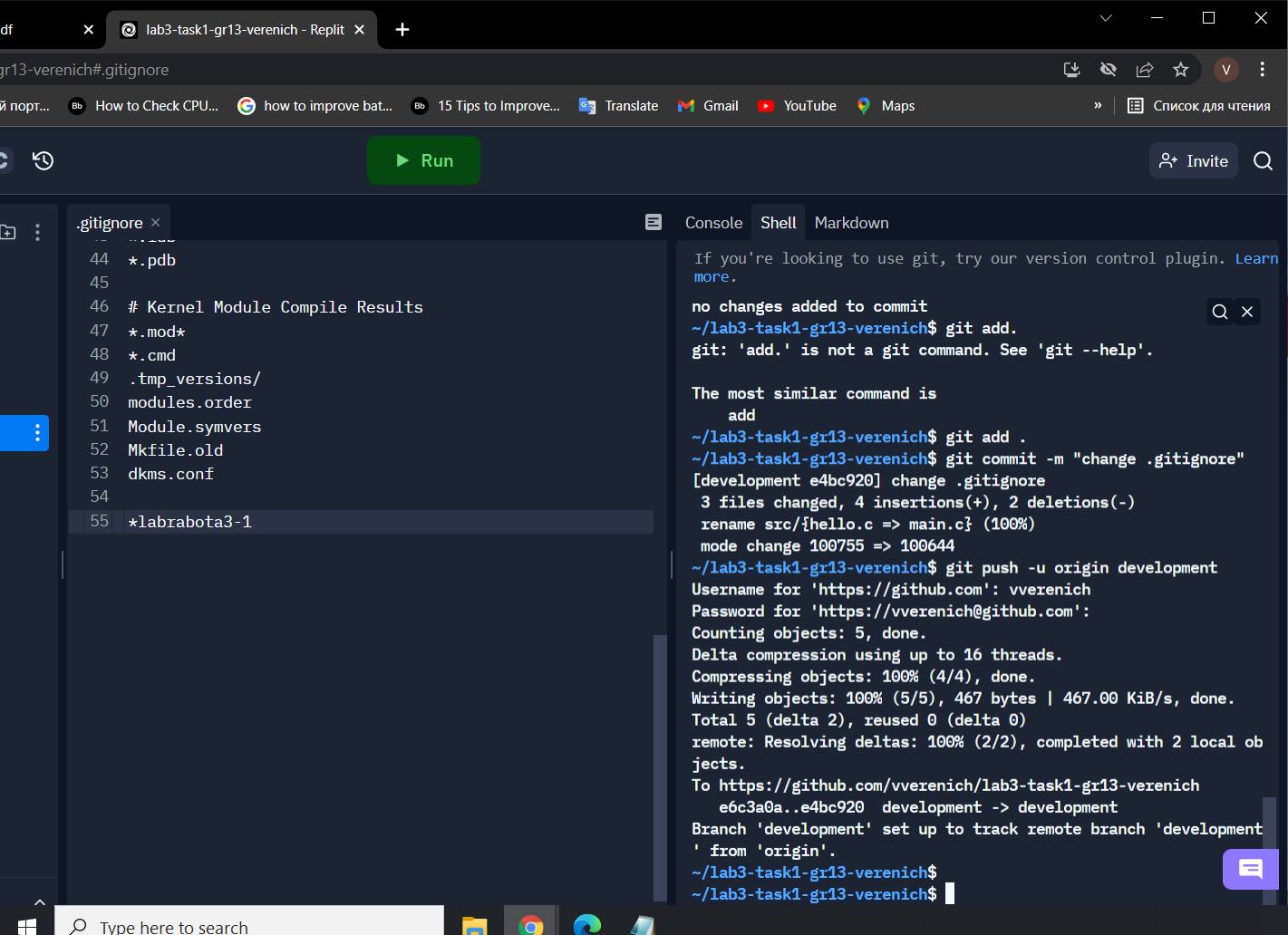
1. Переименовать файл hello.c, например в main.c.
2. Отредактировать файл .replit и указать название файла, который компилируется, например main.c, и имя бинарного файла, например labrabota3-1, а так же путь к этому файлу. Т.е. отредактировать строку run = "gcc -o hello /home/runner/project2/src/hello.c" к виду, например, "gcc -o labrabota3-1 /home/runner/lab3-task1-gr13-petrov/src/main.c", где labrabota3-1 — это исполняемый файл, /home/runner/lab3-task1-gr13-petrov/src/main.c — пусть к исходному файлу с учётом, что /home/runner/ — домашний каталог, lab3-task1-gr13- petrov/ — каталог проекта из Вашего репозитория, совпадающий с названием репозитория. Проверить путь к текущему каталогу можно командой pwd на вкладке Shell в repl.it (см. п. 16 в руководстве из пункта 1).



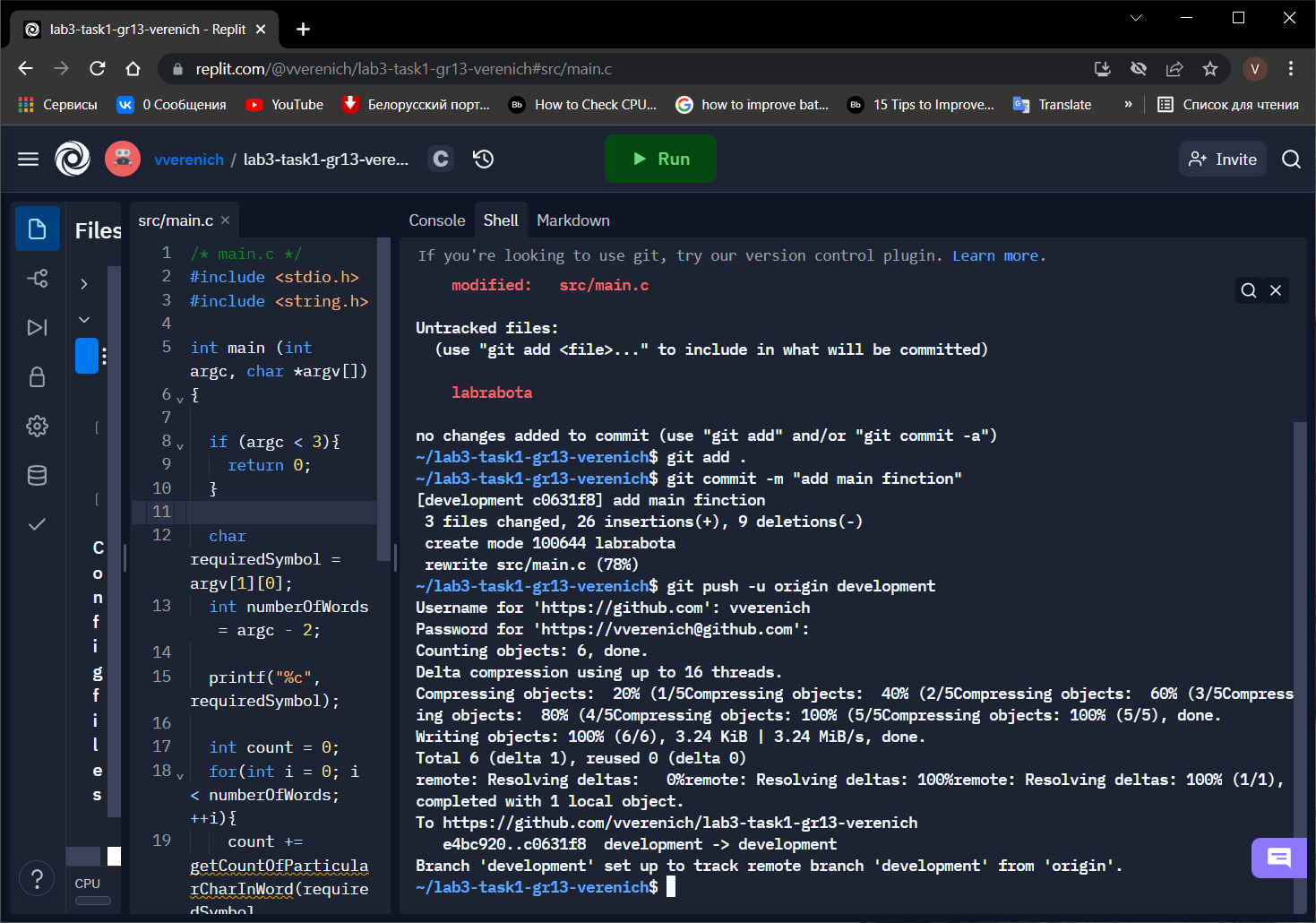
1. Отредактировать .gitignore и указать имя бинарного файла, который должен получиться в результате labrabota3-1 и который не должен быть опубликован в репозиторий, т. е. Проигнорирован при публикации измений.



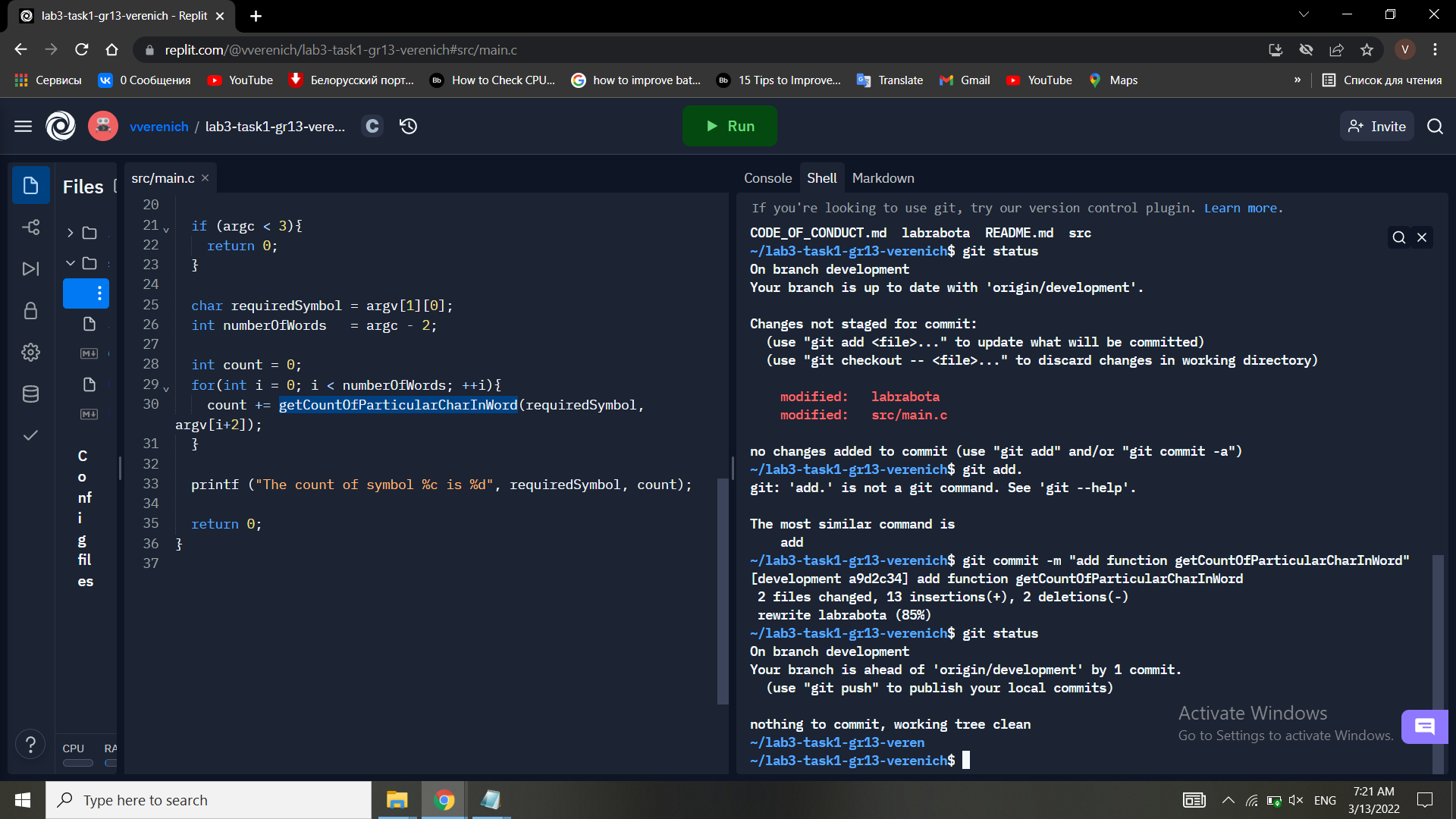
1. Закоммитить изменения и опубликовать их в репозиторий на github.



1. Изучить документацию Стандарты кодирования для оформления кода и при разработке программы применить правила оформления кода согласно стандартам.
2. Коммитить изменения в репозиторий пошагово, например добавили новую функцию, проверили, коммитим.

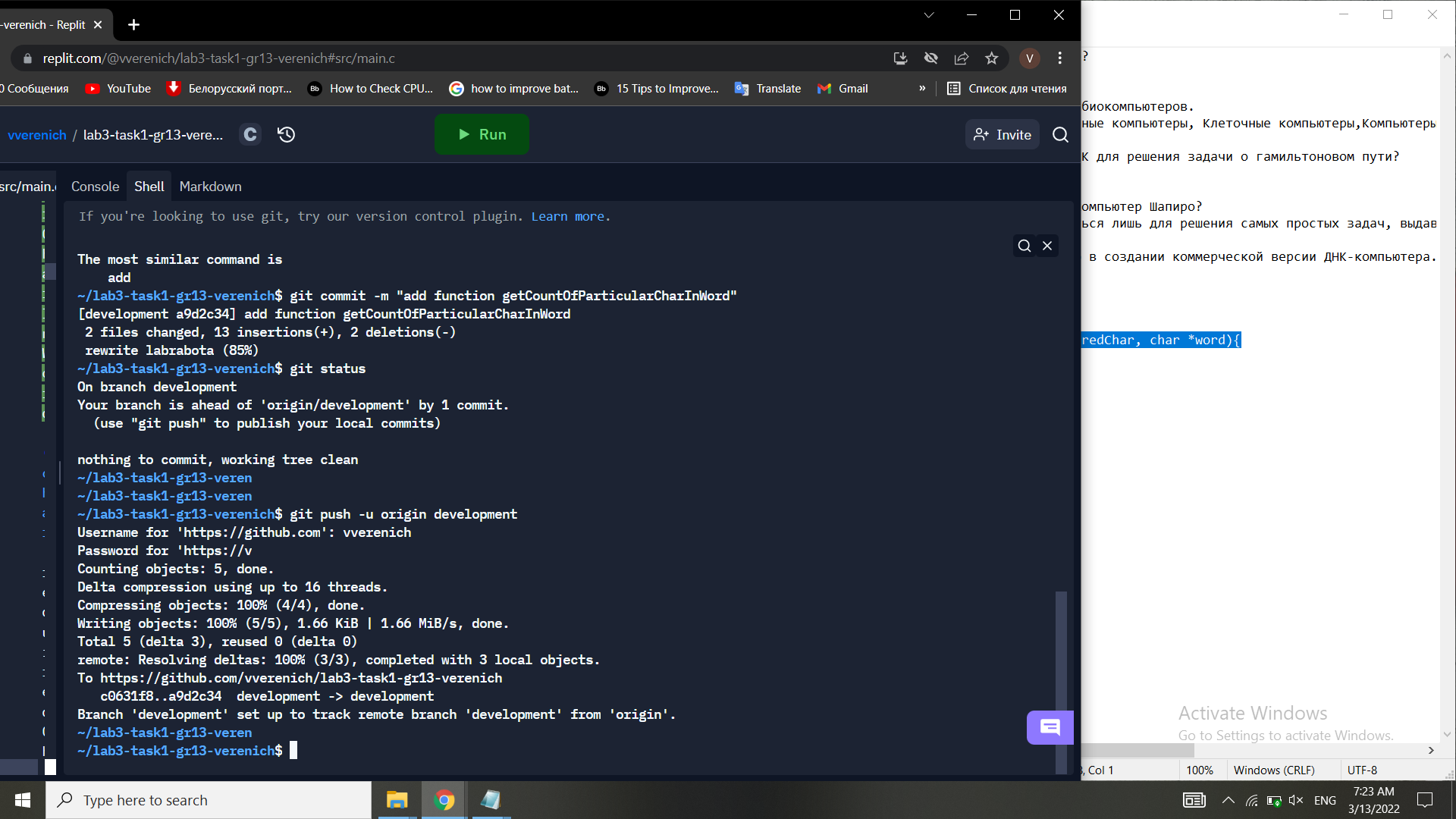


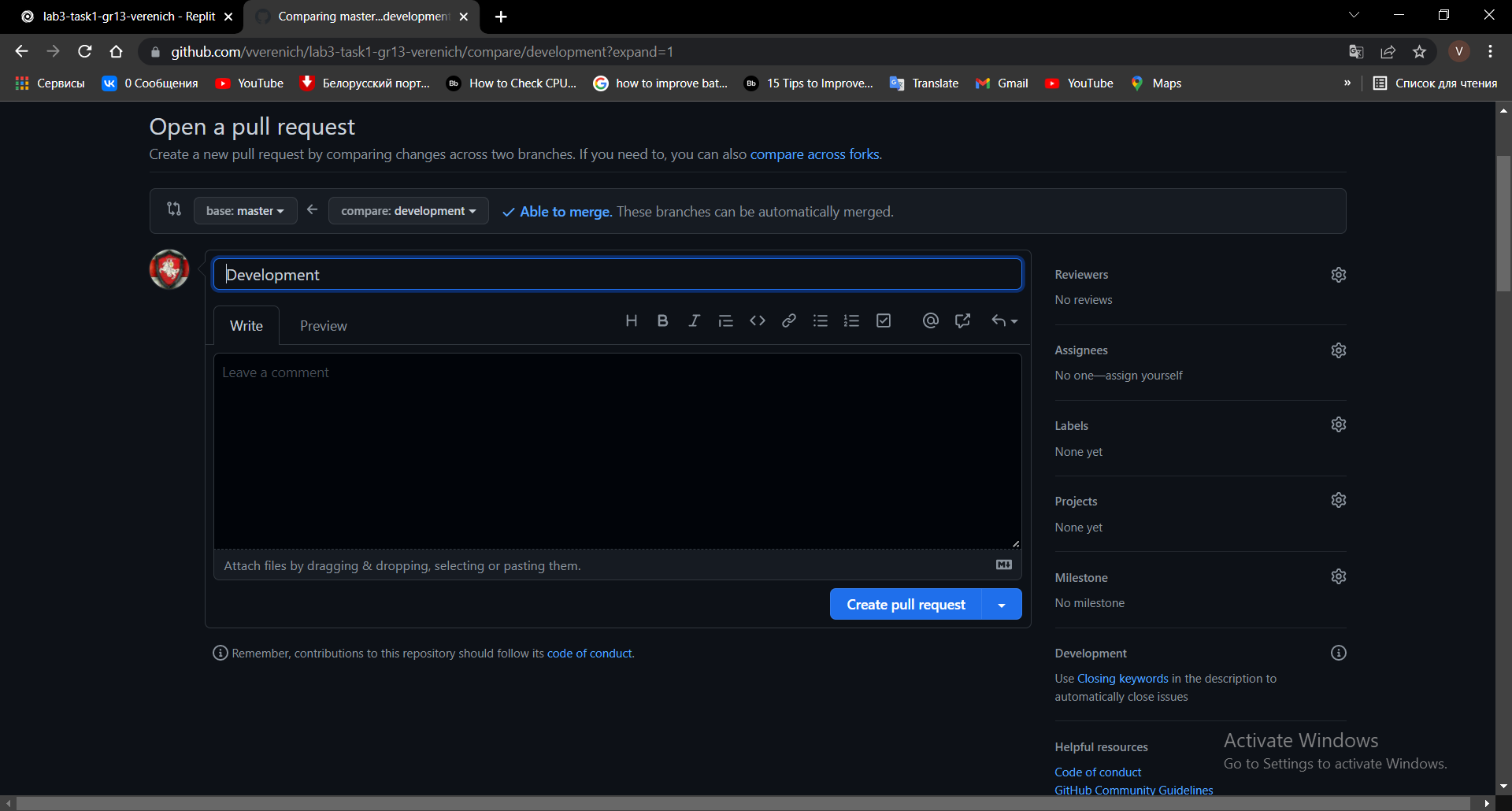


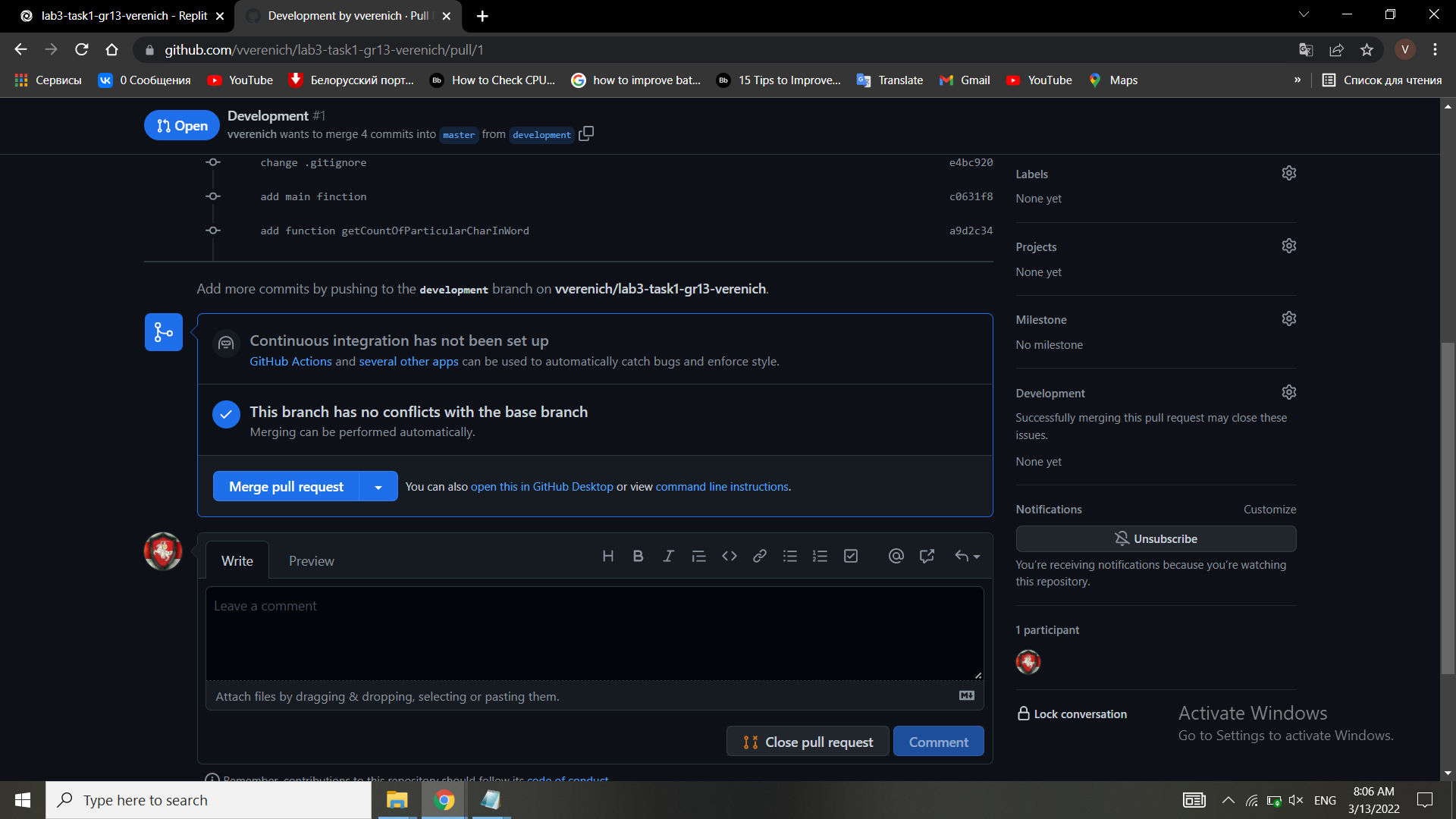


1. В каждом задании указано, какую подзадачу должна решать разрабатываемая функция. Исходные данные программа получает через аргументы командной строки. Если в задании необходимо в качестве исходных данных использовать массивы, их нужно сформировать с помощью генератора псевдослучайных чисел (размерности массивов программа получает через аргументы командной строки). На экран вывести исходные данные и полученные результаты.
2. Провести тесты (контрольный расчет) для проверки работоспособности алгоритма решения задачи и включить их в отчёт.

| № п/п | Входные данные | Ожидаемые выходные данные | Тест пройден |
| --- | --- | --- | --- |
| 1 | r reading any book is very interesting and exciting event | The count of symbol r is 3 | да |
| 2 | a You wanna me reading it? I can’t believe you are asking about it | The count of symbol a is 7 | да |
| 3 | y Do you love me? What do you really want from me? | The count of symbol y is 3 | да |
| 4 | t Party for everybody, dance! | The count of symbol t is 1 | да |

1. После завершения работы над заданием опубликовать изменения в репозиторий, т. е. выполнить git push. 
2. Переключиться в репозиторий на github и, используя pull request, слить изменения из ветки development в ветку master (см. Как сделать pull request на Github).





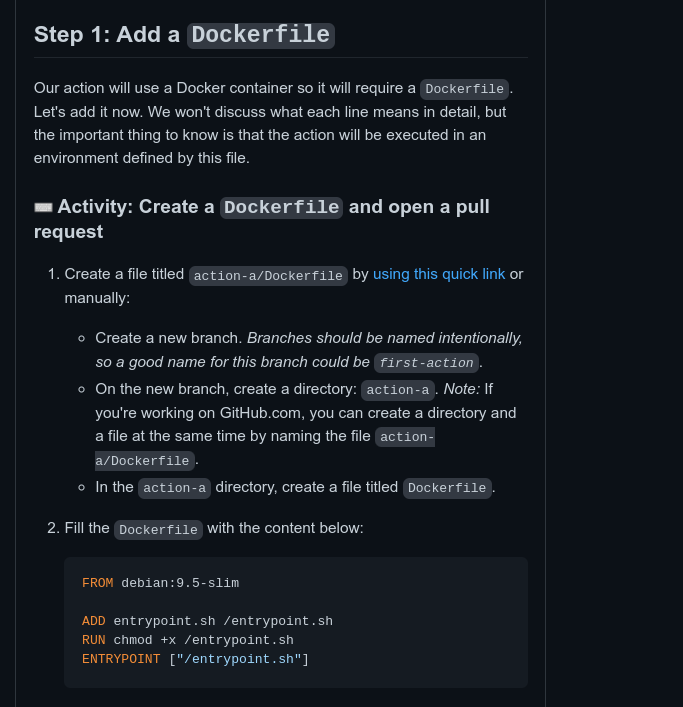
**Дополнительные материалы:**

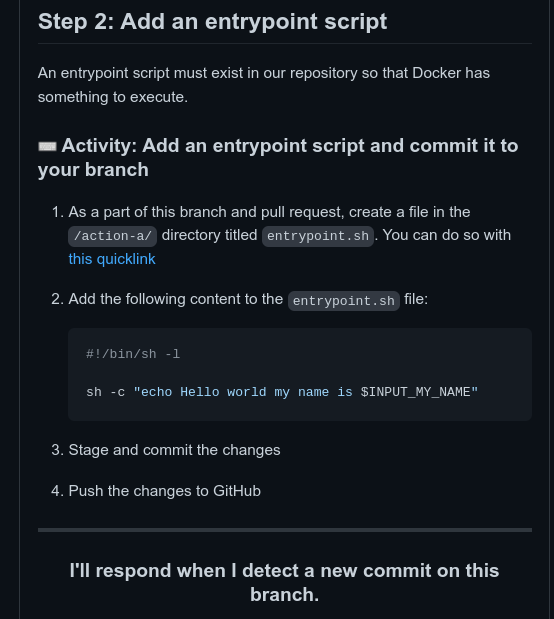
Познакомьтесь с кратким руководством по созданию простой программы и компиляции в repl.it:

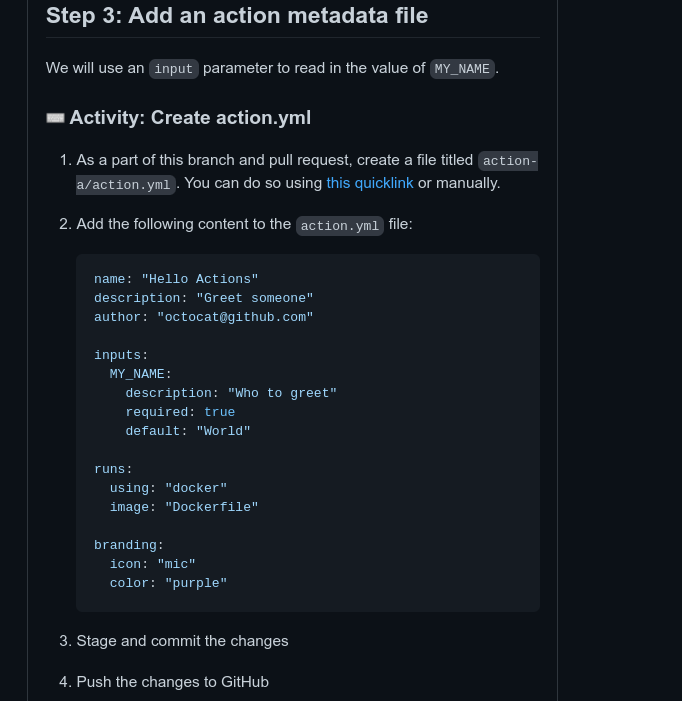
* https://www.youtube.com/watch?v=1CZBf1y4IvA;
* https://docs.repl.it/tutorials/01-introduction-to-the-repl-it-ide.

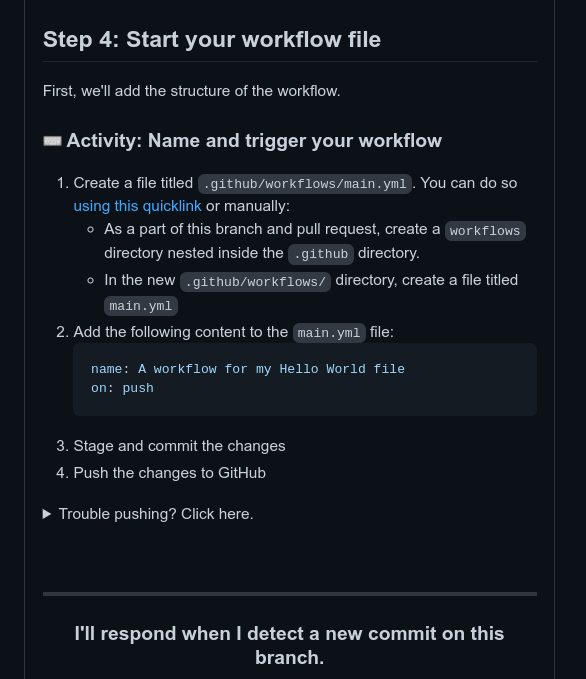
ЗАДАНИЕ 2. ИЗУЧАЕМ КУРС GitHub Actions: Hello World.

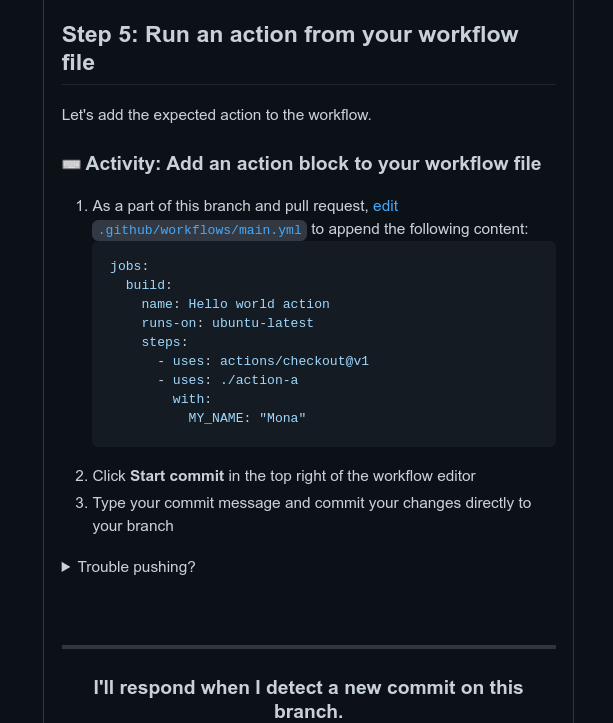
1. Познакомьтесь с описанием функционала в Сборка проекта с Github Action и в руководстве Как пользоваться Github Actions.
2. Откройте курс «GitHub Actions: Hello World» по ссылке https://lab.github.com/githubtraining/github-actions:-hello-world
3. Подпишитесь на курс и в процессе подписки создайте публичный репозиторий и 14 выберите режим работы в консоли, как и при изучении курса «Introduction to Github» из лабораторной работы 2.
4. Изучите курс, материалы которого помогут использовать Github Actions в заданиях 3 и 4.

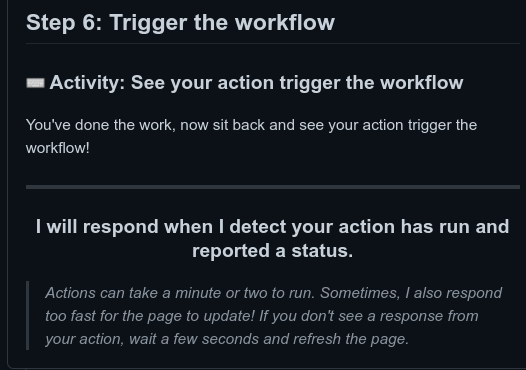


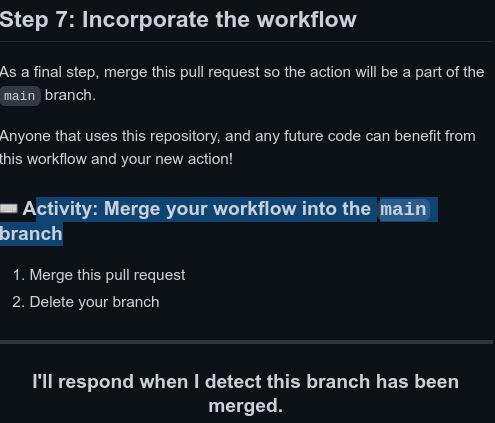










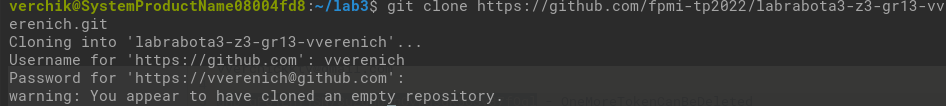




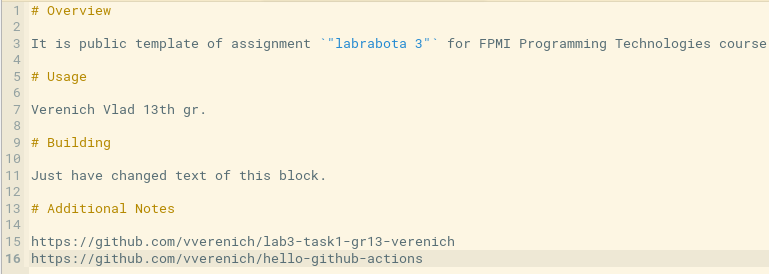
ЗАДАНИЕ 3. ИСПОЛЬЗУЕМ СТРУКТУРЫ ДЛЯ ПРЕДСТАВЛЕНИЯ ДАННЫХ

Repl.it не использовать для выполнения данного задания!

1. Cоздайте репозиторий, используя ссылку репозитория для задания 3 лабораторной работы 3 с сайта https://edufpmi.bsu.by для Вашей группы.

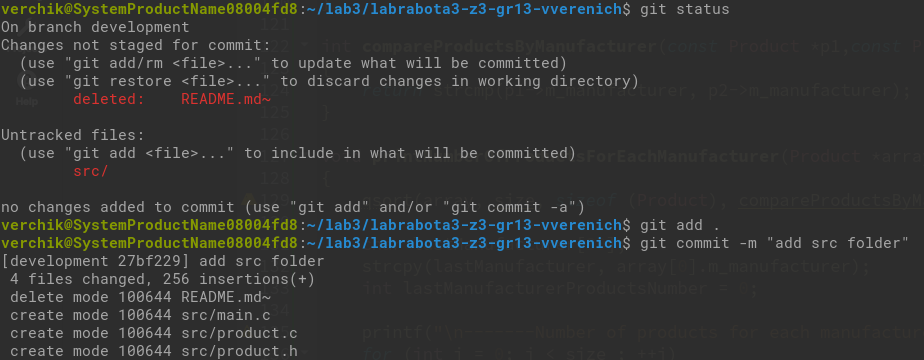


1. Файл README.md должен содержать разделы Overview, Usage, Building, Additional Notes. В Additional Notes добавить ссылки на репозитории из задания 1, 2.



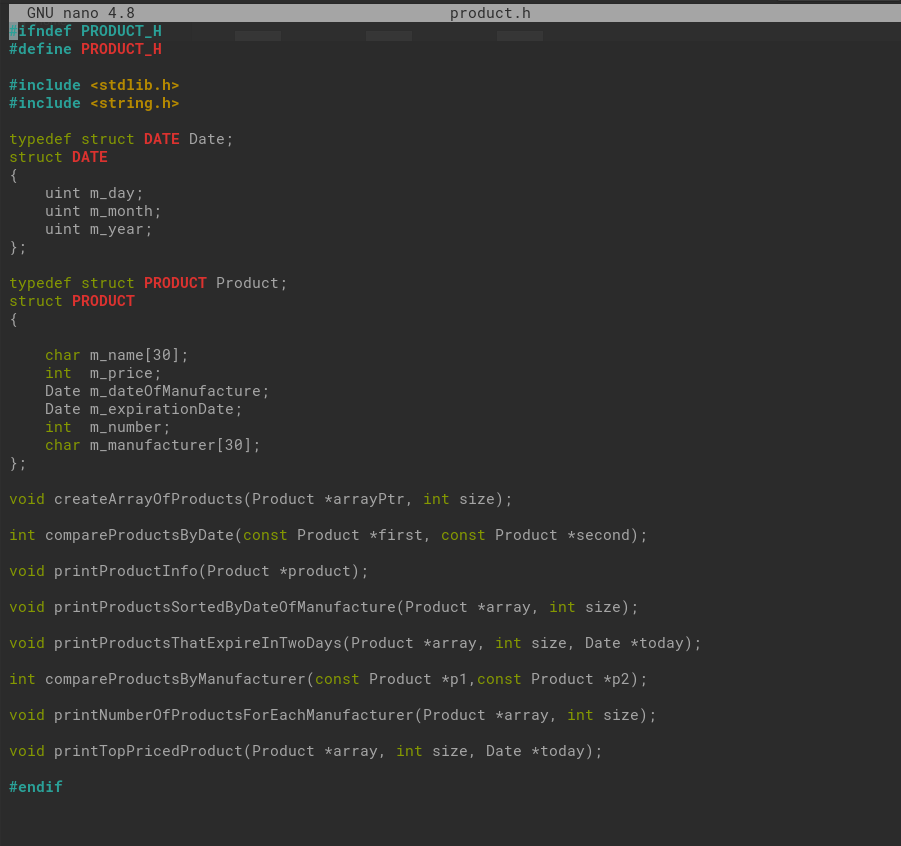
1. При работе над заданием следуйте правилам создания веток, изложенным в документации Требования к репозиториям.



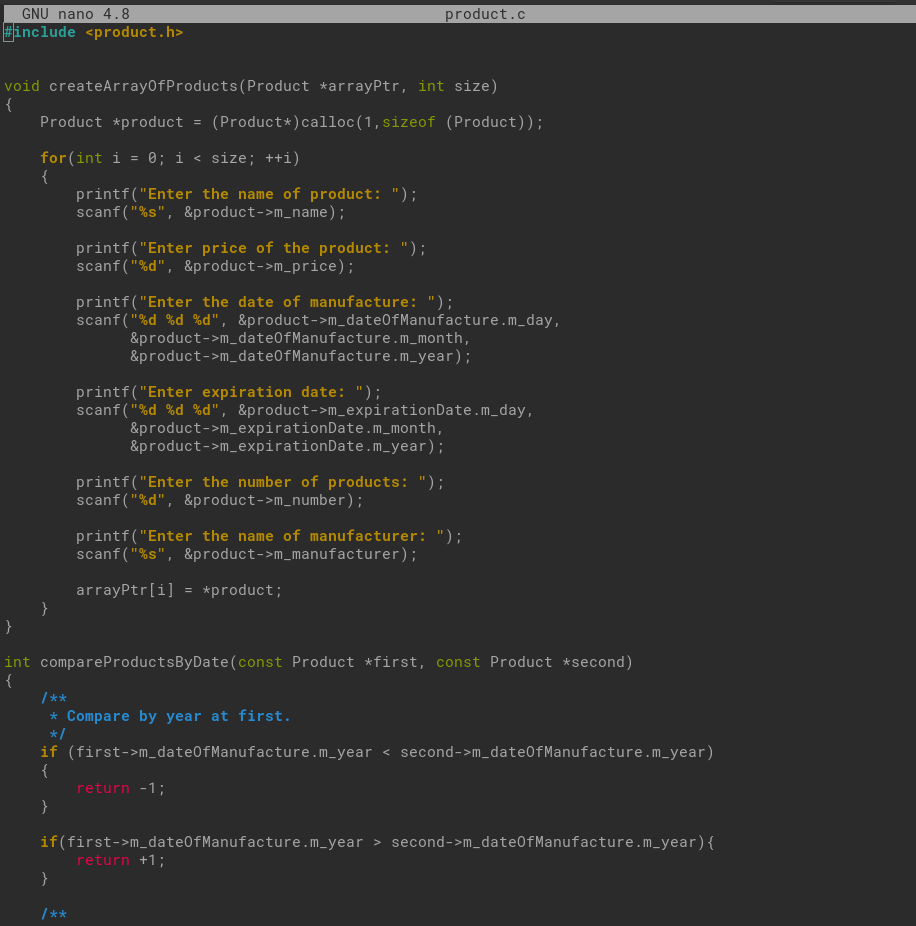


1. Изучите документацию Сборка приложений по сборке проектов с помощью утилиты make. Создайте файл Makefile.
2. Используя редактор nano, напишите программу с использованием структуры согласно варианту. Программа должна состоять из нескольких файлов с разделением полномочий на клиенты, серверы и интерфейсы согласно модели КИС — 2. Организация программы согласно модели КИС. Все файлы находятся в одном каталоге, например src.

**product.h:**

****

**product.c:**



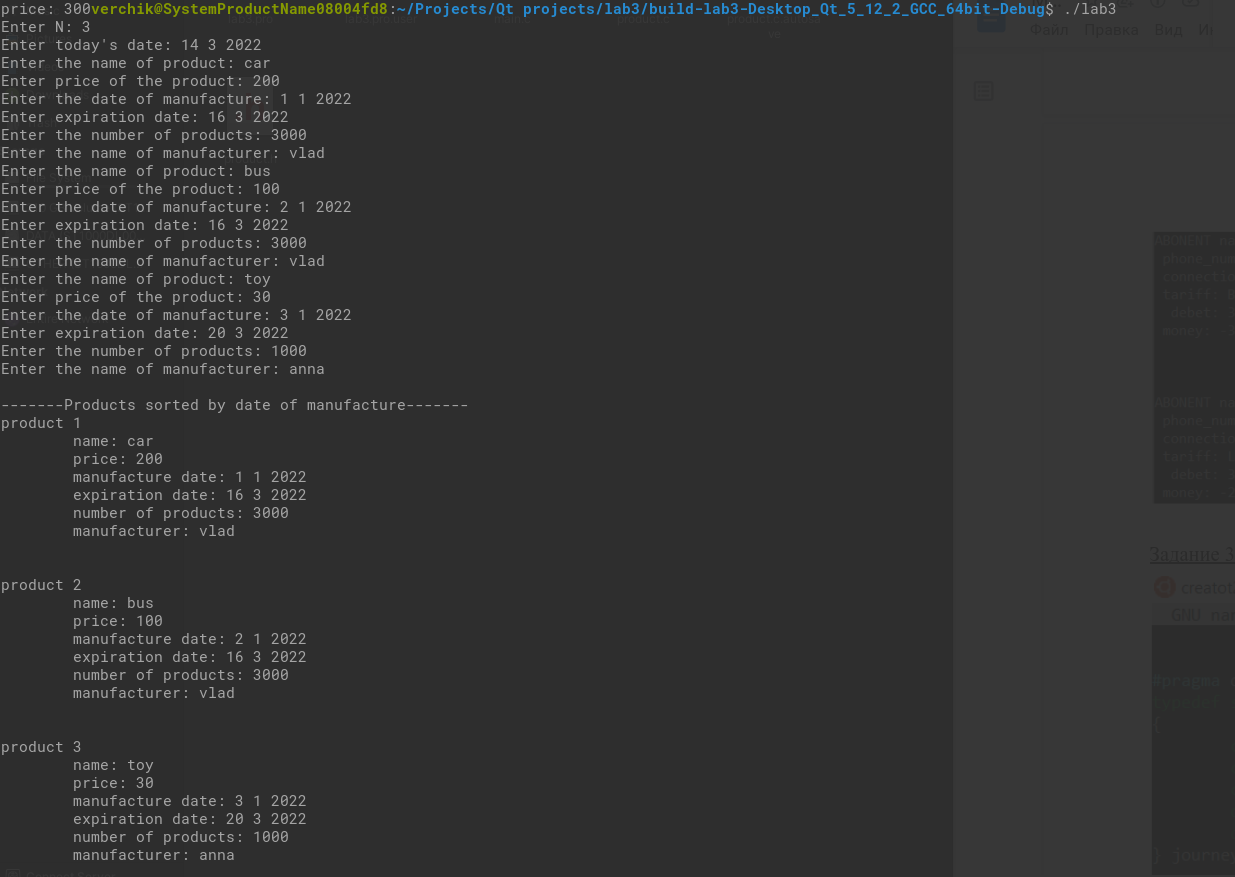


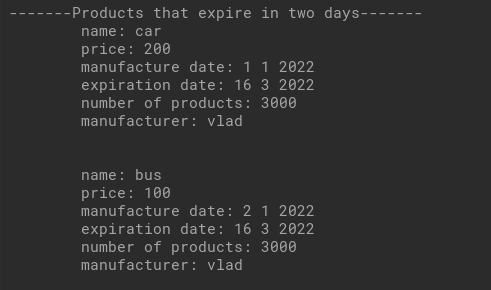




**main.c:**

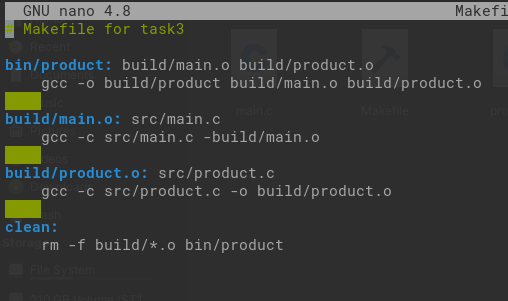








1. Требуется следовать стандарту оформления кода для языка С и файла сборки Makefile с помощью утилиты make.



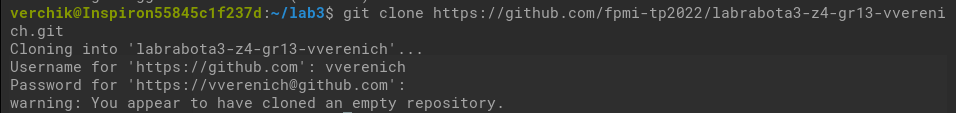
1. Выполнить сборку с помощью make и Github Actions согласно рекомендациям из Сборка проекта с Github Action и руководства Как пользоваться Github Actions. Одним из шагов для запуска рабочего процесса должен быть запуск утилиты make.
2. Провести тесты (контрольный расчет) для проверки работоспособности алгоритма и включить их в отчет.

| № п/п | Входные  данные | Ожидаемые выходные данные | Тест пройден |
| --- | --- | --- | --- |
| 1 | N = 3  today = 14 03 2022  name = car  price = 100  manufacture date = 01 02 2022  expiration date = 16 03 2022  number of products = 1000  manufacturer = vlad  name = bus  price = 200  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 3000  manufacturer = vlad  name = toy  price = 300  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 4000  manufacturer = anna | ------Products sorted by date of manufacture------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  product 2  name: bus  price: 200  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 3000  manufacturer: vlad  product 3  name: toy  price: 300  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 4000  manufacturer: anna  ------Products that expire in two days------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  ------Number of products for each manufacturer------  vlad: 4000  anna: 4000  ------Top priced product------  name: toy  price: 300 | да |
| 2 | N = 5  today = 14 03 2022  name = car  price = 100  manufacture date = 01 02 2022  expiration date = 16 03 2022  number of products = 1000  manufacturer = vlad  name = bus  price = 200  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 3000  manufacturer = vlad  name = toy  price = 300  manufacture date = 03 02 2022  expiration date = 17 03 2022  number of products = 4000  manufacturer = anna  name = computer  price = 200  manufacture date = 04 02 2022  expiration date = 17 03 2022  number of products = 10000  manufacturer = victor  name = telegraph  price = 250  manufacture date = 05 02 2022  expiration date = 17 03 2022  number of products = 8000  manufacturer = anna | ------Products sorted by date of manufacture------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  product 2  name: bus  price: 200  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 3000  manufacturer: vlad  product 3  name: toy  price: 300  manufacture date: 03 02 2022  expiration date: 17 03 2022  number of products: 4000  manufacturer: anna  product 4  name = computer  price = 200  manufacture date = 04 02 2022  expiration date = 17 03 2022  number of products = 10000  manufacturer = victor  product 5  name = telegraph  price = 250  manufacture date = 05 02 2022  expiration date = 17 03 2022  number of products = 8000  manufacturer = anna  ------Products that expire in two days------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  ------Number of products for each manufacturer------  vlad: 4000  anna: 12000  victor: 10000  ------Top priced product------  name: toy  price: 300 | да |
| 3 | N = 3  today = 14 03 2022  name = bus  price = 200  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 3000  manufacturer = vlad  name = toy  price = 300  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 4000  manufacturer = anna | ------Products sorted by date of manufacture------  product 1  name: bus  price: 200  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 3000  manufacturer: vlad  product 2  name: toy  price: 300  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 4000  manufacturer: anna  ------Products that expire in two days------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  ------Number of products for each manufacturer------  vlad: 3000  anna: 4000  ------Top priced product------  name: toy  price: 300 | да |
| 4 | N = 4  today = 14 03 2022  name = car  price = 100  manufacture date = 01 02 2022  expiration date = 16 03 2022  number of products = 1000  manufacturer = vlad  name = bus  price = 200  manufacture date = 02 02 2022  expiration date = 17 03 2022  number of products = 3000  manufacturer = vlad  name = toy  price = 300  manufacture date = 03 02 2022  expiration date = 17 03 2022  number of products = 4000  manufacturer = anna  name = computer  price = 200  manufacture date = 04 02 2022  expiration date = 17 03 2022  number of products = 10000  manufacturer = victor | ------Products sorted by date of manufacture------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  product 2  name: bus  price: 200  manufacture date: 02 02 2022  expiration date: 17 03 2022  number of products: 3000  manufacturer: vlad  product 3  name: toy  price: 300  manufacture date: 03 02 2022  expiration date: 17 03 2022  number of products: 4000  manufacturer: anna  product 4  name = computer  price = 200  manufacture date = 04 02 2022  expiration date = 17 03 2022  number of products = 10000  manufacturer = victor  ------Products that expire in two days------  product 1  name: car  price: 100  manufacture date: 01 02 2022  expiration date: 16 03 2022  number of products: 1000  manufacturer: vlad  ------Number of products for each manufacturer------  vlad: 4000  anna: 4000  victor: 10000  ------Top priced product------  name: toy  price: 300 | да |

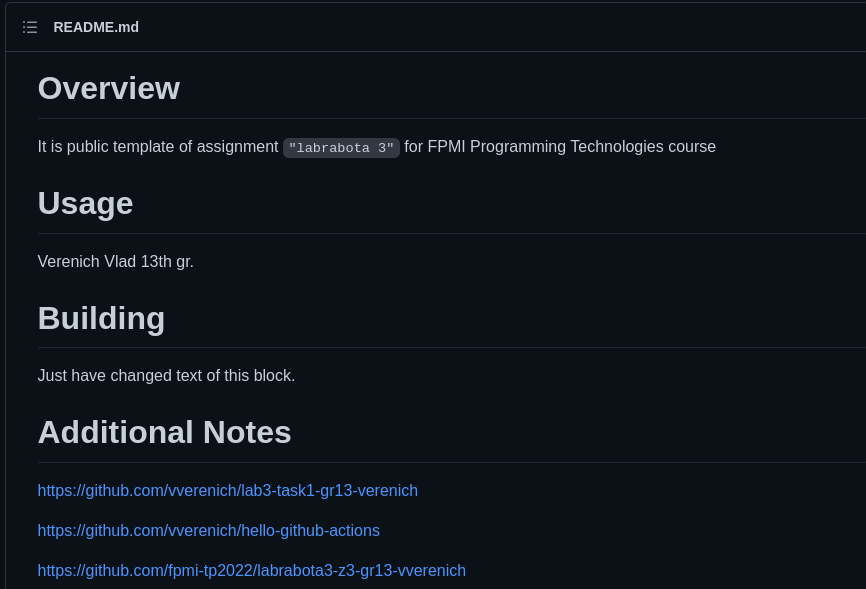
ЗАДАНИЕ 4. ЗАПИСЬ И ЧТЕНИЕ ТЕКСТОВЫХ ФАЙЛОВ

Repl.it не использовать для выполнения данного задания!

1. Cоздайте репозиторий, используя ссылку репозитория для задания 4 лабораторной работы 3 с сайта https://edufpmi.bsu.by для Вашей группы.

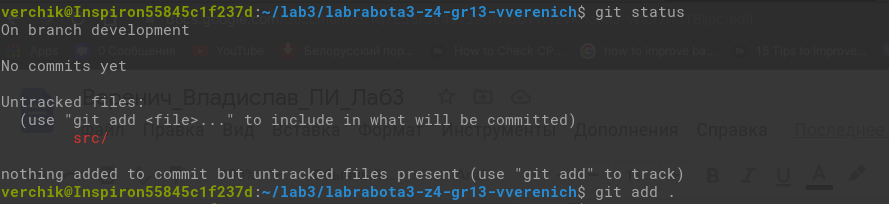


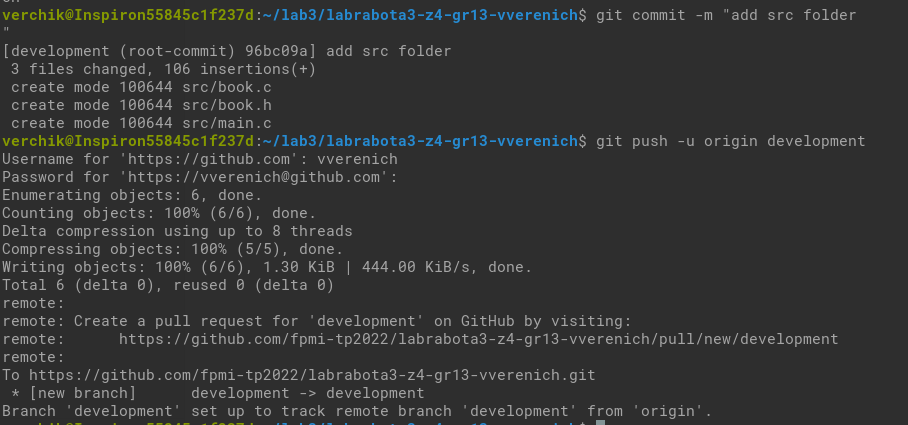
1. Файл README.md должен содержать разделы Overview, Usage, Building, Additional Notes. В Additional Notes добавить ссылки на репозитории из задания 1, 2, 3.



1. Внести в них изменения с учетом использования файловой структуры проекта согласно модели КИС.
2. При работе над заданием следуйте правилам создания веток, изложенным в документации Требования к репозиториям.



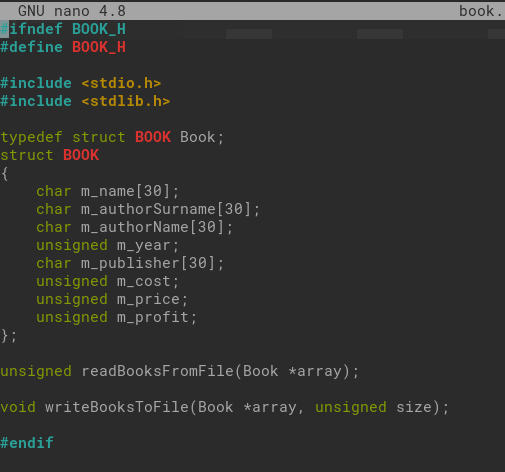




1. Создайте структуру проекта согласно Структура проекта согласно модели КИС и правил сборки.
2. Используя редактор nano, напишите программу для обработки текстовых файлов. В ходе выполнения требуется:

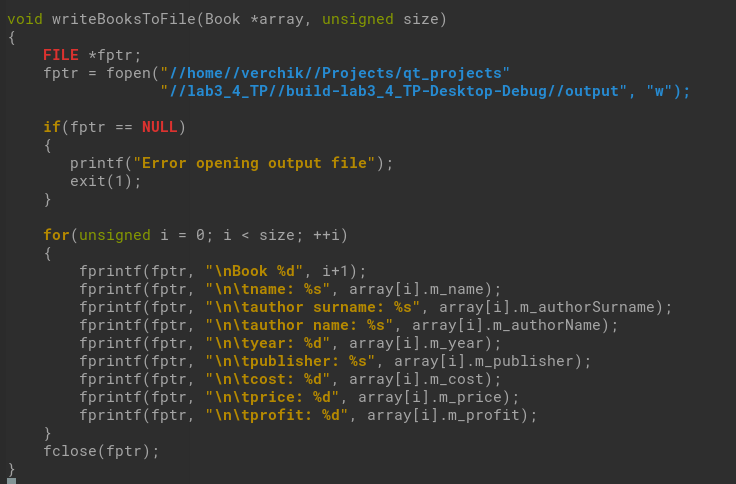
* Создать текстовый файл с произвольной информацией.
* Организовать просмотр содержимого файла.
* Организовать чтение и обработку данных из файла в соответствии с индивидуальным заданием. 15
* Сохранить полученные результаты в новый текстовый файл.

**Book.h**

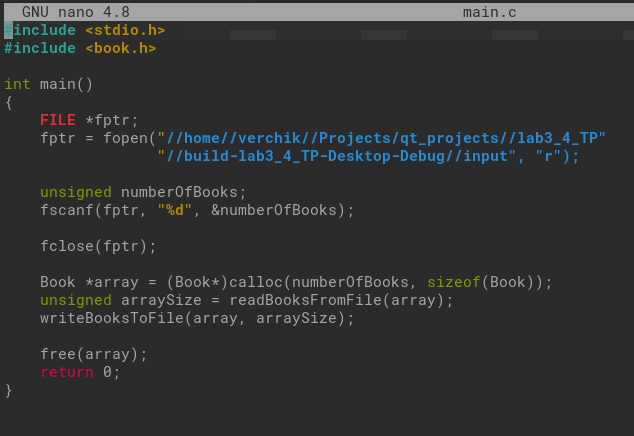


**Book.c**

****

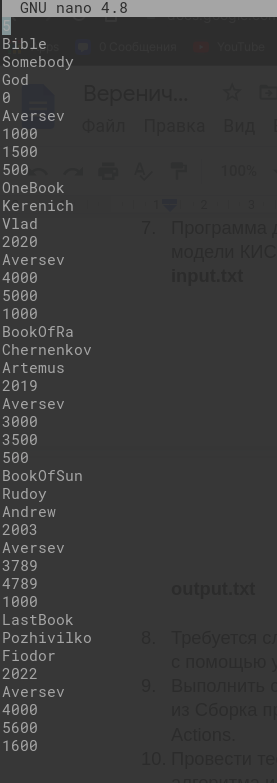
****

**main.c**

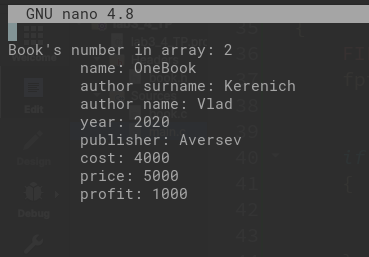
****

1. Программа должна состоять из нескольких файлов и иметь структуру согласно модели КИС.

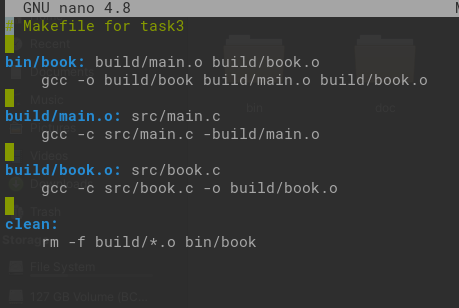
**input.txt**

****

**output.txt**

****

1. Требуется следовать стандарту оформления кода для языка С и файлов сборки с помощью утилиты make.



1. Выполнить сборку с помощью make и Github Actions согласно рекомендациям из Сборка проекта с Github Action и руководства Как пользоваться Github Actions.
2. Провести тесты (контрольный расчет) для проверки работоспособности алгоритма и включить их в отчет.

| № п/п | Входные данные | Ожидаемые результаты | Тест пройден |
| --- | --- | --- | --- |
| 1 | 5  Bible  Somebody  God  0  Aversev  1000  1500  500  OneBook  Kerenich  Vlad  2020  Aversev  4000  5000  1000  BookOfRa  Chernenkov  Artemus  2019  Aversev  3000  3500  500  BookOfSun  Rudoy  Andrew  2003  Aversev  3789  4789  1000  LastBook  Pozhivilko  Fiodor  2022  Aversev  4000  5600  1600 | Book's number in array: 2  name: OneBook  author surname: Kerenich  author name: Vlad  year: 2020  publisher: Aversev  cost: 4000  price: 5000  profit: 1000 | да |
| 2 | 3  Bible  Komebody  God  0  Aversev  1000  1500  500  OneBook  Kerenich  Vlad  2020  Aversev  4000  5000  1000  BookOfRa  Chernenkov  Artemus  2019  Aversev  3000  3500  500 | Book's number in array: 1  name: Bible  author surname: Komebody  author name: God  year: 0  publisher: Aversev  cost: 1000  price: 1500  profit: 500  Book's number in array: 2  name: OneBook  author surname: Kerenich  author name: Vlad  year: 2020  publisher: Aversev  cost: 4000  price: 5000  profit: 1000 | да |
| 3 | 5  Bible  Somebody  God  0  Aversev  1000  1500  500  OneBook  Verenich  Vlad  2020  Aversev  4000  5000  1000  BookOfRa  Chernenkov  Artemus  2019  Aversev  3000  3500  500  BookOfSun  Rudoy  Andrew  2003  Aversev  3789  4789  1000  LastBook  Pozhivilko  Fiodor  2022  Aversev  4000  5600  1600 |  | да |
| 4 | 4  Bible  Komebody  God  0  Aversev  1000  1500  500  OneBook  Kerenich  Vlad  2020  Aversev  4000  5000  1000  BookOfRa  Khernenkov  Artemus  2019  Aversev  3000  3500  500  BookOfSun  Kudoy  Andrew  2003  Aversev  3789  4789  1000 | Book's number in array: 1  name: Bible  author surname: Komebody  author name: God  year: 0  publisher: Aversev  cost: 1000  price: 1500  profit: 500  Book's number in array: 2  name: OneBook  author surname: Kerenich  author name: Vlad  year: 2020  publisher: Aversev  cost: 4000  price: 5000  profit: 1000  Book's number in array: 3  name: BookOfRa  author surname: Khernenkov  author name: Artemus  year: 2019  publisher: Aversev  cost: 3000  price: 3500  profit: 500  Book's number in array: 4  name: BookOfSun  author surname: Kudoy  author name: Andrew  year: 2003  publisher: Aversev  cost: 3789  price: 4789  profit: 1000 | да |

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Приведите примеры облачных IDE.

Koding Cloud 9 ShiftEdit Codeanywhere.

1. Опишите процесс подключения репозитория в repl.it

Достаточно авторизоваться при помощи GitHub, далее можно открыть существующий репозиторий и использовать виртуальный терминал.

1. Что такое «сборка»?

Получение исполняемых файлов из исходников кода. (компиляция + линковка).

1. Какие утилиты могут использоваться для сборки?

make, gcc.

1. Для чего служит Makefile?

Для описания сборки при помощи утилиты make.

1. Что такое цель в Makefile? Приведите пример.

то, что данное правило делает

Пример:**hello: ma**

1. Что такое связка в Makefile? Приведите пример.

определяют: 1) что нужно сделать (цель); 2) что для этого нужно

(зависимости); 3) как это сделать (правила).

Пример: **in.o hello.o**

1. Что такое зависимость в Makefile? Приведите пример.

Зависимости — это список файлов и целей, разделенных пробелом.

Пример: **gcc -o hello main.o hello.o**

1. Что такое правило в Makefile? Приведите пример.

Правила — это команды передаваемые оболочке.

Пример: hello - определение - далее идет связка, main.o: main.c - правило

1. Что такое макроопределение в Makefile? Приведите пример.

OBJ = main.o mod1.o mod2.o - макроопределение

1. Какая связка используется для очистки проекта?

rm

1. В чем заключается модель КИС для проектов на языке C?

Модель КИС (Клиент-Интерфейс-Сервер) — это элегантная концепция

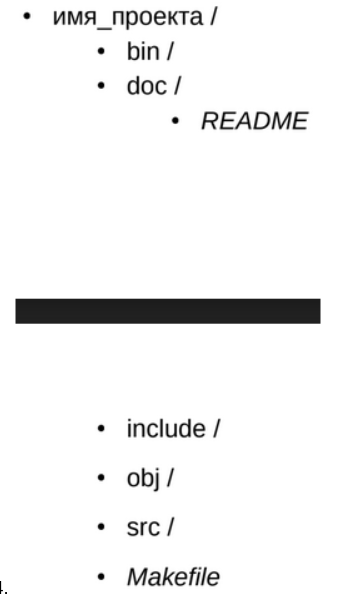
распределения исходного кода в репозитории, в рамках которой все исходники можно

поделить на клиенты, интерфейсы и серверы.

1. Что такое стандарты кодирования и почему рекомендуется им следовать?

Правила написания кода и организации программ. Для того, чтобы код можно было поддерживать.

1. Какой должна быть структура проекта согласно модели КИС и правилам сборки?



1. Что такое Github Actions и для каких задач применяется данный функционал?

GitHub Actions — бесплатная для публичных репозиториев система непрерывной интеграции. Github Actions позволяет выполнять ряд полезных задач:

• запустить проверку кода тестами;

• выполнить сборку и деплой проекта;

• опубликовать новую версию пакета;

• и ряд других задач.

1. Что такое workflow в GitHub Actions? Приведите пример.

Workflows (рабочий процесс) — каждый репозиторий на Github может содержать один или несколько workflow. Каждый workflow описывается в отдельном файле конфигурации в каталоге репозитория .github/workflows. Несколько workflow могут выполняться параллельно

1. Что такое event в GitHub Actions? Приведите пример.

Events (события) — рабочий процесс запускается одним или несколькими событиями. Это могут быть внутренние события (push или pull request), запланированные события (по расписанию планировщика, например cron) или произвольными внешними событиями, например Webhook API GitHub.

1. Что такое job в GitHub Actions? Приведите пример.

Jobs (задания) — рабочий процесс состоит из одного или нескольких заданий. Задание содержит набор команд, которые запускаются вместе с рабочим процессом. По умолчанию при запуске рабочего процесса все его задания выполняются параллельно, но если между ними определена зависимость, то — последовательно.

1. Что такое step в GitHub Actions? Приведите пример.

Steps (шаг) — задания jobs состоят из последовательности шагов. Шаг — это либо команда оболочки (shell command), либо действие (action). Все шаги задания выполняются последовательно в исполнителе runner, связанном с заданием. В случае сбоя шага все следующие шаги задания пропускаются.

1. Что такое action в GitHub Actions? Приведите пример.

Actions (действие) — это многократно используемый блок код, который может служить шагом задания. Каждое действие action может принимать на вход параметры и создавать любые значения, которые затем можно использовать в других действиях actions.