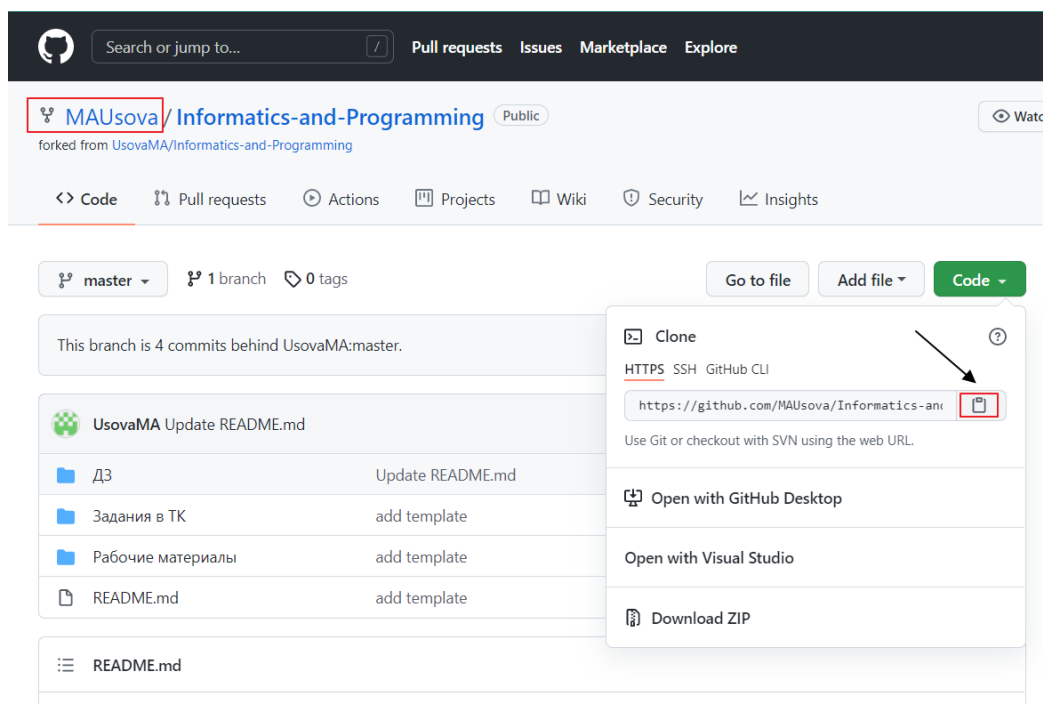


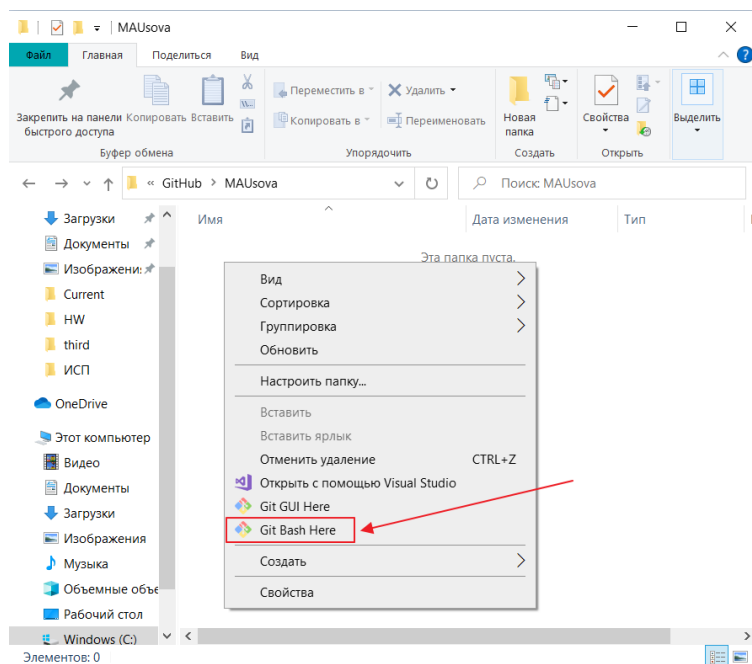
# Инструкция по работе с Github

Если какие-то шаги уже были выполнены – пропусти их.

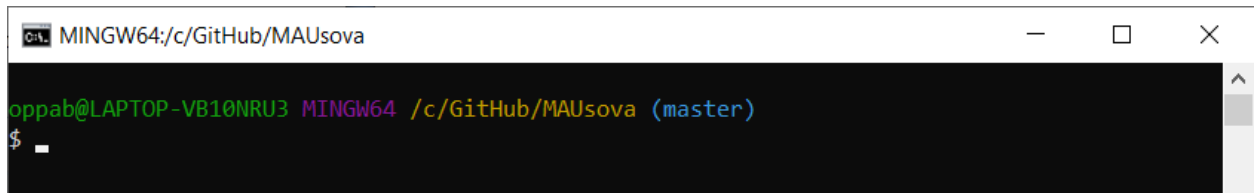
1. Чтобы получить себе копию репозитория курса, нужно сделать fork исходного репозитория. Для этого переходим на [UsovaMA/Informatics-and-Programming \(github.com\)](https://github.com/UsovaMA/Informatics-and-Programming) и жмём кнопку Fork.
2. Открываем на гитхабе ваш форкнутый репозиторий. Убеждаемся, что это не исходный репозиторий (UsovaMA/Informatics-and-Programming), а именно ваш. Копируем ссылку на него во вкладке Code.



3. Создаём ваше рабочее пространство (папку), в которой будет локальная копия вашего рабочего репозитория. Жмём ПКМ, выбираем git bash here...



Появляется консоль, в которой мы будем выполнять git команды.

A screenshot of a terminal window titled 'MINGW64:/c/GitHub/MAUsova'. The prompt shows the user 'oppab@LAPTOP-VB10NRU3' in the 'MINGW64' shell, currently on the 'master' branch of the '/c/GitHub/MAUsova' repository. The prompt is '\$' followed by a cursor.

Для начала работы с консолью, необходимо указать имя пользователя через следующую Git команду:

```
git config --global user.name "Name LastName"
```

Чтобы указать Email пользователя, нужно ввести Git команду:

```
git config --global user.email user@mail.ru
```

Проверить текущие настройки можно Git командой:

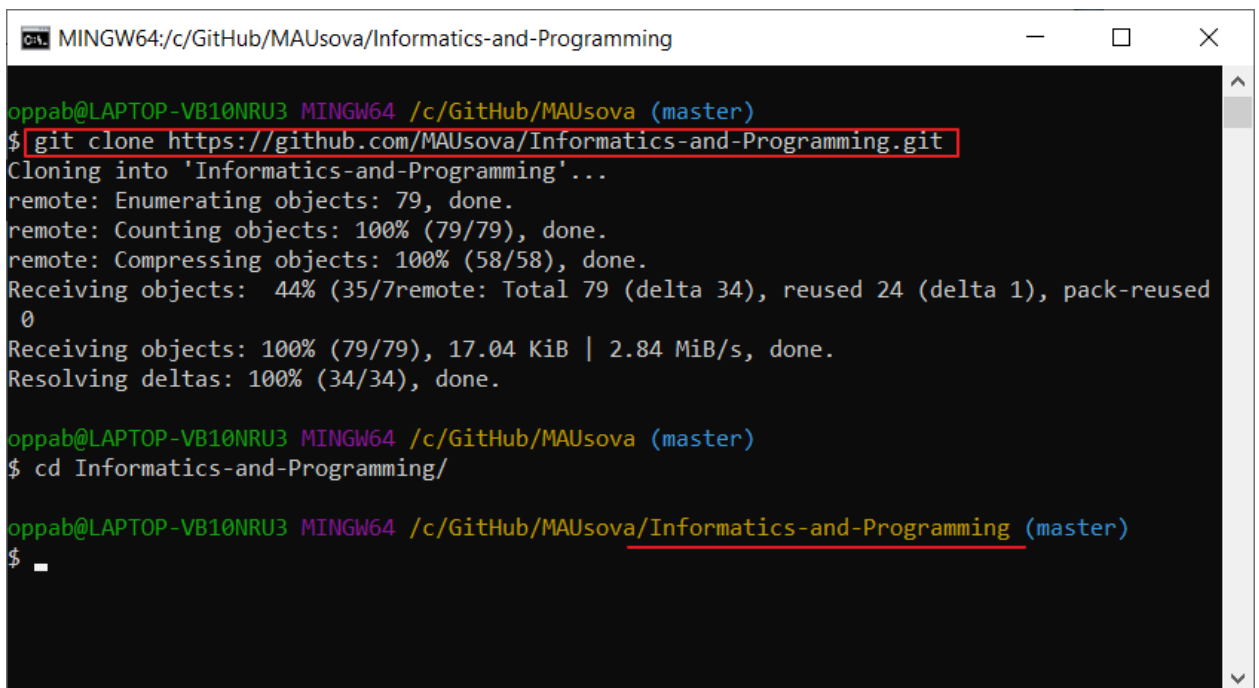
```
git config -list
```

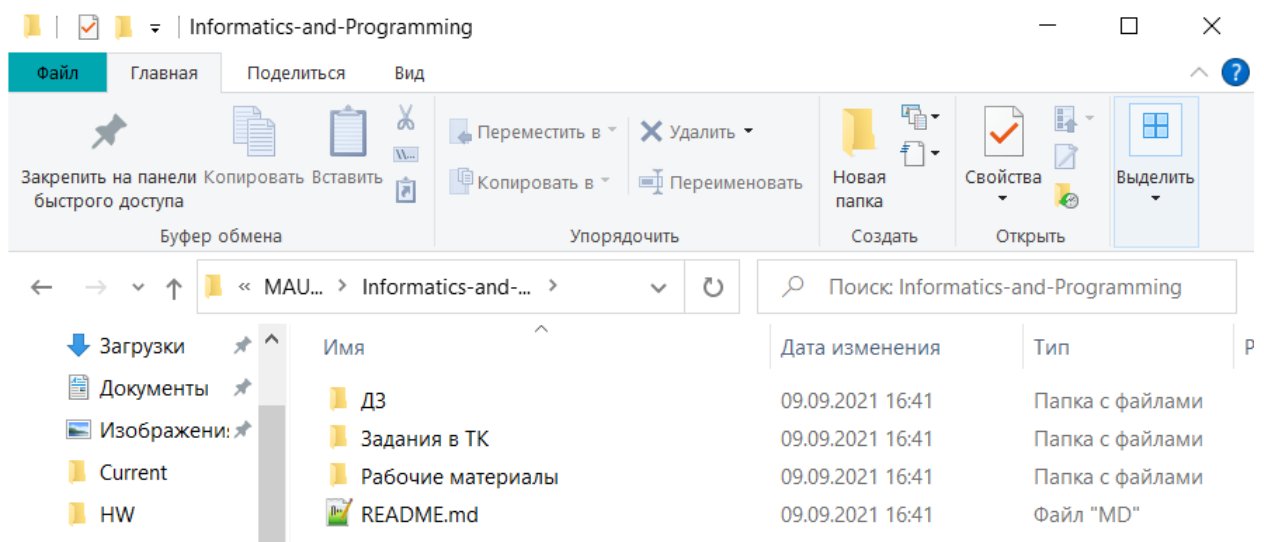
4. Клонировать себе содержимое вашего репозитория на вашу локальную машину с помощью команды

```
git clone <ссылка на репозиторий>
```

И переходим в появившуюся папку с помощью команды

```
cd Informatics-and-Programming/
```

A screenshot of a terminal window titled 'MINGW64:/c/GitHub/MAUsova/Informatics-and-Programming'. The prompt shows the user 'oppab@LAPTOP-VB10NRU3' in the 'MINGW64' shell, currently on the 'master' branch of the '/c/GitHub/MAUsova' repository. The user enters the command 'git clone https://github.com/MAUsova/Informatics-and-Programming.git', which is highlighted with a red box. The output shows the cloning process: 'Cloning into 'Informatics-and-Programming'...', 'remote: Enumerating objects: 79, done.', 'remote: Counting objects: 100% (79/79), done.', 'remote: Compressing objects: 100% (58/58), done.', 'Receiving objects: 44% (35/79) remote: Total 79 (delta 34), reused 24 (delta 1), pack-reused 0', 'Receiving objects: 100% (79/79), 17.04 KiB | 2.84 MiB/s, done.', 'Resolving deltas: 100% (34/34), done.'. The user then enters 'cd Informatics-and-Programming/' and the prompt changes to 'oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)'. The prompt is '\$' followed by a cursor.



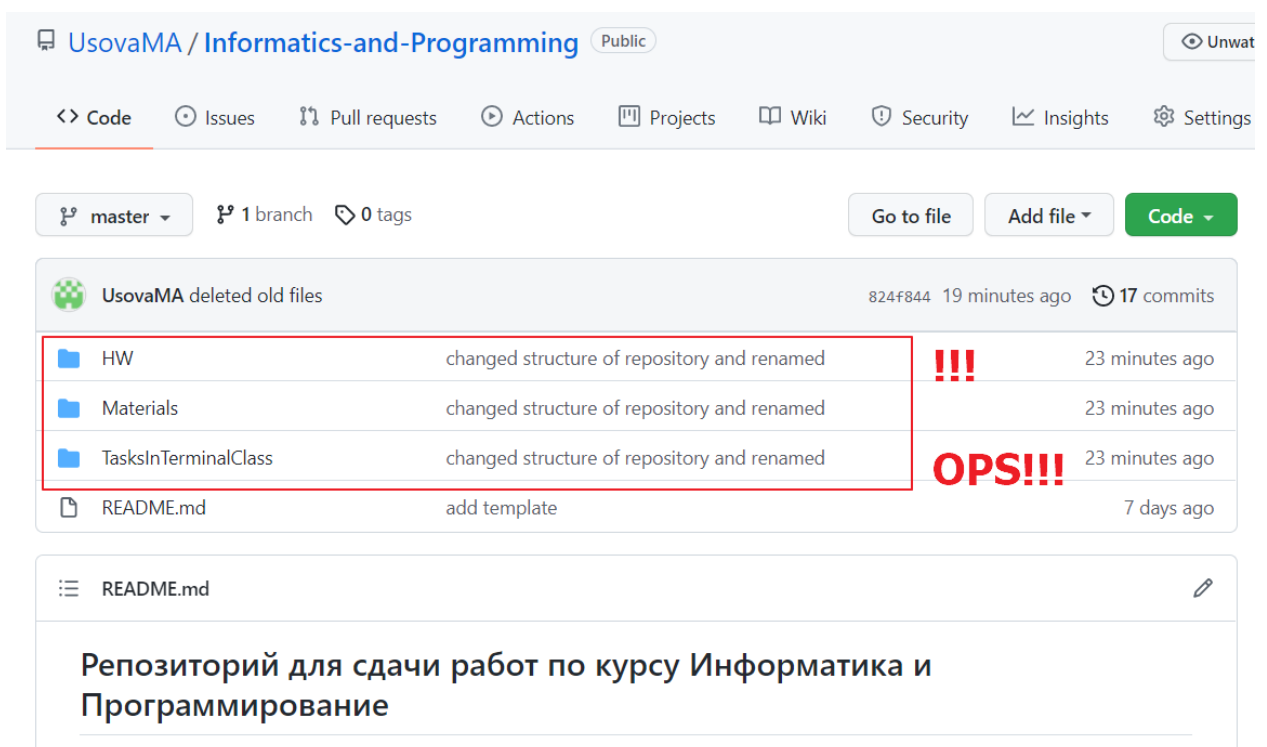
На данном этапе наш репозиторий имеет локальную копию и удаленную на гитхабе. Репозиторий на гитхабе будет называться origin. Проверить это можно, написав в консоли:

```
git remote -v
```

```

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git remote -v
origin https://github.com/MAUsova/Informatics-and-Programming.git (fetch)
origin https://github.com/MAUsova/Informatics-and-Programming.git (push)
  
```

Однако если мы посмотрим на главный репозиторий, то там произошли какие-то изменения.



**Наш репозиторий перестал соответствовать исходному.** Нужно обновиться до текущего состояния исходного репозитория.

5. Для начала необходимо синхронизировать ваш репозиторий и репозиторий курса, для этого необходимо подключить репозиторий курса к своему:

```
git remote add upstream https://github.com/UsovaMA/Informatics-and-Programming.git
```

Затем необходимо получить все изменения с этого репозитория командой

```
git fetch upstream
```

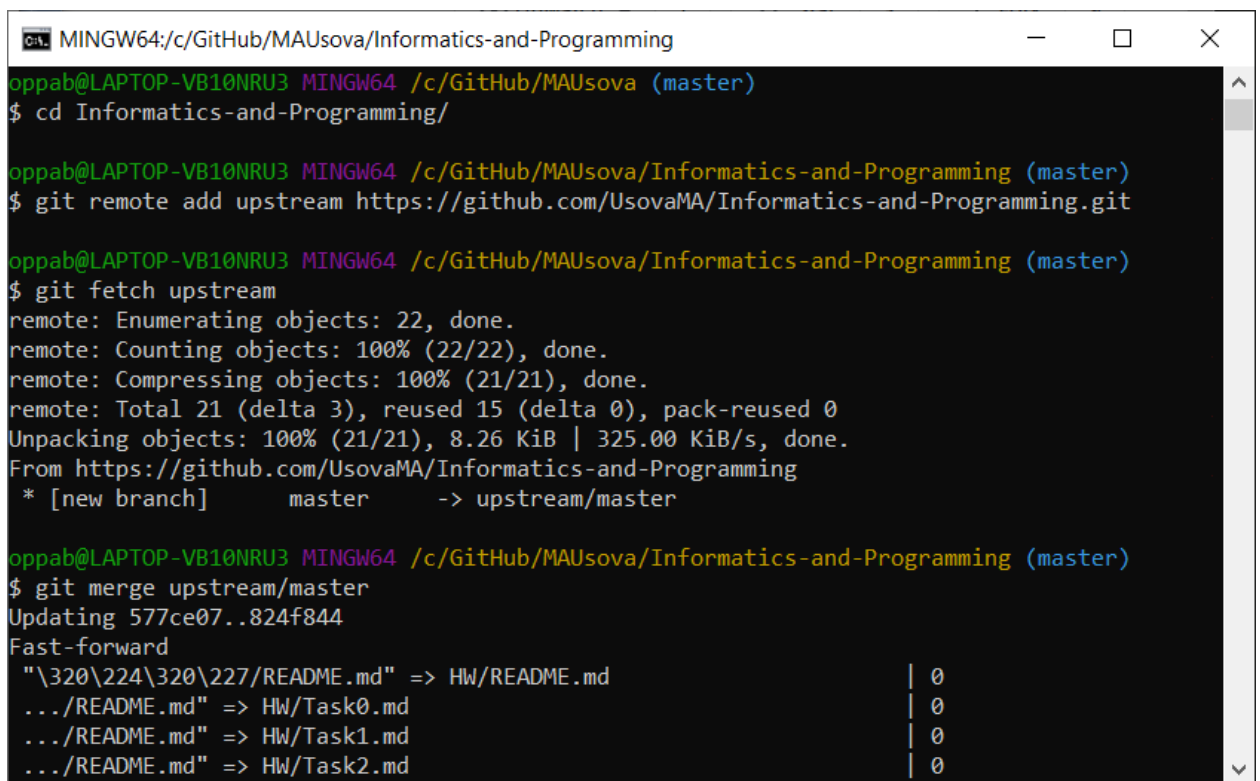
Переключаемся на свою ветку master (если мы вдруг не в ней)

```
git checkout master
```

И поместить наши текущие изменения поверх полученных с оригинального репозитория

```
git merge upstream/master
```

Если при слиянии не произойдет конфликтов, то все данные из репозитория курса синхронизируются в ваш репозиторий при этом не затронув ваши данные.



```
MINGW64:/c/GitHub/MAUsova/Informatics-and-Programming
oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova (master)
$ cd Informatics-and-Programming/

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git remote add upstream https://github.com/UsovaMA/Informatics-and-Programming.git

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git fetch upstream
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 21 (delta 3), reused 15 (delta 0), pack-reused 0
Unpacking objects: 100% (21/21), 8.26 KiB | 325.00 KiB/s, done.
From https://github.com/UsovaMA/Informatics-and-Programming
* [new branch]      master      -> upstream/master

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git merge upstream/master
Updating 577ce07..824f844
Fast-forward
 "\320\224\320\227\README.md" => HW/README.md      | 0
 .../README.md" => HW/Task0.md                      | 0
 .../README.md" => HW/Task1.md                      | 0
 .../README.md" => HW/Task2.md                      | 0
```

6. Отправим все изменения в свой репозиторий на github:

```
git push
```

Структура проекта была переделана, чтобы не плодить папки с проектами. В папках HW и TasksInTerminalClass будет по одному решению, в котором будет множество проектов.

7. При работе с какой-то задачей рекомендуется уходить для разработки в другую ветку. Для этого можно использовать следующие команды:

```
git branch — посмотреть имеющиеся ветки
```

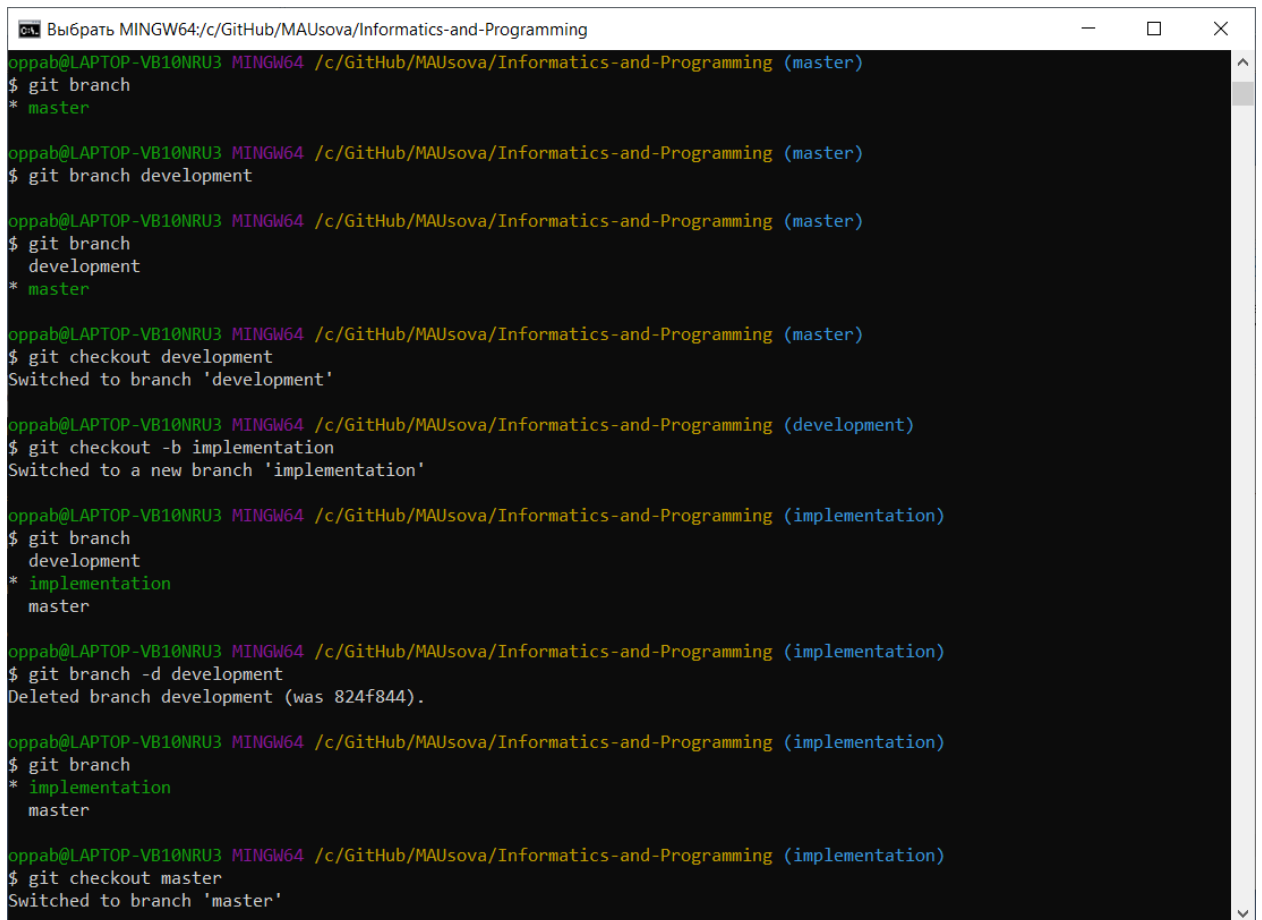
`git branch <name_of_branch>` - создать ветку с указанным именем

`git checkout <name_of_branch>` - перейти в ветку с указанным именем

`git checkout -b <name_of_branch>` - создать ветку и сразу перейти в неё

`git branch -d <name_of_branch>` - удалить указанную ветку

**Замечание.** Нельзя удалить ветку, в которой вы находитесь, нужно сначала перейти в другую ветку.



```
Выбрать MINGW64/c/GitHub/MAUsova/Informatics-and-Programming
oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git branch
* master

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git branch development

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git branch
development
* master

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (master)
$ git checkout development
Switched to branch 'development'

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (development)
$ git checkout -b implementation
Switched to a new branch 'implementation'

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (implementation)
$ git branch
development
* implementation
master

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (implementation)
$ git branch -d development
Deleted branch development (was 824f844).

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (implementation)
$ git branch
* implementation
master

oppab@LAPTOP-VB10NRU3 MINGW64 /c/GitHub/MAUsova/Informatics-and-Programming (implementation)
$ git checkout master
Switched to branch 'master'
```

Прежде чем писать код, создайте и уйдите в рабочую ветку. Далее везде будет подразумеваться, что мы работаем в ветке `implementation` (название может быть любым).

Переидём к созданию проекта в локальном репозитории.

8. Будем создавать решение для задач, выполняемых в классе. Создаём проект в MVS.

Выбираем команду создать ПУСТОЙ ПРОЕКТ и настраиваем проект.

1. Указываем название. Допустим, мы собираемся решать задачу по поиску площадей геометрических фигур. Задаём название ПРОЕКТА (одна отдельная задача) – `CalculationsOfAreas`.
2. Затем настраиваем путь до папки, в которой у нас будут выполняться задачи терминал-класса.

- Затем меняем название РЕШЕНИЯ на подходящее. В данном решении мы будем решать только задачи в терминал-классе, поэтому даём название `SolutionTasksInTerminal`.

После этого жмём создать.

Настроить новый проект

Пустой проект C++ Windows Консоль

Имя проекта  
CalculationOfAreas

Расположение  
C:\GitHub\MAUsova\Informatics-and-Programming\TasksInTerminalClass\

Имя решения ⓘ  
SolutionTasksInTerminal

☐ Поместить решение и проект в одном каталоге

Назад Создать

Проект создан и он должен быть пустым! Никаких файлов не должно быть.

- Жмём правой кнопкой мыши по ПРОЕКТУ и выбираем добавить – создать элемент. Создаём файл C++, но меняем расширение на `.c`.

Добавление нового элемента - CalculationOfAreas

Установленные

- Visual C++
  - Код
  - Форматирование
  - Библиотека ATL
  - Данные
  - Ресурсы
  - Веб
  - Служебные программы
  - Таблицы свойств
  - Тест
  - HLSL
  - Графика
- В сети

Сортировка: По умолчанию

Поиск (Ctrl+E)

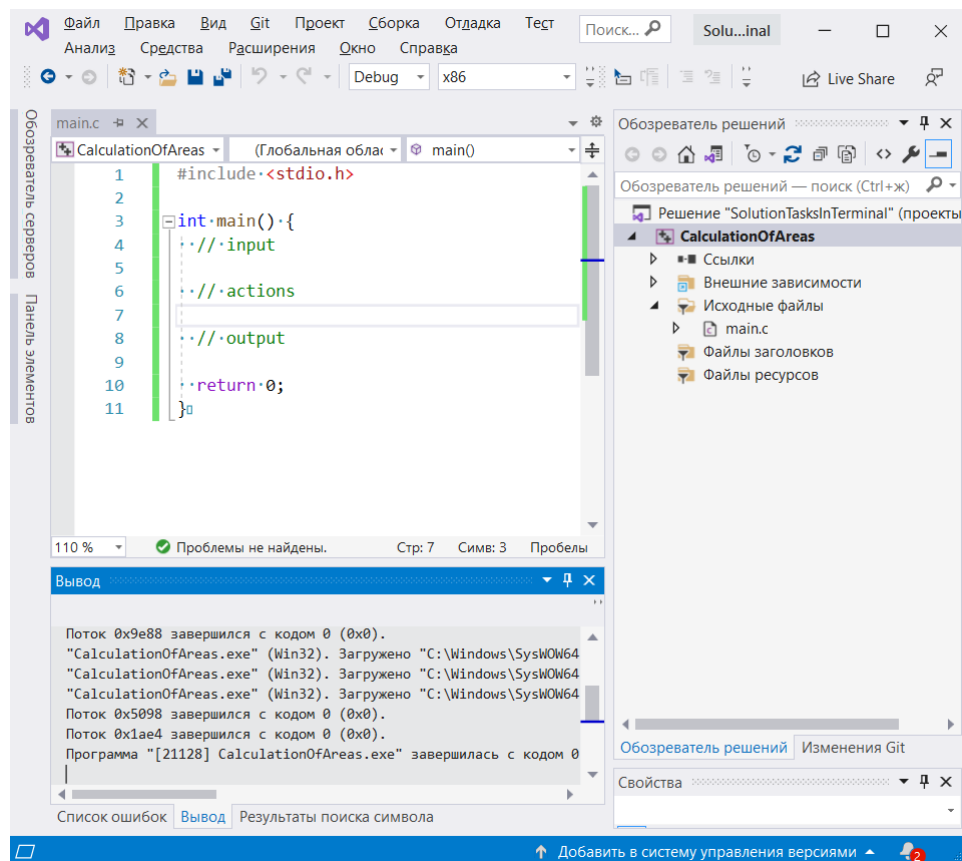
Файл C++ (.cpp)	Visual C++	<b>Тип:</b> Visual C++ Создает файл, содержащий исходный код C++.
Файл заголовка (.h)	Visual C++	
Класс C++	Visual C++	
Блок интерфейса модуля C...	Visual C++	

Имя: main.c

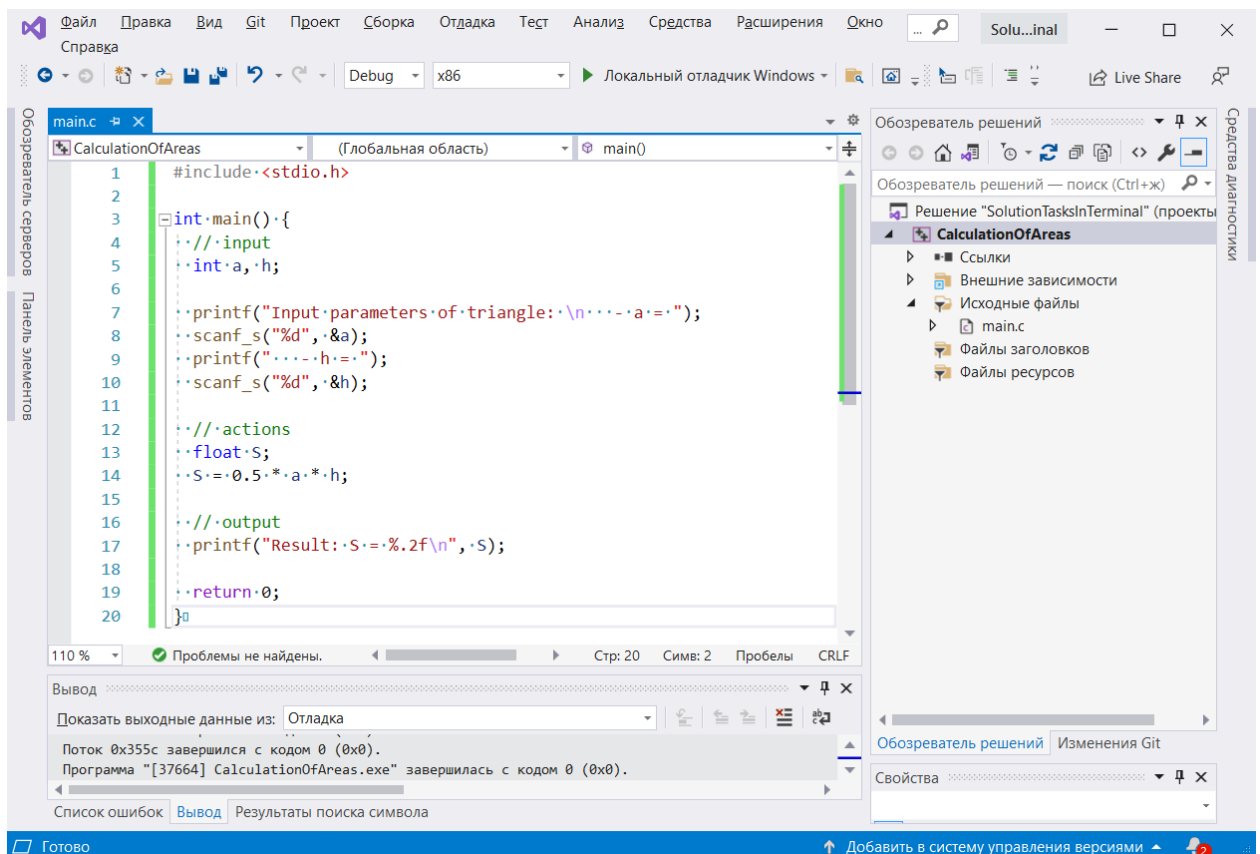
Расположение: C:\GitHub\MAUsova\Informatics-and-Programming\TasksInTerminalC


Обзор... Добавить Отмена

10. Пишем заготовку под вашу задачу и запускаем, чтобы проверить работоспособность данного проекта.



11. Добавляем какую-то реализацию задачи, проверяем работоспособность.





Консоль отладки Microsoft Visual Studio

```

Input parameters of triangle:
- a = 3
- h = 7
Result: S = 10.50

C:\GitHub\MAUsova\Informatics-and-Programming\TasksInTerminalClass\SolutionTasksInTerminal\
Debug\CalculationOfAreas.exe (процесс 37664) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->
"Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...

```

12. Допустим, нас всё устраивает и мы хотим выложить данную реализацию в нашем удалённом репозитории на гитхабе.

Смотрим какие изменения были внесены с помощью команды

```
git status
```

Там будет отображаться ваш добавленный проект. Можно добавить все изменения к комиту (который мы будем отправлять на удалённый репозиторий) с помощью команды

```
git add .
```

ОДНАКО! Возьмём за привычку не заливать подряд все файлы. Папки **Debug** выкладывать не нужно. Решения:

1. Удалить их (удалять не обязательно через консоль, обе папки можно удалить привычным способом удаления файлов – это файлы сгенерируются снова при запуске вами программы, поэтому их выкладывать не стоит),
2. Выкладывать файлы выборочно.

```

MINGW64/c:/Git/Hub/MAUsova/Informatics-and-Programming
oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git add TasksInTerminalClass/SolutionTasksInTerminal/SolutionTasksInTerminal.sln

oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git add TasksInTerminalClass/SolutionTasksInTerminal/CalculationOfAreas/main.c

oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git add TasksInTerminalClass/SolutionTasksInTerminal/CalculationOfAreas/CalculationOfAreas.vcxproj

oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git add TasksInTerminalClass/SolutionTasksInTerminal/CalculationOfAreas/CalculationOfAreas.vcxproj.filters

oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git add TasksInTerminalClass/SolutionTasksInTerminal/CalculationOfAreas/CalculationOfAreas.vcxproj.user

oppab@LAPTOP-VB10NRU3 MINGW64 /c:/Git/Hub/MAUsova/Informatics-and-Programming (implementation)
$ git status

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   TasksInTerminalClass/SolutionTasksInTerminal/SolutionTasksInTerminal.sln
        new file:   "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/main.c"
        new file:   "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/\320\241alculationOfAreas.vcxproj"
        new file:   "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/\320\241alculationOfAreas.vcxproj.filters"
        new file:   "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/\320\241alculationOfAreas.vcxproj.user"

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        TasksInTerminalClass/SolutionTasksInTerminal/.vs/
        TasksInTerminalClass/SolutionTasksInTerminal/Debug/
        "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/Debug/"
        TasksInTerminalClass/SolutionTasksInTerminal/Debug/
        "TasksInTerminalClass/SolutionTasksInTerminal/\320\241alculationOfAreas/Debug/"

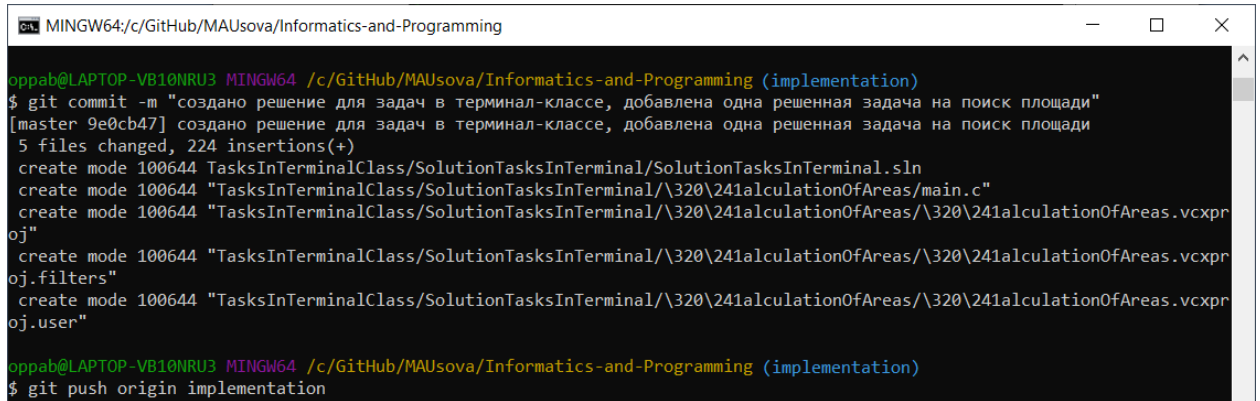
```



После вызова статуса видим, что недобавленными остались только папки Debug.

Затем добавляем эти файлы к коммиту, даём комментарий по поводу того, что это за изменения, можно даже на русском языке

```
git commit -m "коммент"
```

A screenshot of a Windows terminal window with a dark background. The title bar shows the path 'MINGW64:/c:/GitHub/MAUsova/Informatics-and-Programming'. The terminal text shows a user named 'oppab' at a prompt. They run 'git commit -m "создано решение для задач в терминал-классе, добавлена одна решенная задача на поиск площади"'. The output shows the commit hash '9e0cb47', the commit message, and a list of 5 new files created. Then they run 'git push origin implementation'.

Затем заливаем изменения в ваш удалённый репозиторий на гитхабе. Можно явно указать ветку, в которую заливаются

```
git push origin implementation
```

По умолчанию, данные заливаются в ветку, в которой вы находитесь.