AGH University of Krakow | Faculty of Mechanical Engineering and Robotics
| Department of Robotics and Mechatronics

MECHATRONIC ENGINEERING | I cycle | semester VI | 2025/2026

Object oriented programming and software engineering

Instruction:

Git

You will learn:

How to use basic version control with Git, including creating repositories, tracking changes, creating a commit and working with branches.

You will be able to use Git to manage your code and collaborate with others. You will learn how to create and manage repositories, track and commit changes, work with branches, merge and resolve conflicts, use remote repositories, understand the history of your project.

By the end of this laboratory, you should be able to use Git confidently working in teams.

Course supervisor:
dr inż. Lucjan Miękina, miekina@agh.edu.pl

Instruction author:
mgr inż. Joanna Koszyk, jkoszyk@agh.edu.pl

1. Initial information

Git is a distributed version control system, which means that every repository contains the same information. In most cases, there is a central server used to store a shared repository, but each cloned repository is a full copy of that repository. Version control systems are not limited to working with text files, they can handle files of any type, including binary files.

Git allows multiple people to work on the same project without overwriting each other's work. Beside files it stores history of changes and versions of the project.

2. Theoretical Content

Git:

git init – creates a new, empty repository in the current directory
git clone – clones a remote repository into a local directory
git add – adds a file to the index
git add . – adds all modified files
git commit – creates a commit
git commit -m "message" – creates a commit with a message
git commit -am "message" – adds changes in tracked files and creates a commit
git status – shows the current status of files
git remote add origin [address] – sets the remote repository as origin
git fetch – downloads the latest changes from the remote repository but does not merge them
git pull – downloads and merges changes from the remote repository
git push – sends commits to the remote repository
git branch – shows the list of branches
git branch [new_branch_name] – creates a new branch
git checkout – switches to another branch
git diff – shows changes between commits, the commit and the working directory, etc.
git merge – merges a branch into the current branch

The typical workflow in Git:

```
git add -> git commit -> git push
```

Checking status:

```
git status
```

Adding files:

```
git add file.cpp
git add .
```

Committing changes:

```
git commit -m "message"
```

Viewing history:

```
git log
```

Branches:

```
git branch feature
git checkout feature
```

or:

```
git checkout -b feature
```

Merging:

```
git checkout main
git merge feature
```

If two changes affect the same part of a file, conflicts might occur. Git marks conflicts in files and it should be resolved manually.

Remote repository:

```
git remote add origin <repository_url>
git push -u origin main
```

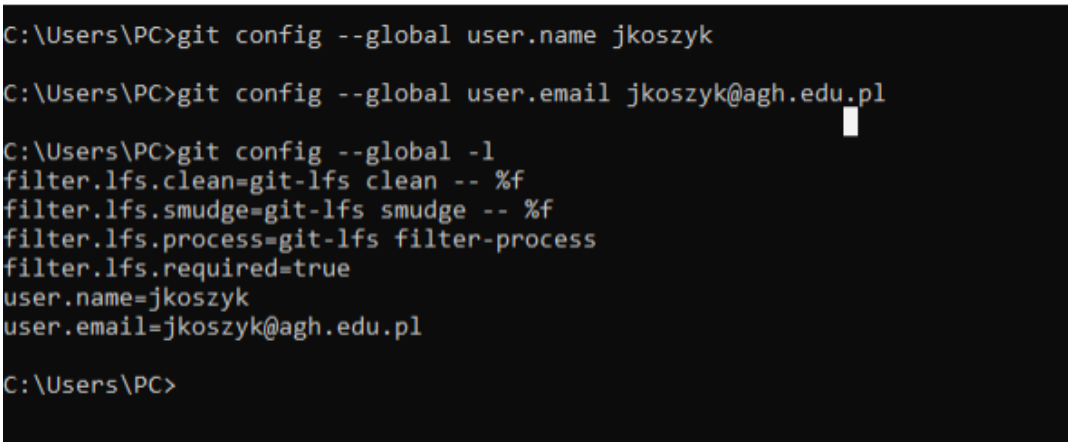
Pulling changes:

```
git pull
```

3. Tasks

TASK 1.

Create an account at: https://gitlab.com/ In the next step, you need to log in (identify the person making the changes). To do this, you should provide your details (username and email address) in the config. For the purposes of the course, you must use an email address in the student domain.



Create a repository:

```
C:\Users\PC\Downloads>mkdir poio-proj2

C:\Users\PC\Downloads>cd poio-proj2

C:\Users\PC\Downloads\poio-proj2>git init
Initialized empty Git repository in C:/Users/PC/Downloads/poio-proj2/.git/

C:\Users\PC\Downloads\poio-proj2>echo "# project1" >> README.md

C:\Users\PC\Downloads\poio-proj2>git add README.md

C:\Users\PC\Downloads\poio-proj2>git commit -m "first commit"
[master (root-commit) 311e3a4] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

C:\Users\PC\Downloads\poio-proj2>git branch -M main

C:\Users\PC\Downloads\poio-proj2>git remote add origin https://gitlab.com/jkoszyk-agh-group/project2

C:\Users\PC\Downloads\poio-proj2>git push -u origin main
warning: redirecting to https://gitlab.com/jkoszyk-agh-group/project2.git/
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 222 bytes | 222.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote:
remote: The private project jkoszyk-agh-group/project2 was successfully created.
remote:
remote: To configure the remote, run:
remote:   git remote add origin https://gitlab.com/jkoszyk-agh-group/project2.git
remote:
remote: To view the project, visit:
remote:   https://gitlab.com/jkoszyk-agh-group/project2
remote:
remote:
remote:
To https://gitlab.com/jkoszyk-agh-group/project2
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\PC\Downloads\poio-proj2>
```

TASK 2.

Create a simple Hello world .cpp program and add the program in the repository. Modify the code and commit the changes. Create another branch.

TASK 3.

Modify your project multiple times committing changes. Use git log to review your history.

TASK 4.

Modify the code in another branch and commit your changes. Switch back to the main branch and search for the changes. The changes should not appear in branch main. Then merge your branch into the main branch and check the result.

TASK 5.

Simulate a merge conflict. Work in two branches and modify the same line of code in both branches. Attempt to merge them and observe the conflict. Resolve the conflict manually by editing the file and completing the merge.

Use:

`git diff` to see changes

`git restore` to discard changes

`git reset` to undo commits