Task 1 - Setting up a GitHub account

Create an account on the GitHub platform using your school email address. If you already had an account, add your school email address to the existing account in the settings. It's a good idea to generate an SSH key at home and use it for authentication.
https://docs.github.com/en/authentication/connecting-to-github-with-ssh

Task 2 - Git configuration

1. The first time you use it, you may need to configure Git.
2. The minimum set of settings you should configure are the username and email.
    1. git config --global user.name "sxxxx"
    2. git config --global user.email "sxxxx@pjwstk.edu.pl"

Task 2 - New repository

1. Create a new repository on GitHub.
2. Clone the new repository to your desktop.
3. Place the appropriate gitignore file in the repository. In our case, you can choose one of the predefined files for Visual Studio.
4. https://github.com/github/gitignore
5. Then add a new .NET console application to the repository. Make a commit named "Initial project" and push it to the online repository.
6. Then make 3 more commits. In each commit, make a modification to the code. Name each commit "Modification 1", "Modification 2", "Modification 3".
7. All commits should be visible online on GitHub.

Task 3 - New task

Assume you've been given a new task. You must create a static method that takes an array of ints and returns the calculated average.

1. Create a separate branch named "feature-average".
2. Then, on this branch, commit changes that implement the requirements. You can make one or any greater number of commits.
3. Then merge the created branch with the main branch. What default type of merge will git perform?
4. Check the current history of the repository with the command:

```
git log --oneline --graph
```

Task 4 - Rebase

In the next task, we need to add a static method that takes an array of ints and returns the maximum value.

1. Create a new branch feature-max
2. Then implement the described functionality by adding commits on the branch
3. At the end, merge your branch into the main branch. This time, try to perform the merge using rebase.
4. With the git log command, check what the repository history looks like
5. All changes should be pushed to the online repository

Task 5 - Conflict

In this task, we will simulate the creation of a conflict.

1. Create a new branch feature-new
2. Then, while on the newly created branch, try to modify the loop responsible for calculating the average. For example, you could change the name of the variable used within the loop.
3. Then commit on the feature-new branch.

4. Next, switch to the main branch and make a different modification to the same loop. For example, you could change the name of the variable to a different one.
5. Commit on the main branch.
6. In this way, both branches differ from each other. Additionally, we modified the same code on both branches. Such a situation should lead to a conflict.
7. Try to merge your branch with the main branch. Resolve the conflict. Push the changes to GitHub.
8. Finally, check the history of your repository with the git log command.