Exercise 5

Goals

- Add a remote repository
- Examine remote branches
- Exchange information with a remote repository

Reminder: *all text enclosed with* `<>` *denotes a placeholder to be replaced by a specific string appropriate in your context.*

Initialization

In [ ]:
```
# check current directory with "pwd"
pwd
# go to folder of this exercise using "cd"
```

In [ ]:
```
# execute this code at the very beginning to get access to the helper funct
source ../helpers.sh
init_exercise
```

Optional: clear notebook and restart

In case you mess up your notebook completely,
execute *reset* in the following cell. This will restore a clean environment!

In [ ]:
```
## only execute in case of (serious) trouble ##
## it will delete your entire work-directory ##
reset
```

Exercise

In this exercise we are going to use the same simple repository that we used in Exercise 2. Let's initialize it with our helper function.

In [ ]:
```
# this line will setup the simple Git-repository from Exercise 2 for you
init_simple_repo_remote
```

Add a remote repository

Let's add a remote repository to our local repository.

The setup script has already created one that you can use at:

../party_planning_remote

Use the line above as *remote_path* to the remote repository.

In [ ]:
```
# use "git remote add <some_remote_name> <remote_path>" to add the remote
```

In [ ]:
```
# use "git remote -v" to check that the remote was added correctly
```

The output should look something like this:

```
my_remote  ../party_planning_remote (fetch)
my_remote  ../party_planning_remote (push)
```

Get information from remote

So far all we have done is given our local repository the location of the remote repository. Now, we should get the information from the remote repository.

In [ ]:
```
# use "git fetch <remote_name>" to get information from the remote
```

In [ ]:
```
# use "git branch -a" to view ALL of the branches in your local repository
```

The output should look like this:

```
* master
  remotes/my_remote/master
  remotes/my_remote/updated_flyers
```

You can see that you now have local branches (`master`), and remote branches (`remotes/<remote_name>/<branch_name>`).

Add to local repository

The remote repository has a branch called `updated_flyers` . Let's check out this branch to work on it.

In [ ]:
```
# use "git checkout updated_flyers"
```

The output should look like this:

```
Branch 'updated_flyers' set up to track remote branch
'updated_flyers' from 'test'.
Switched to a new branch 'updated_flyers'
```

Git has automatically created a local branch in our local repository that tracks the remote branch from the remote repository.

Next, let's make a change in the "updated_flyers" branch.

Make a change in your local repository. Remember to do all modifications of the flyers directly via Jupyter Notebooks.

- Go to folder *work* and enter *party_planning*
- Open *flyer_A*
- Add more information to your flyer, i.e. music, dresscode, etc.

Don't forget to save your modifications before coming back!

In [ ]:
```
# add and commit your changes
```

In [ ]:
```
# check the status of your repository
```

The output should look like this:

```
On branch updated_flyers
Your branch is ahead of 'test/updated_flyers' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Send information to remote repository

Finally, we will send the changes we made to the remote repository.

In [ ]:
```
# use "git push"
```

The output should look something like this:

```
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 306 bytes | 306.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To ../party_planning_remote
   4f64811..a2244c4  updated_flyers -> updated_flyers
```