Object-Oriented Language and Theory
Object-Oriented Programming

Nguyen Thi Thu Trang, trangntt@soict.hust.edu.vn

Hands-on lab guidelines

1. You have to follow the instructions in the hands-on lab and complete all exercises. If you can not finish them at class, please do it at home.

2. For each lab, you will have to turn in your work twice, specifically:

   a. Right after the lab class: for this deadline, you should include any work you have done within the lab class time to Github.

   b. 10 PM the day after the class: for this deadline, you should complete all tasks/exercises in the hands-on labs including the guided exercises or the one with sample code, commit&push it to your **master/main** branch of the valid repository as mentioned in section 4.

   Each student is expected to turn in his or her work and not give or receive unpermitted aid. Otherwise, we would apply extreme methods for measurement to prevent cheating. If there is any question in the lab, please write down answers for all questions into a text file named "answers.txt" and submit it within your repository.

3. Git and github tutorial:

   a. Git tutorial:
      i. Learn Git commands in https://git-scm.com/book/en/v2
      ii. https://git-scm.com/docs/gittutorial
      iii. https://www.atlassian.com/git/tutorials
      iv. https://www.youtube.com/watch?v=HVsySz-h9r4&list=PL-osiE80TeTuRUfjRe54Eea17-YfnOOAx

   b. Github tutorial:
      i. https://www.freecodecamp.org/news/git-and-github-for-beginners/
      ii. https://www.youtube.com/watch?v=RGOj5yH7evk

4. Guidelines to git/github.com to submit your assignment/mini-project:

   a. Install Git (with optional tools: Source Tree, eGit for Eclipse…)
      i. https://www.youtube.com/watch?v=HVsySz-h9r4

   b. Create an account on https://github.com/

   c. Create a **private** repository naming "OOLT.ICT.20212.StudentID.StudentName" (for OOLT classes) or "OOP.20212.StudentID.StudentName" (for OOP classes), e.g. `OOLT.ICT.20212.20162912.NguyenHoangTu` or `OOP.20212.20162912.NguyenHoangTu`. If you don't follow this naming convention, your repository will be ignored.

   d. Add trangntt.for.student@gmail.com as a member of your repository

   e. Clone a repository, commit and push to the default and main one (master/main) or create branch as you wish:

```
cd <local_working_folder>
git clone <repository_url>
cd <repository_name> /* e.g. OOLT.ICT.20212.20162912.NguyenHoangTu */
git branch /* let you know which branch you're working on */
```

```
git checkout -b <your_branch> /* create and switch to your new branch */
```

Then you can make any changes such as create a new file, modify an existing file, delete a file…

  f. Commit and push after changing some resources:
```
cd <local_working_folder/repository_name>
git add -A /* -A: for all operations or . for without deletions in the current folder */
git status /* check status of the stage */
git commit -m "Your comment for the update" /* commit with comment*/
git push origin <your_branch> /* push from local repo to remote repo */
```

After checking carefully the new branch code, the leader can go to the Bitbucket to *Create merge request*, then *Approve merge request* to merge the code to the master branch.

  g. Pull without conflicts
```
git stash /* run this command if you want to ignore your current change */
git pull origin <your_branch>
```
  h. Pull and resolve conflicts
After you commit and push to the remote repository, if there is any conflict when creating merge request, you need to resolve conflicts.
```
git checkout master
git pull origin master
git checkout <your_branch>
git rebase master
git add -A
git status
git rebase –continue
```

You need to resolve conflicts by checking the resources that have conflicts. There are both versions in the resources so that you can observe and make decisions, e.g.
Having conflig:
```
<<<<<<< HEAD /* master branch in the remote repository */
      suggestor.setClientId(clientID);
      suggestion = suggestor.translate(input).replaceAll("-", " ");
======= /* your branch in the local repository */
      suggestion = suggestor.translate(input);
>>>>>>> Your comment for the update
```
Merging (merge your local repository version with the replaceAll() method, remove the first statement of the remote repository):
```
      suggestion = suggestor.translate(input).replaceAll("-", " ");
```

Then continue, force to push the merged version to the remote repository
```
git push origin <your_branch> -f
```