Final Project

Intro to Computer Science (CS-UH 1001) — Section 1

Spring 2019

1 Code of Conduct

(Last update 25 February 2019)

All assignments are graded, meaning we expect you to adhere to the academic integrity standards of NYU Abu Dhabi. To avoid any confusion regarding this, we will briefly state what is and isn't allowed when working on an assignment.

Any documents and program code that you submit must be fully made by yourself. You can of course discuss your work with fellow students, as long as these discussions are restricted to general solution techniques. Put differently, these discussions should not be about concrete code you are writing, nor about specific results you wish to submit. When discussing an assignment with others, this should *never* lead to you possessing the complete or partial solution of others, regardless of whether the solution is in paper or digital form, and independent of who made the solution, meaning you are also not allowed posses solutions by someone from a different year or course, by someone from another university, or code from the internet, and so on. This also implies that there never is a valid reason to share your code with fellow students, and that there is no valid reason to publish your code online in any form.

Every student is responsible for the work they submit. If during the grading there is any doubt about whether a student made the assignment themselves (e.g. if the solution matches that of others), we reserve the option to let the student explain why this is the case. In case doubts remain, or we decide to directly escalate the issue, the suspected violations will be reported according to the policies of NYU Abu Dhabi (see `https://students.nyuad.nyu.edu/campus-life/community-standards/policies/academic-integrity/`).

2 Introduction

The final project builds upon all the knowledge you gained in this course. In particular, you will create your own program to illustrate that you understand the programming features that we covered during the lectures. To make this project more exciting, we will let you create your own program (for example a simple game).

You can optionally make this project in groups of two, but you can also work on your own. If you want to work in a group, you have to form a group of two people yourself. Whether you are working on your own or in a group, you must use GitHub to share your code with us.

3 Creating Your Own Program

In this project you need to come up with your own program or game that you want to implement. Your idea must be complex enough so we can judge whether you master the programming concepts that you learned in this course.

3.1 Requirements

As more concrete guidelines, the program or game you create must at least contain the following features:

1. You must use Processing to create a graphical user interface.
2. The game must draw at least a few images on the screen.
3. The player should have a score and/or be able to win or lose the game.
4. The player should be able to interact with the game using the mouse and/or keyboard.
5. In the implementation of the game you must use at least one class (and preferably more).
6. It should have one or more sound effects.

You can implement an existing game if you want, or a simplified version of an existing game. You can also come up with a new game yourself. It can be any type of game: a classical game where you control a player that walks around, games like on your smartphone, a game to learn a new language, an interactive educational game, a smaller game like on Facebook, etc.

3.2 Idea and Group Proposals

To guide you in selecting a good program or game to implement, you must submit your proposal by email before you start working on it. We will then provide feedback on your idea, and whether it meets (or exceeds) our expectations. When submitting your idea proposals, you must also inform us whether you want to form a group of two, and if so with whom you will collaborate.

3.3 Usage of GitHub

You must put your code in a private repository on GitHub. At the beginning of the project, create a private repository, and invite Thomas (username `tpoetsch`) and Mathy (username `vanhoefm`) to your private GitHub repository. You can give us access by going to Settings on GitHub, selecting Collaborators, and the adding us by our username.

Make sure to put all files in the Processing directory on GitHub. This includes the `.pyde` and `sketch.properties` files, along with any other images or files that you use.

4 Advice and Extra Info

4.1 Processing Examples

Processing has a lot of built-in functions that are useful. You can find an **overview of all functions on their website at** `https://py.processing.org/reference/` There are also online tutorials to learn more about Processing at `https://py.processing.org/tutorials/`

Finally, there are also many example programs that use Processing at `https://processing.org/examples/`. These examples are written in Java and not Python. To convert the code from Java to Python, you need to know to difference in the syntax between these two languages:

- Python uses indentation to form code blocks, while Java uses braces (i.e. it uses `{` and `}`).

- When declaring and initializing variables in Python, you don't need to tell which type the variable will be.
- In Python there is no `;` at the end of every line of code.

4.2 Git Instructions

For this project you have to create a private repository on GitHub. Recall that on your local machine, you can initialize an empty Git repository and push it to GitHub using the following commands:

```
# cd mydirectory
# git init
# git add code.py
# git commit -m "initial commit"
# git remote add origin git@github.com:username/projectname.git
# git push -u origin master
```

You can get the precise URL from GitHub, and you only need to execute the `git remote` command once. When you have extra changes to push, you can now simply type `git push` without the extra arguments. For example, to commit a new file, or to commit changes to an existing file, you can execute the following commands:

```
# git add modifiedfile.py
# git add newfile.py
# git commit -m "describe your changes here"
# git push
```

The last command pushes your commit to GitHub, so that others can access it by either cloning the repository, or by executing `git pull` in case they already cloned the repository previously. If you previously cloned a repository, and you want to pull updates that your partner pushed, first commit any changes that you already made to the code (but don't push these changes to GitHub yet). Once you have no local changes, meaning the command `git status` shows `working tree clean`, you can execute the pull command:

```
# git pull
```

If there are no merge conflicts, you can now also execute `git push` to push your commits to GitHub. To track a Processing project using Git, open a terminal, navigate to the directory where the project is saved, and then let Git track all the files. To let Git track all files in a directory, you can execute `git add *`. For example, when saving a Processing project named `finalproject` on the Desktop on mac OS, you need to execute:

```
# cd /Users/mathy/Desktop
# cd finalproject
# git init
# git add *
# git commit -m "initial commit"
# ...
```

Replace `mathy` with your own account name. See Lecture 3 and 21 for extra instructions on how to use the terminal and Git, respectively. For more information and examples on how to use Git, you can also read the “Getting Started” and “Git Basics” chapters at `https://git-scm.com/book/en/v2`

Git is a very powerful tool, but can be difficult to use in the beginning. In case you encounter a problem with Git that you cannot solve, one option is to clone the repository from scratch into a new directory, and delete the old directory.

4.3 Grading

The points are roughly allocated as follows. However, do note that we might make changes to this grading scheme if deemed necessary.

Description	Score (/20)
Game proposal	2
Two commits before the second intermediate deadline	2
Regular commits throughout the project	2
Clear documentation	2
Final submission	12

As shown in the table, you can earn two points by submitting your idea proposal on time, and another two points by making at least two meaningful commits on GitHub before the second intermediate deadline (see Section 5). Two more points are earned if you regularly make meaningful commits throughout the whole project (at least two or three commits per week). The remaining 12 points are based on your final submission, which will be evaluated based on the following aspects:

- Required features mentioned in Section 3.1 are implemented.
- Usage of Git to share the code with us.
- The code is clearly documented with comments.
- Overall code structure and style of the code.
- Proper use of classes and functions.
- No sudden crashes or apparent bugs.
- Overall game design and graphics.
- Complexity of the project.

Note that a passing grade is always reached if your game contains the requirements that are listed in Section 3.1, independent of the quality of the code.

5 Submission Deadlines

1. The first intermediate deadline is on **Wednesday 1 May at 23:59**. You must email `mv100@nyu.edu` and `tp53@nyu.edu` a brief description of what you want to build for your final project. You must also specify whether you will work on your own, or with whom you will make the project with. For a group, only one student has to mail this information to us, but you must put the other student in CC (so they also receive the email).

2. The second intermediate deadline is on **Wednesday 8 May at 23:59**. By that time you must have invited Thomas and Mathy to your private repository on GitHub, and you must at least have two commits.

3. The final deadline of the project is on **Thursday 16 May at 23:59**. You must commit and push your code to GitHub no later than this deadline. The submission should include all files necessary to run your project and/or reproduce your results. The repository should **include a text file that briefly explains how the game works**, and it should give an overview of all the features that you implemented. This can be a PDF file or a simple `.txt` file created in, for example, Sublime Text. Additionally, a sample screenshot should be added that adequately represents what your game is about.

   On NYUClasses you can submit a link to your private repository.

   Note that your solution must work using Processing 3. In case your code does not work using Processing 3, your submission will not be graded.