Lab 06

Recap: Lab (Github) Workflow 📄 - How to Work on Labs

Follow these steps for every lab carefully to access, complete, and submit your assignment.

1. **Accept the Assignment**
    - Open the Lab Assignment Link the professor provided.
    - Click **"Accept the assignment"**. This will create your personal assignment repository on GitHub under the `OOP-Fall-2025` organization.
    - You'll be taken to your repository page. Verify that the URL looks like `github.com/OOP-Fall-2025/lab-number-yourusername`.
2. **Clone the Repository to Your Computer**
    - On your repository page, click the blue `<> Code` button.
    - In the dropdown menu, choose **"Open with GitHub Desktop"**.
    - GitHub Desktop will launch. Choose a preferred local folder on your computer to save the project and click **"Clone"**.
    - If asked "How are you planning on using this fork?", select **"For my own purpose"** and continue.
3. **Open in VS Code and Start Coding**
    - In GitHub Desktop, ensure the "Current repository" is the one for this lab.
    - Click the **"Open in Visual Studio Code"** button.
    - VS Code will open the project folder. You can now begin writing your solutions in the `Lab6.java` file.
4. **Save and Submit Your Work**
    - **Commit (Save) Changes**: As you work, save your file in VS Code (`Ctrl+S` or `Cmd+S`). To record your progress, go to the **Source Control** tab (the fork icon) on the left sidebar in VS Code. Type a descriptive message in the message box (e.g., "Finished Task 1 and 2") and click **"Commit"**. You must enter a message.
    - **Push (Submit) to GitHub**: When you are finished with the lab or want to back up your work, go back to GitHub Desktop. Click the **"Push origin"** button at the top of the window. This sends your committed changes from your computer to your GitHub repository online.
5. **Verify Your Submission**
    - After you push, you can click **"View on GitHub"** in GitHub Desktop to open your repository in the browser.
    - On the GitHub website, make sure you are viewing the `main` branch and confirm that all of your latest code is visible.

Lab 06 Tasks

`Classy`

Task 1: Crash Course

Objective: Create a Course class and use it in your main program.

Course Class:

- Create a class called Course (You need to create a file called `Course.java`)
- Create 2 public properties: department (String), number (int)
- Create a method called `print()` which prints the department and number together.

Main:

- Inside of main, instantiate the class.
- Ask the user for the department and number (set the properties inside the class).
- Call the print method inside the class.

Example Output

```
Department: CS

Number: 122

CS122
```

Task 2: Head of the Class

Objective: Create a Student class with initialized properties.

Student Class:

- Create a class called Student
- Create 3 public properties: name (String), year (int), gpa (double)
  - Initialize them to "(unknown)", 0 and 0.0
- Create a `print()` method that prints all properties on one line.

Main:

- Inside of main, instantiate the class.
- Call the print method.
- Ask the user for the name, year and gpa (set the properties)
- Call the print method again

Example Output

```
(unknown) 0 0.0
```

```
Name: Allison

Year: 2020

GPA: 3.8

Allison 2020 3.8
```

Task 3: It's D&D

Objective: Create a PlayerCharacter class with a constructor that generates random stats.

PlayerCharacter Class:

- Create a class called PlayerCharacter
- Create a private property called name (string)
- Create 4 private properties (all are integers): strength, dexterity, intelligence, charisma.
- Create a constructor which only takes name as a parameter.
    - It should set the name private variable
    - It should set each of the other properties to a random number from 1 to 20
- Create a method called `stats()` which prints the name and the other properties.

Don't forget to import random

Main:

- Inside of main, ask the user for the name first!
- Instantiate the class. Remember you are only giving it the name, the rest is automatic.
- Call the stats() method.

Example Output

```
Name: Spellbound

Spellbound
Strength: 12
Dexterity: 14
Intelligence: 18
Charisma: 15
```

(Your result may vary because it is random)

Task 4: Constructions

Objective: Create a Transformer class with a constructor and conditional method.

Transformer Class:

- Create a class called Transformer
- Create 2 **private** properties: name (String), team (String)
- Create a constructor that takes name and team as parameters and sets the private variables.
- Create a method called `action()`. If they are a Decepticon they should attack and if they are an Autobot then they should protect.

Main:

- Inside of main, ask the questions first!
- Instantiate the class.
- Call the action method.

Example Output

```
Name: Megatron

Team: Decepticon

Megatron attacks!
```

OR

```
Name: Optimus

Team: Autobot

Optimus protects!
```

Task 5: Mutants

Objective: Create a Mutant class with accessors and mutators (get/set methods).

Mutant Class:

- Create a class called Mutant
- Create a **private** property named "name" with datatype String and "power" with type int.
- Create method called **setName** which takes name as a parameter and sets the private variable name.
- Create a method called **getName** which returns the private variable name.
- Create method called **setPower** which takes power as a parameter and sets the private variable power.
- Create a method called **getPower** which returns the private variable power.

Main:

- Inside of main, Instantiate the class.
- Ask the user for a name, and set the name using **setName**

- Ask the user for power level and set power using **setPower**
- Print the name using **getName** and power using **getPower**

Example Output

```
Name: Storm

Power: 10

Storm has power level 10
```

Need Help?

Ask me or your classmates for help! We are in the same room~~

Finished?

When you are done with the labs (finished and committed on GitHub properly), call me over and show me. Tell me your name and I'll mark you as done!