**Lab 1 Introduction to Programming**

**Due Date: Saturday Sep 7th 6 pm in Moodle**

This lab will introduce you to the process of working with jGRASP to write your code.

**You should have already installed Java and jGRASP on your laptop. If not, continue working with your group. You can install the software later.**

**Learning Objectives**

* Meet your peers, form teams, and collaborate.
* Use jGRASP to edit and write a simple Java program.

|  |  |
| --- | --- |
| **(5 min) Activity 1. Introductions** | Starttime: 9:18 am |

Form a group of four, introduce yourself to your group mates and decide who will be what role for today and write the names in the table below; we will rotate the roles each week.

|  |  |
| --- | --- |
| Role | Name |
| **Manager**: reads the questions aloud, keeps track of time and makes sure everyone contributes appropriately. | Lola |
| **Presenter**: talks to the instructor and other teams. Reports out the **highlighted** questions to the TA or to other teams. | Nishant |
| **Recorder**: **records all answers in a Google doc** and ensures the team agrees on responses. | Maya Geer |
| **Reflector**: Considers how the team could work and learn more effectively. At the end of the lab, provides team reflection in a questionnaire in Moodle. | Jinxuan |

|  |  |
| --- | --- |
| **(10 min) Activity 2. Introduction to jGRASP** | Starttime: |

**2.1. Download zip file and create a folder**

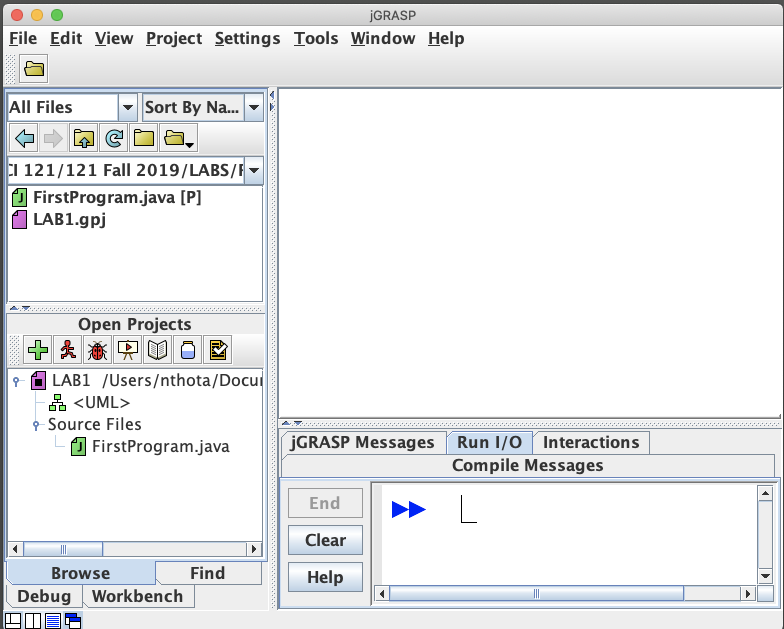
1. Create a COMPSCI121 folder on your laptop. Ideally you should create it in your “Documents” folder and not the Desktop or in your Downloads folder.
2. Log in to Moodle, download the **LAB1.zip** file and save it in the COMPSCI121 folder.
3. Unzip by either clicking on it or right click and extract the files. (You should then delete the zip file so that you don’t use it).

A **LAB1** folder will be created. For example, it might look like this (could be different on Windows/Mac/Linux):



Open **jGRASP** by clicking on the icon on your laptop. 

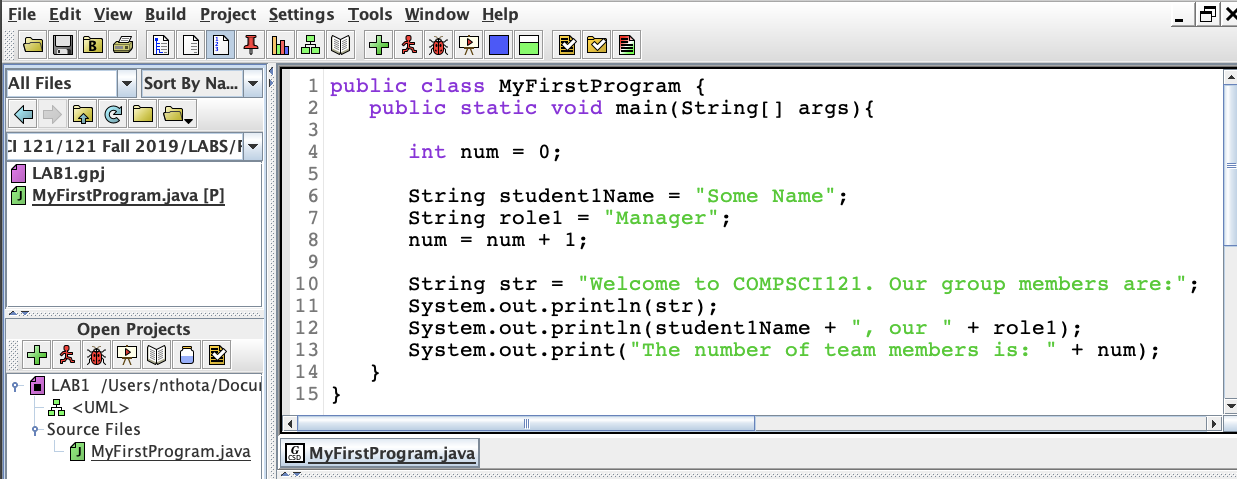
Using the Navigation pane () navigate to the LAB1 folder. Click on **LAB1.gpj** to open the jGRASP project file. Check that you can see the “**Open Projects**” window.



**Report to your TA that all team members have finished step 2.1. Otherwise, help each other!**

**2.2 Open a Java file**

Inside the project is the **M**y**FirstProgram.java** file. Open the file (you may double click). You can now see the code in the main window.



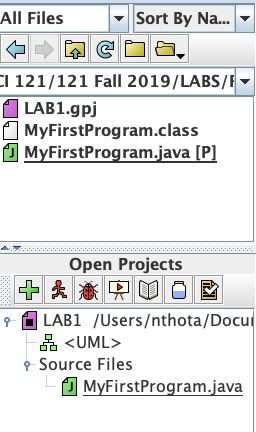
**Q2.2.1. What do you think is the purpose of this program? Discuss with your team and then write the answer below. Give the overall purpose, not a line-by-line list.**

**Answer: To print our group’s names and roles and at last printing the number of members in our group.**

**2.3 Compile the program**

The next step is to compile the program to check for errors.   
Click on the green cross () in the project pane to compile the java file.

Note the messages of successful compilation in the interactions pane, which means there are no errors in the file. See the new file that is created: **FirstProgram.class**. Refresh the folder by clicking on the curved arrow if you cannot see the class file.

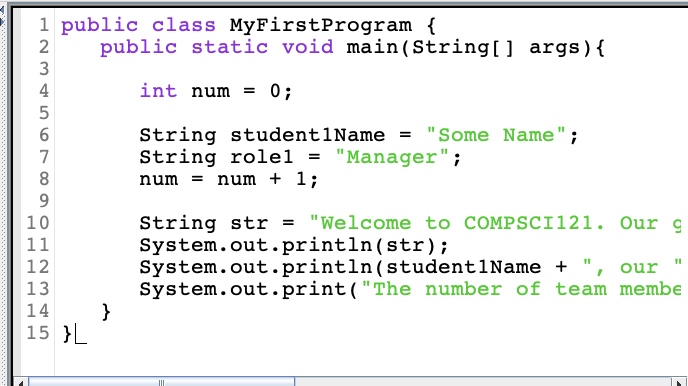


**Q2.3.1. What is the .class file used for?**

**Answer: It is the compiled code (byte code) that is formed after the code is compiled in compiler. That .class file can be interpreted by the Java Virtual Machine.**

**2.4 Use line numbers**

Insert line numbers in the java file by clicking on the line number icon () in the menu bar. You can see the line numbers as in the figure below:



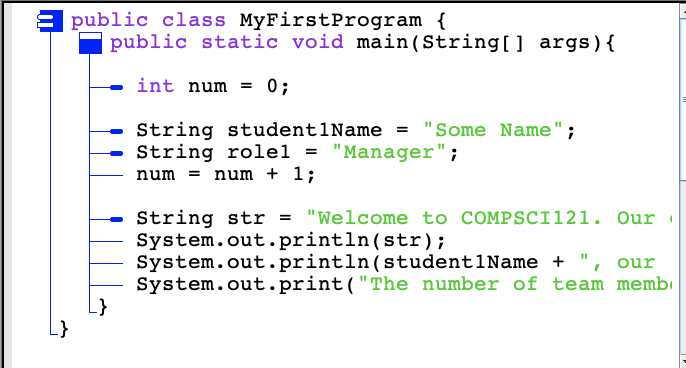
**Q 2.4.1 Why are line numbers useful? Discuss with your team and then write the answer below. List as many reasons as you can.**

**Answer:**

* **locate the general area of errors easily**
* **use for ease of communicating when collaborating (i.e. “look at line number …”)**

**2.5. Indenting code**

Click on the “Generate CSD” icon ( ). Your code should now look like in the figure below.



**Q2.5.1 What do you think is the purpose of the blue lines?**

**Answer: It organizes the code by grouping parts of code into functions and classes to show the program’s structure.**

Click on the “Remove CSD” icon ( ) to see the indented code without the vertical lines.

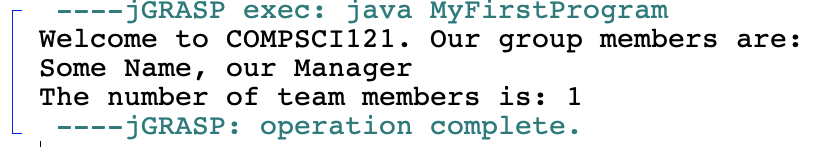
You can use the TAB key to indent your code when working in jGRASP.

**Q2.5.2. Why should you indent your code?**

**Answer: To better understand the structure and organization of your program. This makes it easier to develop and maintain your program.**

**2.6. Run the program**

The next step is to run the program to make it do something useful. Click on the red running man icon () to run the file. You should see the output below:



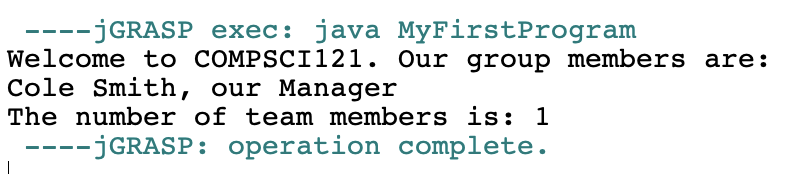
**Good job! To this point, you have opened, compiled, and run a java program!**

**Report to your TA that all team members have finished step 2.6. Otherwise, help each other!**

|  |  |
| --- | --- |
| **(15 min) Activity 3. Modifying an existing Java file** | Starttime: 9:37 am |

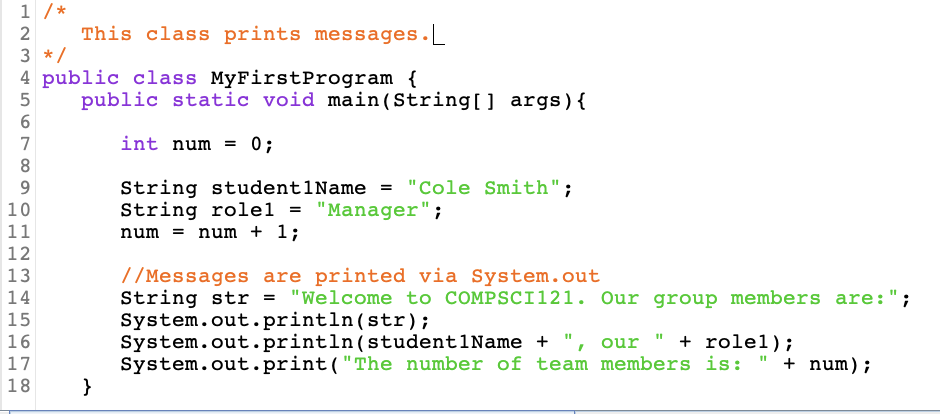
**3.1. Edit a java file**

Edit the existing java code so that the program will output the name of the manager of your team (rather than “Some Name”). Compile and run the program. For instance, if your manager is named Cole Smith, then your output should be:



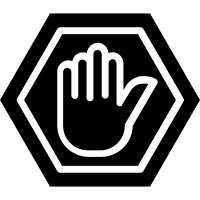
**3.2. Add comments to the java file**

In **MyFirstProgram.java** add comments to the file (see statements in orange in the figure below).



**Q3.3.1 Compile and run FirstProgram.java. What does the compiler do with the comments?**

**Answer: The compiler ignores the comments. These comments are meant for the human programmer to understand the code, not for the computer/compiler.**

Report to your TA/UCA that all team members have finished all steps in Activity 3. Otherwise, help each other!

**Good job! To this point, you have opened, compiled, run, and edited a java program!**

|  |  |
| --- | --- |
| **(15 min) Activity 4. Creating a new Java File** | Starttime: 9:41 am |

**4.1 Create and save a new Java file and add to project**

In jGRASP, go to the “**File**” menu, select “**New**”, then “**Java**”. A new edit window opens. You will create a simple Java program that prints out the information of your team and teammates.

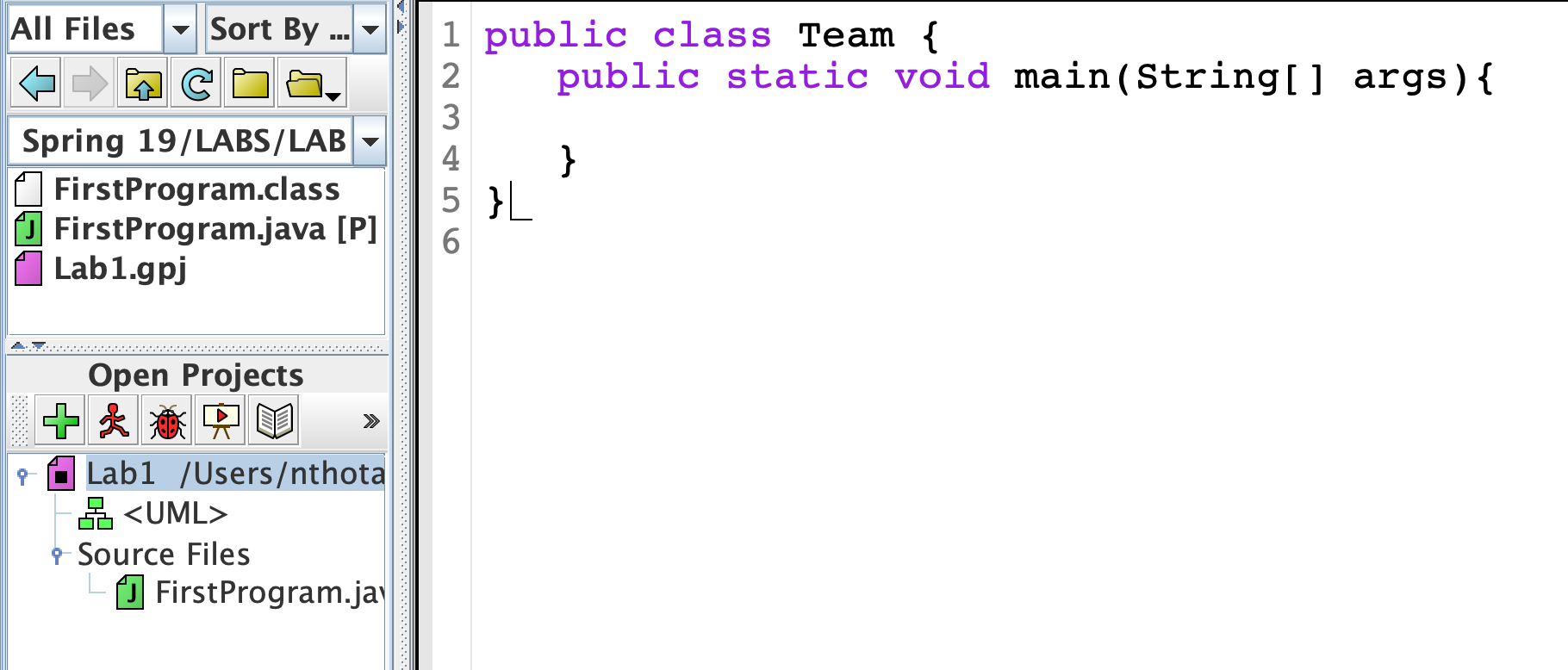
Type (or copy & paste) the following class definition into the new window:

**public class MyTeam {**

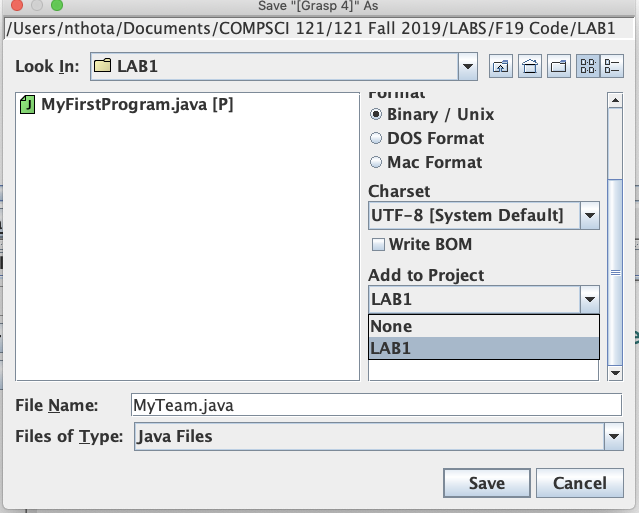
**public static void main(String[] args){**

**}**

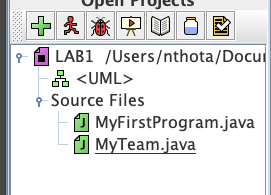
**}**

****

1. Click on the compile icon (the green cross). You will see the ‘save’ dialog box.
2. Click on the “**Save**” button. A file browser is displayed at the directory shown in the navigation pane (should be “LAB1”). Notice that the file name field at the bottom is already populated with “**MyTeam.java**”.
3. Click on the dropdown “**Add to Project**”and select **LAB1**.
4. Click “**Save**”.



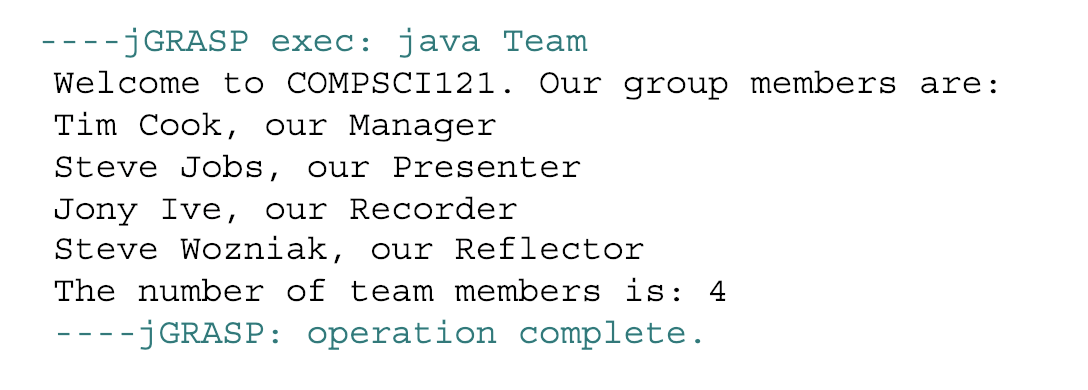
Check the **Open Project**s window that the new file has been added to the project.



**4.2 Write a Java program to print out your team information**

Discuss with your teammates how to make changes to the program so that when you run **MyTeam.java,** the output will be the names of all your team members, their respective roles, and the number of team members.

For example, a team with Tim Cook as Manager, Steve Jobs as Presenter, Jony Ive as Recorder, and Steve Wozniak as Reflector will have the following output:



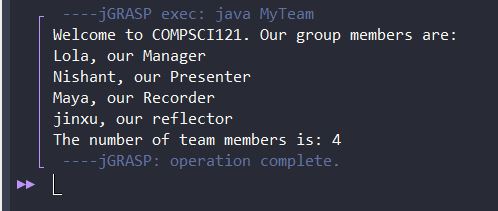
You are welcome to reuse the code you already have in **FirstProgram.java**.

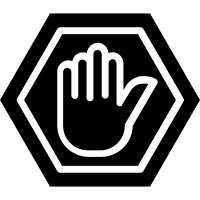
**Hints:**

1. Start by adding the name of a team member and the role (see the code in FirstProgram.java).
2. Increment the **num** variable. (num = num + 1;)
3. Add the **System.out.println** statement to print the name and role of the member.
4. Run the file to see if the output is correct. Correct your errors, if any.
5. Then add the next member and repeat the statement to increase the number of team members. Run the file each time to test the output.
6. Continue till all members are added.

4.2.1 Take a [screenshot](https://lifehacker.com/5825771/how-to-take-a-screenshot-or-picture-of-whats-on-your-computer-screen) of your final output and paste here.

**Answer:**

****

Show your TA/UCA that all team members have the desired output. Otherwise, ask for help!

**Good job! You have now created a jGRASP project with two Java main classes. You are well on your way to becoming a programmer!**

**Create a pdf of this worksheet file and upload in MOODLE. There is no need to upload the java files.**

**\*\*Remember each member of the group needs to do this individually!**

**\*\*\*The Reflector has to fill in the Reflector's Questionnaire in Moodle!**

**It is also OK if you could not complete the lab. The grace period is Sept 7 at 6 pm (the Moodle assignment will not accept submissions after this date or time).**

**CHALLENGE**

Did all your group members finish early? Then answer the following questions.

1. Why did we declare an int variable called num?

**Int variables are used for whole numbers, which we needed to keep count of the members to print later.**

1. Why did we use Strings in our program?

**Strings let us print out words**

1. What is the difference between print and println?

**print() prints whatever you input. println() does the same thing, but it also makes anything that is subsequently printed print in a new line.**