Abstracting the Tree Exploration in Existing Module Code
5004 Object Oriented Design

In lessons 6 & 7 the hierarchy was abstracted using generic tree nodes, but the code was not supplied in a downloadable form. I thought it would be an interesting challenge for you to implement this on your own and add some additional functionality. I created a proof of concept and it led me down some interesting rabbit holes. I would like you to duplicate that journey.

1. **Goals:**

    - Demonstrate you can examine existing code
    - Extend the functionality of an existing complex program
    - Walk through a modules code
    - Approach class design from a different angle

2. **In Recitation:**

    ~~At this point you all have experience setting up and creating projects. I'd like you to share that experience with each other and hold a strategy session. Make sure your strategy sessions don't include sharing code. By the end of recitation make sure you have a plan going forward and a frame to build from.~~

    ~~Your TA will divide you into strategy groups and check on your progress. Your group should have a group leader. If one isn't self-selected, your TA will appoint one.~~

    ~~Objective 1 : Share your experiences~~

    ~~In your individual groups discuss the following and remember your objective is to learn and to help your fellow students.~~

    ~~Preliminary discussion:~~

    1. ~~With these lab assignments what has been your biggest challenge?~~
    2. ~~How do you get focused and started with the assignments?~~
    3. ~~How do you make sure you aren't programming by accident?~~

    ~~Collect your answers to these questions for your recitation ICE.~~

~~Objective 2 : Strategy session (This will likely take the longest of the objectives)~~

> ~~Review all of the tasks for this assignment and "verbally" strategize with your group. Review any of the code from the module you don't understand and lookup or demo. For example, what's a bifunction?~~
>
> ~~What do you think is going to be the hardest and what is going to be the easiest?~~

~~Objective 3 : Create your plan~~

> ~~Take a few minutes to create a plan of attack for this assignment. Perhaps a step-by-ste list.~~

~~Do not implement any of the tasks as a group.~~
~~Do not copy code.~~

~~Submit this plan as part of your recitation ICE work.~~

Complete part 2 and part 3 from this week's ICE assignment.
Complete at least one of the challenge tasks.
Submit your code as proof of ICE completion.

3. Instructions:

Download the following files from this week's module:
https://www.dropbox.com/s/fnsuxkz1e283vxw/StarterCode.zip?dl=0

Load these files into a project so you can explore and modify them. I've included an OrganizationDriver file that should look similar to this week's demo code. Your first step should be to get this code to work without removing any of the comments.

As part of this, I recommend adding comments to files and making notes. Make sure you know what everything does.

Now your goal is to abstract out this program. In the end your OrganizationImpl file will use a generic tree node instead of a specific node.

Some of the files are carried over from the demo code, but the following files are mostly blank and will need to be completed by you:

- OrganizationImpl.java
- AbstractTreeNode.java
- TreeNode.java
- LeafNode.java
- GroupNode.java

…and any other classes needed to make your application run

In addition, there are three numbered tasks in the driver. I want you to add this functionality to your completed application.

```
//Task 1:
//This you'll have to add. We did it as part of the
demo code, but you'll have to modify it to make it
work with the new structure
//Print all the employees along with all their
information
//MonsterCorp.printEmployees();

//Task 2:
//This too is something we did in the demo code, but
you'll have to modify it to work here.
//The syntax on this is a bit challenging, but if you
study the getSizeByGender you should be able to
figure it out
```

```
//Remember, don't try to guess. Try and understand
the existing code and then see how you can change it
//This code segment should return a single int value:
the number of employees who make 300.00 annually

//System.out.println(MonsterCorp.getSize(m->m.getAnnu
alPay()==300.00));

//Task 3:
//Print out a list of just employee names using
allEmployees()
//This is not a hard task. Just make sure you
understand what all Employees returns
//This function should return a string of all
employee names that you can then print as output
```

Warning: code is provided in the video and on the module, and it helps, but you'll need to understand it and make the correct modifications to make it work.

Tip: don't forget imports. For example, Function and BiFunction are standard java interfaces.

Tip: don't guess. Try to understand the example code first.

Tip: take it one step at a time. Comment out the code to get it to a working state and then add things in one at a time.

Tip: when unsure what to do, check the rubric.

4. Extensions:

Outside academics you will not get specific requirements. Each lab assignment is worth 100 points, but the base requirements will only get you to 85% - 90%. If you want an A, you'll have to find a way to go above and beyond what is asked. I'll often make some suggestions to you in this section, but it is entirely up to you what you'd like to add to the assignment. Make sure you know who your grader is and discuss extension expectations with them. You won't have to do all of the extensions to get 100% credit.

Extension suggestions:

- Extend the functionality of the existing code beyond what is ask

- Complete some of the extra tasks
- Add testing

5. Report:

Each assignment must include a short report. The generation of this report should take you no more than 15 minutes. This gives you a chance to reflect back on what you learned and it makes grading easier on your grader. For this report, I want the following sections:

1. Reflection (*What did you learn?)*
2. Extensions (*What extensions are you requesting?)*
3. Grading Statement (*Based on the rubric, what grade do you feel you deserve? Be honest.)*

6. Submission:

Please read carefully. Failure to follow submission instructions can result in a reduced score.

Submit all files on Canvas under the appropriate assignment. Make sure to include the following named as follows:

Submit your files as a single zip file named: `"Your Name"_"Assignment".zip`

Your zip file should contain all files needed to make the application run. Include your report as: `"project_report_05.pdf"`

Submission checklist:

- ☐ Did you include adequate comments?
- ☐ Did you include comment blocks at the top of each file?
- ☐ Did you name your files as requested?
- ☐ Does your code compile?
- ☐ Did you remove any package lines generated by your IDE?
- ☐ Did you take care of any warnings presented by your IDE?

7. Rubric

	Possible	**Given**
Pre-task Getting setup		
Basic driver working without additional tasks	20	0
Additional functionality added		
Task 1: printEmployees	10	0
Task 2: getSize(Predicate)	10	0
Task 3: allEmployees printed	10	0
Abstraction added		
Code functions as an abstracted tree	20	0
Misc		
Additional comments added explaining functionality	5	0
Report	5	0
Code Quality (correct indentation, comment blocks, variable naming, etc)	10	0
Not included in total possible:		
Does not compile	-100	0
Extensions (Not calculated without report)	10	0
Late penalty	-20	0
Creative or went above and beyond	10	0
Code contains warnings	-20	0
TOTAL POINTS POSSIBLE out of 100	90	0