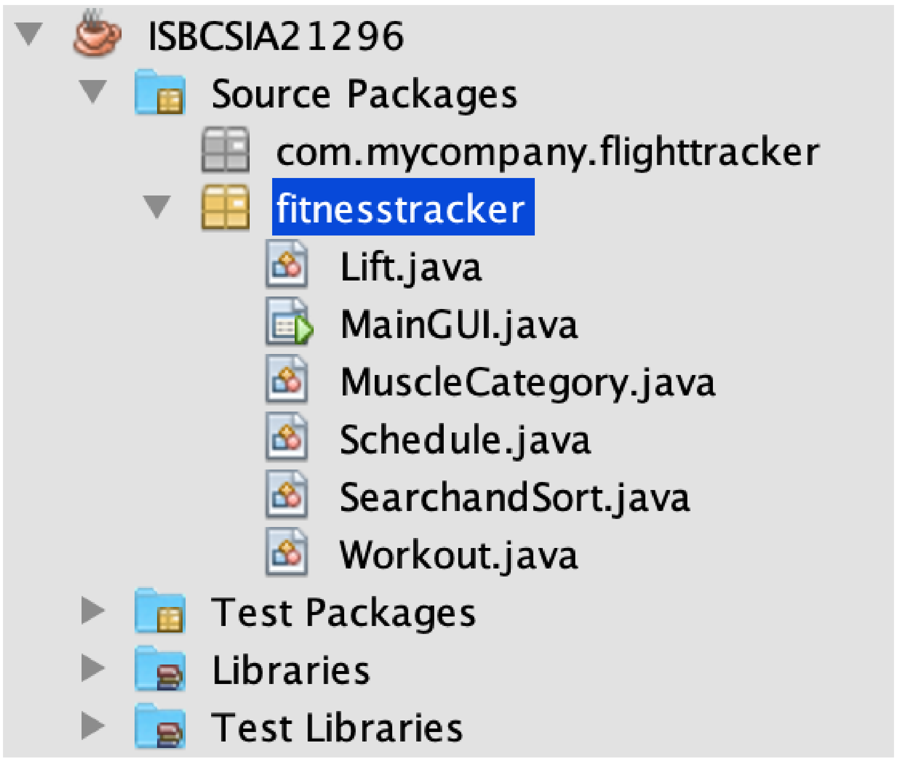
# Introduction

I created a program called WorkoutPlanner, which is a program for gym students to plan and schedule their workouts and see more lifting analytics. I used the NetBeans IDE and an OOP approach to make a GUI Interface which would be easy for gym students to use.

Summary of Programming Techniques

- parameter passing
- decimalFormat Solution
- for loop
- recursion
- method returning a value
- user defined objects made from an OOP "template" class
- encapsulation of private methods that work on public attribute of a "template" class
- making an array of objects
- simple and compound selection (if/else)
- Bubble sort
- Linear search
- saving to a file
- opening a file to a table
- error handling - GUI tabs
- GUI popup menus
- Use of a flag value (such as -999 or "not set yet")
- inheritance between a super class and a sub class
- LinkedList

## Structure of the Program



```
ISBCSIA21296
    Source Packages
        com.mycompany.flighttracker
        fitnesstracker
            Lift.java
            MainGUI.java
            MuscleCategory.java
            Schedule.java
            SearchandSort.java
            Workout.java
    Test Packages
    Libraries
    Test Libraries
```

### Relationships

Aggregation:
- Workout has a MuscleCategory

Dependency:
- MuscleCategory uses Lift

### OOP Features

### Aggregation

I used the OOP concept of having a one way relationship to break down the process of having the different features of a workout. This made sense because there are many different types of muscle categories, but there are underlying features of a workout that are going to be universal for all workouts

Encapsulation?

Polymorphism?

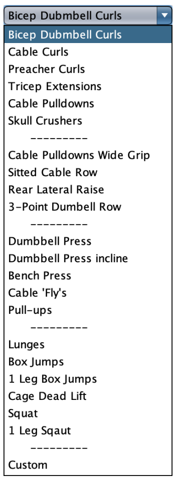
## Data Structures Used

I made use of linkedList for the queue of workouts to which the information processed is to go to. This was a logical choice, since queues are lists , and as such are dynamic in size. linkedList, like all dynamic data types, does not waste memory with excess nodes. I chose a list because the different workouts need to be compiled together and the way someone workouts out is universal and will stay the same with the different components. Sets, reps, and weights are the way to break down a traditional weight room workout.

The declaration of the LikedList of Workouts.

```
LinkedList<Workout> scheduleQueue = new LinkedList<Workout>();
```

Adding a choice of several different lifts.



```
Bicep Dumbbell Curls
Bicep Dumbbell Curls
Cable Curls
Preacher Curls
Tricep Extensions
Cable Pulldowns
Skull Crushers
    ---------
Cable Pulldowns Wide Grip
Sitted Cable Row
Rear Lateral Raise
3-Point Dumbell Row
    ---------
Dumbbell Press
Dumbbell Press incline
Bench Press
Cable 'Fly's
Pull-ups
    ---------
Lunges
Box Jumps
1 Leg Box Jumps
Cage Dead Lift
Squat
1 Leg Sqaut
    ---------
Custom
```

## Main Unique Algorithms

Adding workouts to the LinkedList.

```
if(liftCB.getSelectedItem().equals("Custom")){
    MuscleCategory muscleCategory =  new MuscleCategory(categoryComboBox.getSelectedItem() +"",
            intensityCB.getSelectedItem() + "");
    int weekNumber = Integer.parseInt((String)weekNumberCB.getSelectedItem());
    int dayNumber = Integer.parseInt((String)dayNumberCB.getSelectedItem());
    String lift = customWorkoutTF.getText() + "";
    double weight = Double.parseDouble(weightTF.getText());
    int reps = Integer.parseInt(repititionsTF.getText());
    int sets = Integer.parseInt(setsTF.getText());
    Workout workout = new Workout(weekNumber, dayNumber, muscleCategory, sets, reps,  weight, lift);
    scheduleQueue.addLast(workout);
    customWorkoutTF.setText("");

}else{
    MuscleCategory muscleCategory =  new MuscleCategory(categoryComboBox.getSelectedItem() +"",
            intensityCB.getSelectedItem() + "");
    int weekNumber = Integer.parseInt((String)weekNumberCB.getSelectedItem());
    int dayNumber = Integer.parseInt((String)dayNumberCB.getSelectedItem());
    String lift = liftCB.getSelectedItem() + "";
    double weight = Double.parseDouble(weightTF.getText());
    int reps = Integer.parseInt(repititionsTF.getText());
    int sets = Integer.parseInt(setsTF.getText());
    Workout workout = new Workout(weekNumber, dayNumber, muscleCategory, sets, reps,  weight, lift);
    scheduleQueue.addLast(workout);
}
```

The purpose of this if statement is to have the option of a custom lift where the lift is manually inputted by the user.  A workout is created with all of the different parameters where the information is collected from the user's input with .getText() and .getSelectedItem().

## Software Tools Used

The main software tool I used was the Netbeans IDE.  IDE stands for Integrated Development Environment, and as the name suggests, it gives us more than just a text editor for typing in code; included is a debugger, a compiler, even a spell checker, and other useful features are integrated as well.

## User Interface/GUI Work

One of the key interface capabilities of Netbeans is the ability to drag and drop Java Swing objects into a file, without having to hardcode them.  This makes it easier for me as the creator and saves time.  This also makes the database somewhat familiar to the client because the GUI features are common for most applications.