Group 9 SOLID and GRASP Write Up

Contributors (6 Total):

- Nathan Zheng
- Dat Mai
- Taylor West
- Saddiq Rupani
- Jay Patel
- David Casanova

Principles (6 Total):

1. Interface Segregation Principle: Our design of the system uses the RepeatableTask interface, which is a small, single-purpose interface that is implemented by the WeeklyTask class.
2. Open-Closed Principle: Our design uses Task and TeamMember as abstract classes. This means that future types of Tasks and TeamMembers can be added without having to change old code and instead allow new classes to be developed. This follows the principle of disallowing changes to existing classes, but allowing new classes to be plugged into extension points.
3. Low Coupling (Stamp Coupling): Our design uses the addTeamMember and removeTeamMember methods in TeamMember's methods of joinProject and leaveProject.
4. Information Expert: Our design assigns the Project class with the job to add and remove tasks and teamMembers, because the Project class already has the list of current tasks and teamMembers to alter. We assign the responsibility to a class that already has the necessary information to fulfill the responsibility.
5. High Cohesion (Communicational Cohesion): Our design's Project class has methods of adding and removing tasks and teamMembers, which operates only on the ArrayLists provided in the Project class. Because the methods manipulate the same data fields, this is communicational cohesion, with bits of procedural cohesion as well.
6. Single Responsibility Principle: The TeamMember class fulfills the Single Responsibility Principle as a member is able to join or leave projects with the joinProject and leaveProject methods. The only reason for TeamMember to change is if a TeamMember wants to join or leave a Project.