Laboratory Work #25

Java Object-Oriented Programming. Polymorphism

LEARN. GROW. SUCCEED.

 Department: <Software of Information Systems and Technologies>
Faculty of Information Technology and Robotics
Belarusian National Technical University
by Viktor Ivanchenko / ivanvikvik@bntu.by / Minsk

ЛАБОРАТОРНАЯ РАБОТА #25
ООП в Java. Полиморфизм

Цель работы 

Углубить свои фундаментальные знания в использовании методологии ООП, а также научиться практически применять динамический полиморфизм с использованием средств, которые предоставляет язык Java.

Требования 

1) Необходимо скорректировать UML-диаграмму взаимодействия классов и объектов программной системы с учётом вносимых дополнений и изменений.
2) При корректировки программной системы необходимо полностью использовать своё ООП-воображение и по максимум использовать возможности, которые предоставляет язык Java для программирования с использованием методологии ООП.
3) Каждый пользовательский тип должен иметь адекватное осмысленное имя и информативный состав (соответствующие конструкторы: по умолчанию, с параметрами, конструктор-копирования; ***get***- и ***set***-методы для доступа к состоянию объекта; корректно переопределённые методы базового класса *Object*: ***toString()***, ***equals()***, ***hashCode()*** и др.).
4) Также рекомендуется придерживаться ***Single Responsibility Principle***, ***SRP*** (принципа единственной ответственности): у каждого пакета, класса или метода должна быть только одна ответственность (цель), т.е. должна быть только одна причина изменить в дальнейшем соответствующий блок кода.
5) Добавляемые классы необходимо грамотно разложить по соответствующим пакетам, которые должны иметь «адекватные» названия и быть вложены в указанные стартовые пакеты: ***by.bntu.fitr.poisit.nameofstudent.nameofproject***.
6) В соответствующих важных компонентах программной системы необходимо предусмотреть «защиту от дурака».

7) В качестве хранилище данных использовать Java-массивы!!! Все контейнерный класс должны поддерживать расширяемость, т.е. они не ограничены в объёме хранимых данных и динамически должны расширяться.
8) При разработке программ придерживайтесь соглашений по написанию кода на *Java* (***Java Code-Convention***) !!!

Основное задание 

Необходимо произвести рефакторинг программной системы, созданной в предыдущей лабораторной работе, следующим образом:

- наделить сущности (бизнес-объекты), которыми манипулирует бизнес-логика, соответствующим полиморфным поведением таким образом, чтобы не пришлось переписывать саму бизнес-логику;
- используя поведенческий шаблон проектирования стратегия (The Strategy Pattern) внедрить логику по поиску соответствующих данных в предметную область, а также добавить различные типы сортировок (сортировки по разным критериям сущностей из предметной области);
- на уровне всей программы добавить глобальные характеристики, на базе которых реализуется бизнес-логика (на пример, механизм отслеживания количества создаваемых объектов предметной области, которыми манипулирует система при выполнении основных расчётов программной системы).

Контрольные вопросы 

1. Что такое полиморфизм?
2. Какие разновидности полиморфизма бывают в языках программирования?
3. Как реализуется статический полиморфизм в языке Java?
4. Что такое перегрузка методов (функций или операторов) в языке Java?
5. Как динамическая типизация влияет на реализацию динамического полиморфизма в языке Java?
6. Чем полиморфизм языка Java отличается от многих других языков программирования?
7. Что такое виртуальный метод? Что такое переопределение методов в языке Java?
8. Что такое абстрактный класс? А что такое абстрактный метод?
9. Можно ли реализовать абстрактный класс (метод) в языке Java? Если да, то как?
10. Что должен всегда делать абстрактный метод в языке Java при его объявлении в базовом классе?