Маляр Иван Владиславович

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2**

**НА ТЕМУ: «ХИМИЧЕСКАЯ ЛАБОРАТОРИЯ»**

***Ссылка***

<https://github.com/malyariv/labs/tree/master/lab2>

***Реализованная функциональность:***

* стандартный консольный интерфейс ввода/вывода;
* программа генерирует задание как синтез какого-либо химического соединения;
* для выполнения задания предлагается на выбор 5 реагентов;
* также предлагаются две операции: мешать и нагреть;
* есть возможность очистить реактор — место, куда складываются реагенты и протекают реакции, посмотреть его содержимое, получить новое задание или выйти;
* список возможных реакций можно получить из файла или базы данных (тут MySQL), список реагентов из файла, список заданий расположен в enum;

***Описание UML-схемы и реализованных классов:***

***Основные классы***

* The class *Main* is a console application simulating chemical laboratory where user can some operations with chemical reagents.
* The class *UserInteraction* provides interaction between user and application using console interface.
* The class *Reactor*, implementing interface *Observerable*, is a container for objects implementing interface *ChemicalObserver*. This class is a model of chemical reactor where chemical interaction between chemical reagents occurs.
* The class *Substance*, implementing interface *ChemicalObserver*, is a model of a chemical reagent. It contains methods simulating response of the reagent upon different exposures.

***Вспомогательные классы:***

* The class *TaskGenerator* generates a chemical compound and a list of answers containing chemical reagents.
* The class *ChemicalGenerator* implements the factory method pattern. It generates *ChemicalFactoryImpl1* or *ChemicalFactoryImpl2* depending on number of arguments.
* The class *ChemicalFactoryImpl1* generates *ChemicalObserver* objects using data from file.
* The class *ChemicalFactoryImpl2* generates *ChemicalObserver* objects using data from file.
* The class *ReactionAdapterDB*, implementing interface *ReactionAdapter*, provides receiving information about chemical reaction from MySQL database.
* The class *ReactionAdapterFile*, implementing interface *ReactionAdapter*, provides receiving information about chemical reaction from text file.
* The enumeration *Tasks* is a set of tasks.
* The enumeration *Marker* is a set of chemical classes.
* The class *QuitException* extends *Exception*. Here, it allows to quit the program.

***Интерфейсы***

* The interface *Chemicable* allows receiving information about used chemical reagents
* The interface *ChemicalObserver* extends *Observer* and *Chemicable* It allows operations with chemical reagents
* The interface *Observer* allows receiving information from interface *Observerable*
* The interface *Observerable* allows informing objects from extending interface *ChemicalObserver* and changing some fields
* The interface *State* is a GoF patterns. Here, it allows different behaviour of chemicals upon different exposures.
* The interface *ReactionAdapter* allows receiving information about chemical reaction
* The interface *Factory* is a GoF patterns. Here, it allows generation of chemicals

сама UML-диаграмма в файле UML.pdf

***Использованные шаблоны программирования:***

* Абстрактная фабрика (интерфейс Factory и его реализации)
* Прототип (копирующий конструктор в классе Substance)
* Адаптер (интерфейс ReactionAdapter и его реализации)
* Состояние (интерфейс State и его реализация внутренними классами класса Substance)
* Стратегия — использование функционального интерфейса Consumer в методе notifyObservers() класса Reactor.
* Наблюдатель — объекты класса Substance получают сообщения от класса Reactor.