**Университет ИТМО**

**Факультет программной инженерии и компьютерных технологий**

**Лабораторная работа №5 по Программированию**

**Выполнил: Богатов Александр Сергеевич**

**Группа: P3133**

**Вариант: 312418**

**Преподаватель: Исаев Александр Сергеевич**

**Санкт-Петербург**

**2021**

**Задание:**

Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса Worker, описание которого приведено ниже.

Разработанная программа должна удовлетворять следующим требованиям:

* Класс, коллекцией экземпляров которого управляет программа, должен реализовывать сортировку по умолчанию.
* Все требования к полям класса (указанные в виде комментариев) должны быть выполнены.
* Для хранения необходимо использовать коллекцию типа java.util.HashSet
* При запуске приложения коллекция должна автоматически заполняться значениями из файла.
* Имя файла должно передаваться программе с помощью: **аргумент командной строки**.
* Данные должны храниться в файле в формате csv
* Чтение данных из файла необходимо реализовать с помощью класса java.io.BufferedInputStream
* Запись данных в файл необходимо реализовать с помощью класса java.io.PrintWriter
* Все классы в программе должны быть задокументированы в формате javadoc.
* Программа должна корректно работать с неправильными данными (ошибки пользовательского ввода, отсутсвие прав доступа к файлу и т.п.).

В интерактивном режиме программа должна поддерживать выполнение следующих команд:

* help : вывести справку по доступным командам
* info : вывести в стандартный поток вывода информацию о коллекции (тип, дата инициализации, количество элементов и т.д.)
* show : вывести в стандартный поток вывода все элементы коллекции в строковом представлении
* add {element} : добавить новый элемент в коллекцию
* update id {element} : обновить значение элемента коллекции, id которого равен заданному
* remove\_by\_id id : удалить элемент из коллекции по его id
* clear : очистить коллекцию
* save : сохранить коллекцию в файл
* execute\_script file\_name : считать и исполнить скрипт из указанного файла. В скрипте содержатся команды в таком же виде, в котором их вводит пользователь в интерактивном режиме.
* exit : завершить программу (без сохранения в файл)
* add\_if\_min {element} : добавить новый элемент в коллекцию, если его значение меньше, чем у наименьшего элемента этой коллекции
* remove\_greater {element} : удалить из коллекции все элементы, превышающие заданный
* remove\_lower {element} : удалить из коллекции все элементы, меньшие, чем заданный
* count\_by\_status status : вывести количество элементов, значение поля status которых равно заданному
* print\_ascending : вывести элементы коллекции в порядке возрастания
* print\_unique\_organization : вывести уникальные значения поля organization всех элементов в коллекции

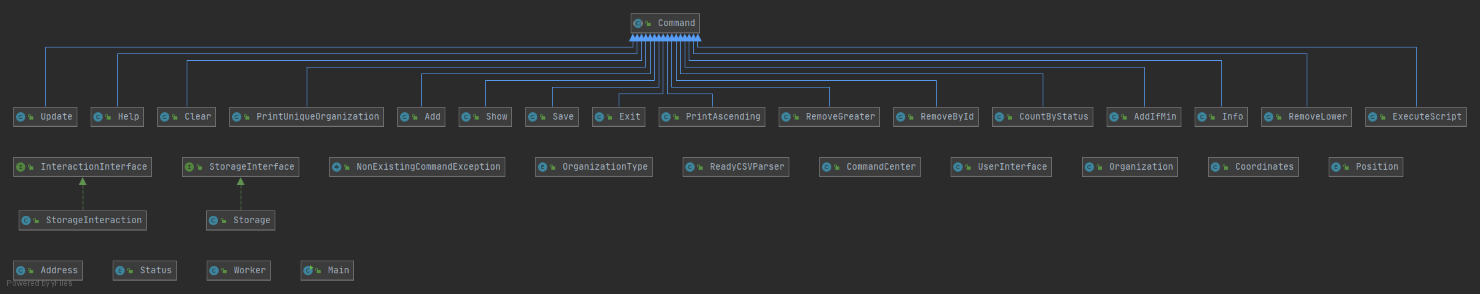
Формат ввода команд:

* Все аргументы команды, являющиеся стандартными типами данных (примитивные типы, классы-оболочки, String, классы для хранения дат), должны вводиться в той же строке, что и имя команды.
* Все составные типы данных (объекты классов, хранящиеся в коллекции) должны вводиться по одному полю в строку.
* При вводе составных типов данных пользователю должно показываться приглашение к вводу, содержащее имя поля (например, "Введите дату рождения:")
* Если поле является enum'ом, то вводится имя одной из его констант (при этом список констант должен быть предварительно выведен).
* При некорректном пользовательском вводе (введена строка, не являющаяся именем константы в enum'е; введена строка вместо числа; введённое число не входит в указанные границы и т.п.) должно быть показано сообщение об ошибке и предложено повторить ввод поля.
* Для ввода значений null использовать пустую строку.
* Поля с комментарием "Значение этого поля должно генерироваться автоматически" не должны вводиться пользователем вручную при добавлении.

Описание хранимых в коллекции классов:

1. **public** **class** Worker {
2. **private** **long** id; //Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически
3. **private** String name; //Поле не может быть null, Строка не может быть пустой
4. **private** Coordinates coordinates; //Поле не может быть null
5. **private** java.time.ZonedDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически
6. **private** Integer salary; //Поле не может быть null, Значение поля должно быть больше 0
7. **private** java.time.LocalDate endDate; //Поле может быть null
8. **private** Position position; //Поле может быть null
9. **private** Status status; //Поле может быть null
10. **private** Organization organization; //Поле может быть null
11. }
12. **public** **class** Coordinates {
13. **private** **int** x; //Максимальное значение поля: 627
14. **private** **long** y; //Максимальное значение поля: 990
15. }
16. **public** **class** Organization {
17. **private** Long annualTurnover; //Поле может быть null, Значение поля должно быть больше 0
18. **private** OrganizationType type; //Поле может быть null
19. **private** Address postalAddress; //Поле не может быть null
20. }
21. **public** **class** Address {
22. **private** String street; //Поле не может быть null
23. **private** String zipCode; //Поле не может быть null
24. }
25. **public** **enum** Position {
26. LABORER,
27. ENGINEER,
28. CLEANER;
29. }
30. **public** **enum** Status {
31. FIRED,
32. HIRED,
33. RECOMMENDED\_FOR\_PROMOTION,
34. REGULAR,
35. PROBATION;
36. }
37. **public** **enum** OrganizationType {
38. COMMERCIAL,
39. PRIVATE\_LIMITED\_COMPANY,
40. OPEN\_JOINT\_STOCK\_COMPANY;
41. }

**Диаграмма классов объектной модели:**



**Исходный код программы:**

<https://github.com/albogatov/lab5>

**Результат работы программы:**

**Вывод:**