**Техническое задание.**

**Тема РГР:** Управление проектами

**Исполнитель:** Кислицын Александр Иванович

**Группа:** ФИ101

**Номер зачетной книжки:** 212090

**Направление:** 02.03.02, «Фундаментальная информатика и информационные технологии»

**Профиль:** «Инженерия программного обеспечения»

**Описание проекта, общие положения.**

Исходя из темы проекта было принято решение создания некоторого аналога проектной программы, которая будет использоваться внутри компании.

Примерная структура была взята из IBM Lotus Notes, которой я пользуюсь на своей работе. Там существуют базы: инциденты, претензии. По их образу был разработан прототип данной РГР.

Программа включает в себя класс компании (предполагается возможность создания нескольких объектов компании, если бы хостинг серверов программы был на стороне разработчика ПО), класс сотрудника, класс проекта.

**Описание классов.**

Class EmployeeBase является классом компании, содержит переменные длины списка сотрудников, динамический массив класса Employee (см. ниже) – список сотрудников.

Содержит следующие методы:

* .firstEmployee() – создание первого сотрудника;
* .firstEmployee(ifstream \*f) - перегруженный метод для создания первого сотрудника при чтении из файла
* .newEmployee() - внесение сотрудника в базу, когда сотрудников > 1
* .init() - инциализация первого сотрудника, либо же нового (только по вызову пользователя, во время сессии)
* .void enter() - метод входа в аккаунт сотрудника
* .write\_to\_file() - метод записи данных в файл, вызывается в конце работы программы

Class Employee – класс сотрудника, содержит private login и password сотрудника, public name, department (отдел сотрудника) а также список проектов UserProject (см. ниже).

Содержит следующие методы:

* .firstProject() - создание первого проекта
* .firstProject(ifstream \*f) - перегруженный метод создания первого проекта, используется только пи чтении данных из файла
* .newProject() - внесение проекта в базу, когда проектов >= 1
* .newProject(ifstream \*f) - перегруженный метод создания списка проектов, когда проектов > 1, используется только пи чтении данных из файла
* .removeProjectList(int const \*number) - метод удаления любого номера проекта из списка
* .setInf() - метод, задающий параметры сотрудника
* .setInf(ifstream \*f) - перегруженный метод задания параметров сотрудника, при чтении из файла
* .check(string const \*l, string const \*p) - метод, отвечающий за проверку введеных имени, логина и пароля на корректность
* .userAccount() - метод пользовательского аккаунта, который дает доступ в аккаунт сотрудника
* .read\_file(ifstream \*f) - прикладной метод для чтения данных из файла, вызывается внутри конструктора класса EmployeeBase, работает только в начале программы

Class UserProject – класс, представляющий проекты сотрудника, их количество, содержание, название, имя и отдел создателя проекта. По умолчанию все поля “NULL”. Содержит методы:

* .createProject(string const \*n, string const \*d) - метод создания проекта, он вызывается только в случае создания проекта, вызванного пользователем во время работы программы
* .createProject(ifstream \*f, string const \*n, string const \*d) - перегруженный метод создания проекта, используется при запуске программы, читает файл
* .getProjectList(int const \*number) - метод вывода информации об имеющемся проекте, number отвечает за номер проекта

**Минусы проекта.**

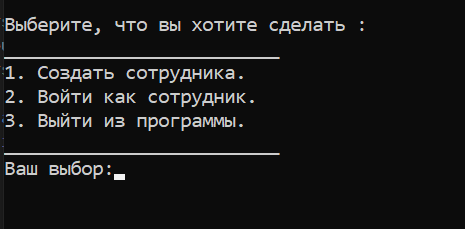
1. Оптимизация
   1. Каждый при завершении работы вызывается .write\_to\_file(), который полностью перезаписывает файл. Необходимо создать отдельные методы/функции для оптимизированной записи данных. Но, возможно, это недостаток исключительно формата .txt. Возможно, в .xlsx удалось бы этого избежать;
2. Спагетти – код
   1. Много методов для создания сотрудника/проекта. Стоит сделать единый метод создания сотрудника/проекта вместо .firstProject(), .newProject() и т.д.;
   2. Загруженная система чтения данных из файла. Идет множественное вложение в различные методы других классов. Однако это обеспечивает взаимодействие и целостность каждого класса, с возможностью реализации protected и private данных и методов. В противном случае, пришлось бы все параметры классов делать public, что являлось бы небезопасным.

**Планы на доработку проекта.**

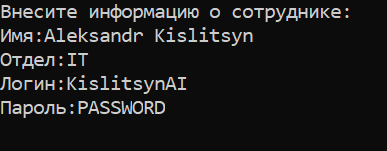
1. Реализовать более практичные методы записи (Минусы проекта, пункт 2, подпункт «a»);
2. Добавить графический интерфейс через фреймворк Qt;
3. Убрать запись данных в .txt, использовать таблицы, либо SQL.

**Скриншоты работы программы.**

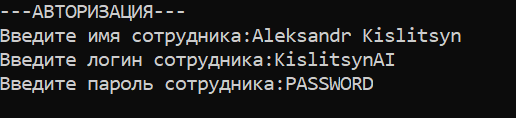
**Меню**

****

**Создание сотрудника**

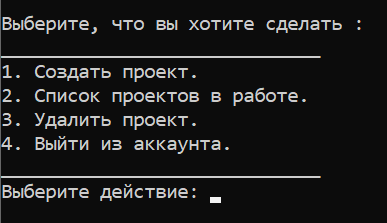
****

**Вход в аккаунт**

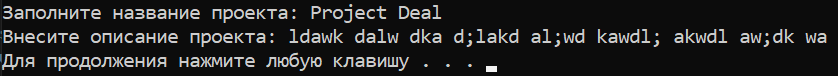
****

****

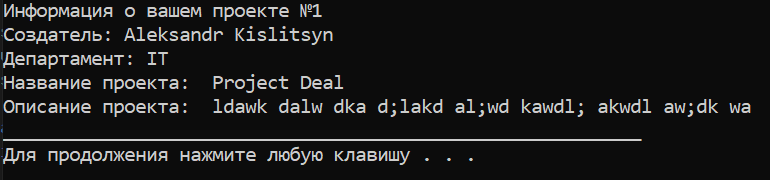
**Меню сотрудника**

****

**Создание проекта**

****

**Просмотр списка проектов, когда проекты есть**

****

**Удаление проекта, когда проекты есть**

****

**Удаление проекта, когда проектов нет**

****

**Просмотр списка проектов, когда проектов нет**

****