**РАЗРАБОТАННЫЕ КЛАССЫ**

1. Класс User (рисунок 1) содержит следующие методы:

– выод меню (реализован для каждого наследника User);

– просмотр всех билетов, доступных пользователю;

– методы для доступа к соответствующим сокрытым полям – геттеры и сеттеры;

– методы для поиска билетов по заданным параметрам: город, место проведения, название;

– просмотр всех билетов, отсортированных по полю «Номер»;

– просмотр всех билетов, отсортированных по полю «Город»;

– просмотр всех билетов, отсортированных по полю «Название»;

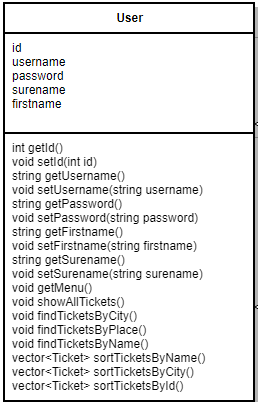


Рисунок 1 – Описание класса User

2. Класс Admin (рисунок 2) наследует все методы класса User, но также содержит свои:

– просмотр всех пользователей;

– просмотр заказанных билетов;

– добавление нового билета в базу;

– редактирование билета;

– удаление билета;

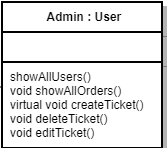


Рисунок 2 – Описание класса Admin

3. Класс Customer (рисунок 3) так же наследует все поля и методы класса User, но также содержит свои методы:

– показать товар;

– сделать заказ;

– показать ранее сделанные заказы.

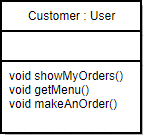


Рисунок 3 – Описание класса Customer

4. Класс App (рисунок 4) отвечает за доступ к данным и предоставляет собой интерфейс, содержащий следующие статические методы:

– получить ссылку на коллекцию объектов Admin;

– получить ссылку на коллекцию объектов Customer;

– получить ссылку на коллекцию объектов Ticket;

– получить ссылку на коллекцию объектов Order;

– получить объект Customer по id;

– получить объект Ticket по id;

– удалить объект Ticket из базы по id;

и поля:

* коллекция администртотров;
* коллекция пользователей;
* коллекция билетов;
* коллекция заказов;

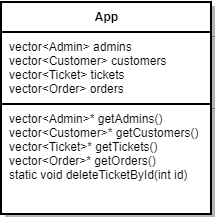


Рисунок 4 – Описание класса App

5. Класс Ticket (рисунок5) представляет собой модель билета, содержит следующие скрытые поля:

– идентификационный номер;

– город проведения мероприятия;

– место проведения;

– адрес

– название мероприятия

– время проведения

– цена одного билета

и методы для доступа к ним (геттеры и сеттеры). А также метод отображения:

– преобразовать в строку;

– вывести на экран, как строку таблицы.

Также иммет набор статических методов для сравнения объектов, используемых при сортировке.

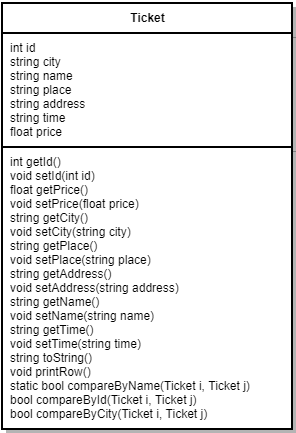


Рисунок 5 – Описание класса Ticket

6. Класс Order (рисунок 6) представляет собой модель заказа, содержит следующие скрытые поля:

– идентификационный номер;

– идентификационный номер заказчика;

– идентификационный номер билета;

– дата и время создания заказа;

– количество купленных билетов;

и методы для доступа к ним (геттеры и сеттеры). А также метод

– подсчет итоговой стоимости заказа.

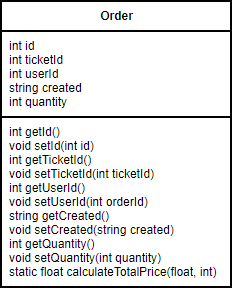


Рисунок 6 – Описание класса Order

7. Класс InOutUtils (рисунок 7) отвечает за чтение и запись информации в/из файла.

Содержит статические методы:

- для чтения списка администраторов из файла;

- записи списка администраторов в файл;

- для чтения списка пользователей из файла;

- записи списка пользователей в файл;

- для чтения списка билетов из файла;

- записи списка билетов в файл;

- для чтения списка заказов из файла;

- записи списка заказов в файл.

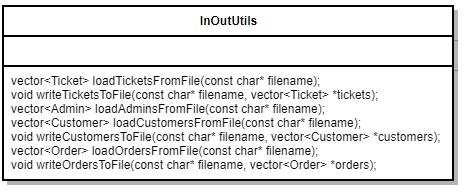


Рисунок 7 – Описание класса InOutUtils

Вся диаграмма классов (рисунок 8) имеет вид:

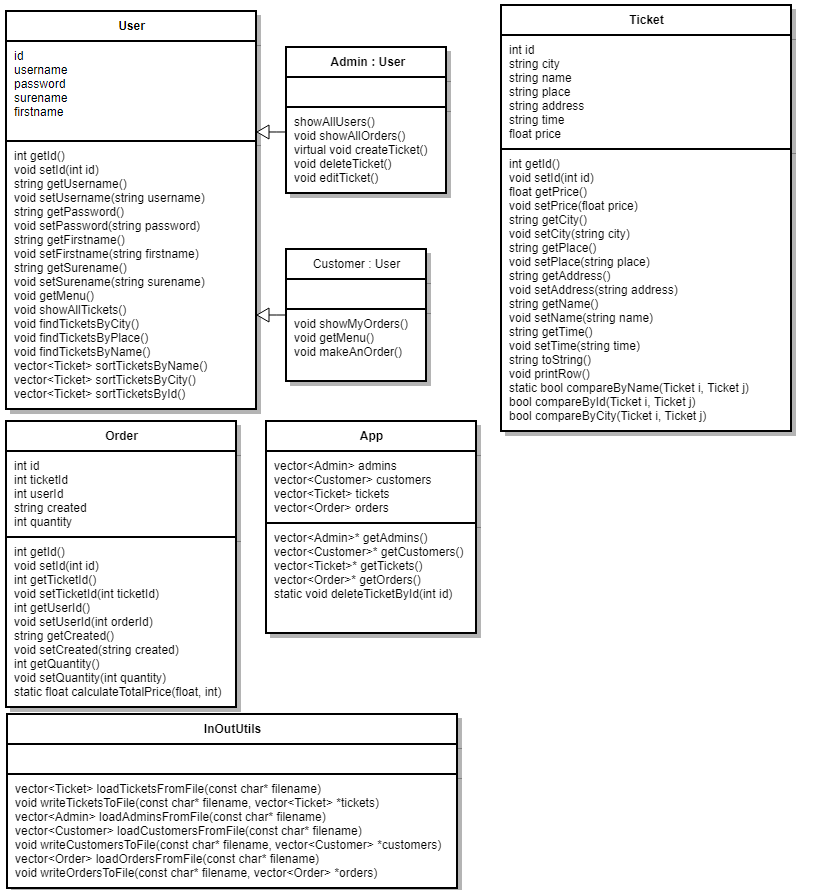


Рисунок 8 – Диаграмма классов проекта

**РАБОТА ПРОГРАММЫ**

При входе в программу возможно войти как администратор или пользователь (9).

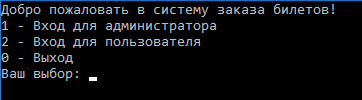


Рисунок 9 – Меню авторизации

При попытке войти в систему, программа требует логин и пароль для авторизации (10).

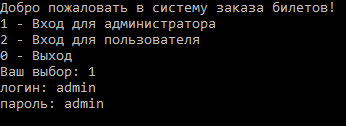


Рисунок 10 – Авторизация пользователя

После удачной авторизации в качестве администратора, перед нами окажется меню администратора, показывающее возможные операции, которые можно совершить. Разберём функционал программы на примере добавления записи в базу данных (рисунок 11).

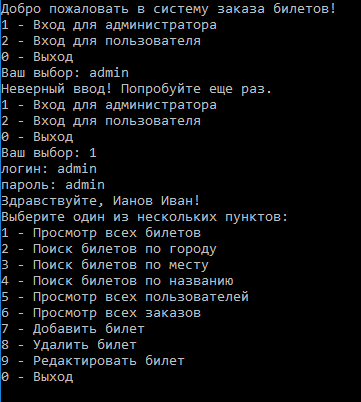


Рисунок 11 – Меню администратора

Одна из возможностей администратора – добавление нового билета в базу данных системы (рисунок 12).

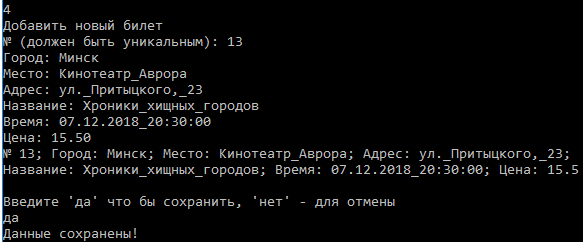


Рисунок 12 – Создание нового билета

При вводе некорректных данных программа попросит ввести данные повторно. Пока не будуд введены допустимые данные или не осуществлён ввод, программа будет запрашивать информацию (рисунок 13).

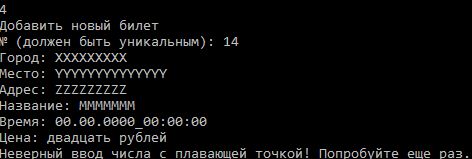


Рисунок 13– Ввод вида

При соблюдении всех условий, программа добавит новую запись в базу данных, откуда её смогут увидеть администратор и пользователи и проводить над ней соответствующие операции.

Рассмотрим так же некоторые операции, совершаемые пользователем или администратором.

Для поиска всех билетов по заданному городу, необходимо воспользоваться 2 пунктом меню (рисунок 14)

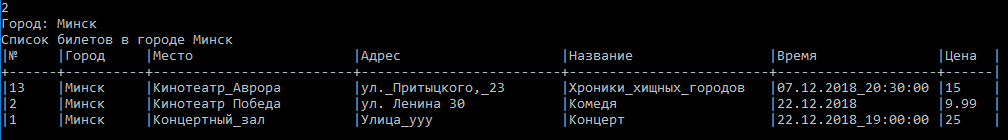


Рисунок 14 – Поиск билетов по городу

Для просмотра всех билетов отсортированных по заданному полю, необходимо воспользоваться 1 пунктом меню (рисунок 15)

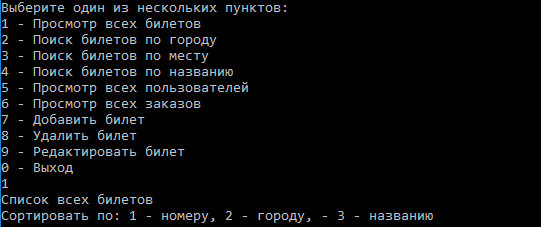


Рисунок 15 – Просмотр билетов в упорядоченном виде

Далее необходимо выбрать, по какому именно полю будут отсортированы записи. Отсортированный по названию список изображен на рисунке 16.

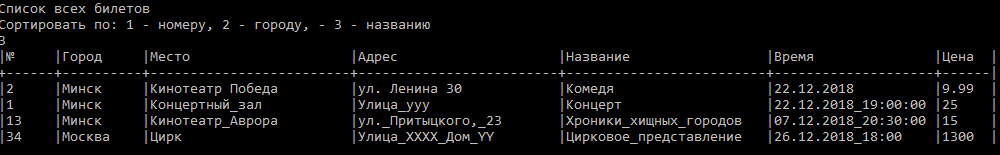


Рисунок 16 – Просмотр билетов в упорядоченном виде

Теперь рассмотрим операцию, специфичную только для пользователя (покупателя) – заказ билетов. Что бы сделать заказ нужно воспользоваться пунктом 6 меню пользователя. Необходимо ввести номер билета, после чего программа проверит существование билета с таким номером (рисунок 17). Если же билет с указанным номером существует, то пользователю будет предложено ввести количество заказываемых билетов (рисунок 18). После этого будет показана информация о заказе, вкючая конечную стоимость, и пользователь может подтвердить заказ или отменить (рисунок 19).

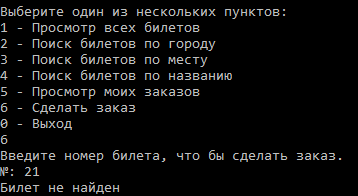


Рисунок 17 – Заказ билета. Ввод номера

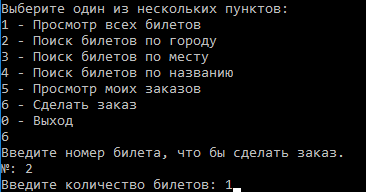


Рисунок 18 – Заказ билета. Количество билетов

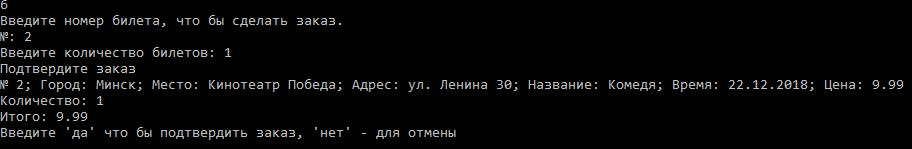


Рисунок 19 – Заказ билета. Подтверждение